CCA-Secure PRE Scheme without Public Verifiability

Jun Shao,[1]* Peng Liu[2] and Jian Weng[3]

[1] College of Computer and Information Engineering,
Zhejiang Gongshang University
[2] College of Information Sciences and Technology
Pennsylvania State University
[3] Department of Computer Science
Jinan University
chn.junshao@gmail.com, pliu@ist.psu.edu, cryptjweng@gmail.com

Abstract. In a proxy re-encryption (PRE) scheme, a semi-trusted proxy can transform a ciphertext under Alice's public key into another ciphertext that Bob can decrypt. However, the proxy cannot access the plaintext. Due to its transformation property, PRE can be used in many applications, such as encrypted email forwarding. *All* the existing CCA-secure PRE schemes have a crucial property: the public verifiability of the original ciphertext, i.e., everyone can check the validity of the original ciphertext. In this paper, we propose a novel CCA-secure PRE scheme *without* public verifiability. This proposal is proven-secure based on the DDH assumption in the standard model. To the best of our knowledge, our proposal is the *first* CCA-secure unidirectional PRE scheme without pairings in the standard model, which answers an open problem in the PRE field.

Keywords: CCA Security, Without Pairings, Without Public Verifiability, PRE

1 Introduction

Proxy re-encryption (PRE), introduced by Blaze, Bleumer and Strauss at EUROCRYPT 1998 [4], allows a semi-trusted proxy, with some additional information (a.k.a., re-encryption key), to transform a ciphertext under Alice's public key into a new ciphertext under Bob's public key on the same message. However, the proxy cannot learn any information about the messages encrypted under the public key of either Alice or Bob.

Generally speaking, there are two main methods to classify PRE schemes. One method is according to the direction of transformation. If the re-encryption key allows the proxy to transform from Alice to Bob, and vice versa, the PRE scheme is *bidirectional*; otherwise, it is *unidirectional.* The other method is according to the times of transformation. If the ciphertext can be transformed from Alice to Bob, and then from Bob to Charlie, and so on, the PRE scheme is *multi-use*; otherwise, it is *single-use.*

Due to its specific transformation property, PRE can be used in many applications, including simplification of key distribution [4], distributed file systems [2,3], security in publish/subscribe systems [18], multicast [8], secure certified email mailing lists [19,17], interoperable architecture of DRM [27], access control [28], and privacy for public transportation [16]. We refer the reader to [1] for the full list.

Since the concept of PRE was proposed, many PRE schemes have been presented. The first (bidirectional) PRE scheme was proposed by Blaze *et al.* [4] based on ElGamal public key encryption [13]. However, their scheme suffers from collusion attacks, i.e., Alice (Bob) can collude with

* Part of this work was finished when Jun Shao was with Pennsylvania State University.

the proxy to reveal Bob's (Alice's) secret key. To resist the collusion attack, Ateniese *et al.* [2,3] proposed several unidirectional PRE schemes by using pairings. Recently, Gentry [14] proposed a novel unidirectional PRE scheme based on homomorphic encryption.

Nevertheless, the above PRE schemes are only CPA-secure, while many PRE applications requires CCA-secure PRE [7]. There are two kinds of CCA-secure PRE schemes: with pairings and without pairings.

By using pairings and the CHK paradigm [6], Canetti and Hohenberger [7] proposed the first CCA-secure (bidirectional) PRE scheme in the standard model. However, their scheme suffers from collusion attacks. Furthermore, they didn't propose any CCA-secure unidirectional PRE scheme, but left it as an open problem. Note that a bidirectional scheme can always be implemented by a unidirectional one with two directions, while the converse is unknown. Based on Canetti-Hohenberger technique, Libert and Vergnaud [22] proposed a new unidirectional PRE scheme which is replayable chosen ciphertext attack (RCCA) secure and collusion resistant in the standard model. Note that RCCA security is weaker than CCA security, since it disallows the adversary to query the decryption oracle with the ciphertext whose corresponding message is one of the challenge messages in the RCCA security model [22], while only the derivatives[1] of the challenge ciphertext are disallowed in the CCA security model [7]. See the details in Remark 2 (Section 3). Recently, Weng et al. [29] and Shao et al. [25] proposed CCA-secure and collusion-resistant unidirectional PRE schemes by improving Libert and Vergnaud's construction.

Due to the heavy cost of pairing computation, it is desired to design CCA-secure PRE schemes without pairings. Shao and Cao [24], and Chow *et al.* [9] proposed CCA-secure and collusion-resistant unidirectional PRE schemes without pairings. However, their schemes are only proven-secure in the random oracle model. Most recently, Matsuda et al. [23] proposed a new CCA-secure (bidirectional) PRE scheme, which is proven-secure in the standard model but without pairings. However, it suffers from the collusion attack. Furthermore, Weng and Zhao [30] pointed out that Matsuda et al.'s scheme is not CCA-secure, but did not give any improvement.

Although the concept of PRE was proposed in 1998, designing a unidirectional PRE scheme without pairings but proven-secure in the standard model is still an open problem in the PRE field.

All the above CCA-secure PRE schemes have a crucial property: public verifiability of the original ciphertext, which prevents the proxy from acting as an oracle. However, this property is the obstacle to obtain PRE without pairings but proven-secure in the standard model. In this paper, we propose the *first* such PRE scheme by removing the public verifiability. Note that we only focus on the single-use, unidirectional PRE in this paper.

We put the previous CCA-secure PRE schemes in chronological order in Table 1.

Table 1. Comparison between CCA-secure PRE schemes.

	CH07[7]	SC09[24]	WCY$^+$10[29]	CWY$^+$10[9]	SCL10[25]	MNT10[23]	Ours
Unidirectional	✗	✓	✓	✓	✓	✗	✓
Standard Model	✓	✗	✓	✗	✓	✓	✓
Without Pairings	✗	✓	✗	✓	✗	✓	✓
Without Public Verifiability	✗	✗	✗	✗	✗	✗	✓

[1] See the definition in Section 3.1.

2 Organization

In the rest of this paper, we first review the definitions for single-use unidirectional PRE. After that, we present our proposal and give its security proof in the standard model. In what follows, we give the comparison between our proposal and the scheme in [29][2] in terms of computational cost and ciphertext length. Finally, the conclusion is given.

3 Preliminaries

In this section, we review some basic knowledge of PRE.

3.1 Definitions for Single-Use Unidirectional PRE

The similar definitions can be found in [2,3,15,7,24].

Definition 1 (Single-Use Unidirectional PRE). *A single-use unidirectional proxy re-encryption scheme* SUPRE *is a tuple of PPT algorithms (KeyGen, ReKeyGen, Enc, ReEnc, Dec):*

- *KeyGen*$(1^k) \rightarrow (pk, sk)$*. On input the security parameter* 1^k*, the key generation algorithm KeyGen outputs a public key* pk *and a secret key* sk*.*
- *ReKeyGen*$(sk_1, pk_2) \rightarrow rk_{1,2}$*. On input a secret key* sk_1 *and a public key* pk_2*, the re-encryption key generation algorithm ReKeyGen outputs a unidirectional re-encryption key* $rk_{1,2}$*.*[3]
- *Enc*$(pk, m) \rightarrow C$*. On input a public key* pk *and a message* m *in the message space, the encryption algorithm Enc outputs a ciphertext* C*.*
- *ReEnc*$(rk_{1,2}, C_1) \rightarrow C_2$*. On input a re-encryption key* $rk_{1,2}$ *and a ciphertext* C_1*, the re-encryption algorithm ReEnc outputs a re-encrypted ciphertext* C_2 *or a special symbol reject.*
- *Dec*$(sk, C) \rightarrow m$*. On input a secret key* sk *and a ciphertext* C*, the decryption algorithm Dec outputs a message* m *in the message space or a special symbol reject.*

Correctness. The correctness property has two requirements. For any message m in the message space and any key pairs $(pk, sk), (pk', sk') \leftarrow \texttt{KeyGen}(1^k)$. Then the following two conditions must hold:

$$\texttt{Dec}(sk, \texttt{Enc}(pk, m)) = m,$$
$$\texttt{Dec}(sk', \texttt{ReEnc}(\texttt{ReKeyGen}(sk, pk'), C)) = m,$$

where C is the ciphertext for message m under pk from algorithm Enc.

Remark 1 (Two types of ciphertexts). In almost all the existing unidirectional proxy re-encryption schemes[4], there are two types of ciphertexts. One is the *first-level* ciphertext, which could be generated from ReEnc (or Enc)[5]; the other is the *second-level* ciphertext, which is generated only from Enc. In this paper, we call the first-level ciphertext and second-level ciphertext as the *re-encrypted* ciphertext and *original* ciphertext, respectively.

[2] The scheme in [29] is more efficient than the scheme in [25].
[3] There are two kinds of ReKeyGen. If the delegatee is involved in ReKeyGen, then it is interactive; otherwise, it is non-interactive. In this paper, we only consider the non-interactive ReKeyGen.
[4] The except one is the scheme proposed by Gentry [14].
[5] The first-level ciphertext is computed only from ReEnc in some PRE schemes, such as [22], while it can also be computed from Enc in other PRE schemes, such as [24].

3.2 Security Notions for SUPRE

Chosen Ciphertext Security for the Original Ciphertext of SUPRE. The CCA security for the original ciphertext of SUPRE is defined by the following CCA-O game played between a challenger $\mathcal{C}$ 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 input into the oracles by the adversary are all from $\mathcal{O}_{pk}$.

Setup: The challenger sets up the system parameters.

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

- *Public key generation oracle* $\mathcal{O}_{pk}$: On input an index i,[6] the challenger takes a security parameter k, and responds by running algorithm $\texttt{KeyGen}(1^k)$ 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 input pk_i by $\mathcal{A}$, where pk_i is from $\mathcal{O}_{pk}$, if pk_i is corrupted, the challenger searches pk_i in the table T_K and returns sk_i; otherwise, the challenger returns `reject`.
- *Re-encryption key generation oracle* $\mathcal{O}_{rk}$: On input (pk_i, pk_j) by $\mathcal{A}$, where pk_i, pk_j are from $\mathcal{O}_{pk}$, the challenger returns the re-encryption key $rk_{i,j} = \texttt{ReKeyGen}(sk_i, pk_j)$, where sk_i is the secret key corresponding to pk_i. Note that pk_i should be uncorrupted. For the case that pk_i is corrupted, the adversary can compute the re-encryption key by itself.
- *Re-encryption oracle* $\mathcal{O}_{re}$: On input (pk_i, pk_j, C) by $\mathcal{A}$, where pk_i, pk_j are from $\mathcal{O}_{pk}$, the challenger returns the re-encrypted ciphertext $C' = \texttt{ReEnc}(\texttt{ReKeyGen}(sk_i, pk_j), C)$, where sk_i is the secret key corresponding to pk_i. Note that pk_i should be uncorrupted. For the case that pk_i is corrupted, the adversary can compute the re-encrypted ciphertext by itself.
- *Decryption oracle* $\mathcal{O}_{dec}$: On input (pk_i, C_i), where pk_i is from $\mathcal{O}_{pk}$, the challenger returns $\texttt{Dec}(sk_i, C_i)$, where sk_i is the secret key corresponding to pk_i. Note that pk_i should be uncorrupted. For the case that pk_i is corrupted, the adversary can decrypt the ciphertext by itself.

Challenge: Once $\mathcal{A}$ decides that Phase 1 is over, it outputs two equal length plaintexts m_0, 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 $C^* = \texttt{Enc}(pk^*, m_{\mathbf{b}})$. 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:

- $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$: The challenger responds as in Phase 1.
- $\mathcal{O}_{rk}$: On input (pk_i, pk_j) by $\mathcal{A}$, if $pk_i = pk^*$, and pk_j is corrupted, the challenger outputs `reject`; otherwise, the challenger responds as in Phase 1.
- $\mathcal{O}_{re}$: On input (pk_i, pk_j, C_i) by $\mathcal{A}$, if $(pk_i, C_i) = (pk^*, C^*)$ and pk_j is corrupted, the challenger outputs `reject`; otherwise, the challenger outputs `reject`.
- $\mathcal{O}_{dec}$: On input (pk_i, C_i), if (pk_i, C_i) is a `derivative` of (pk^*, C^*), the challenger outputs `reject`; 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 $\mathbf{b} = \mathbf{b}'$.

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

We refer to such an adversary $\mathcal{A}$ as a CCA-O adversary. We define adversary $\mathcal{A}$'s advantage in attacking `SUPRE` as the following function of the security parameter k:

$$\mathbf{Adv}_{\texttt{SUPRE}}^{\texttt{CCA-O}}(k) = |\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|.$$

Using the CCA-O game, we can define CCA-O security of SUPRE.

Definition 2 (CCA-O Security). *We say that a single-use, unidirectional proxy re-encryption scheme* `SUPRE` *is semantically secure against an adaptive chosen ciphertext attack for the original ciphertext if for any polynomial time CCA-O adversary $\mathcal{A}$ the function $\mathbf{Adv}_{\texttt{SUPRE}}^{\texttt{CCA-O}}(k)$ is negligible. As shorthand, we say that* `PRE` *is CCA-O-secure.*

Remark 2. Derivatives of (pk^*, C^*) is adapted from that in [7]:

1. (pk^*, C^*) is a derivative of itself.
2. If $\mathcal{A}$ has queried $\mathcal{O}_{re}$ on input (pk, pk', C) and obtained (pk', C'), then (pk', C') is a derivative of (pk, C).
3. If $\mathcal{A}$ has queried $\mathcal{O}_{rk}$ on input (pk, pk'), and $C' = \texttt{ReEnc}(\mathcal{O}_{re}(pk, pk'), C)$, then (pk', C') is a derivative of (pk, C).

From the above, we know that if a re-encrypted ciphertext is not obtained *directly* by `ReEnc` or $\mathcal{O}_{re}$, then this ciphertext cannot be a derivative of any ciphertext. Hence, if $C'' = F(C')$, where F is a transformation function which makes $C'' \neq C'$ and $\texttt{Dec}(C'', pk) = \texttt{Dec}(C', pk)$, and $C' = \mathcal{O}_{re}(pk^*, pk', C^*)$, then C'' is not a derivative of (pk^*, C^*).

However, according to the definition of the derivative of (pk^*, C^*) for the RCCA security [22], the above C'' is a derivative of (pk^*, C^*). Because the derivative of (pk^*, C^*) in [22] is defined as: if the message of a re-encrypted ciphertext is one of the challenge messages, then this re-encrypted ciphertext is a derivative of (pk^*, C^*).

Chosen Ciphertext Security for the Re-encrypted Ciphertext of SUPRE. The CCA-R security of SUPRE is defined by the same method of the CCA-O security.

Phase 1, Guess: Identical to that in CCA-O game.

Challenge: Once the adversary $\mathcal{A}$ decides that Phase 1 is over, it outputs two equal length plaintexts m_0, m_1 from the message space, and two public key pk, pk^* on which it wishes to challenge. The public key pk^* should be uncorrupted. The challenger picks a random bit $\mathbf{b} \in \{0, 1\}$ and sets $C^* = \texttt{ReEnc}(rk, \texttt{Enc}(pk, m_{\mathbf{b}}))$, where rk is a re-encryption key from pk to pk^*. 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:
- $\mathcal{O}_{pk}$, $\mathcal{O}_{sk}$, $\mathcal{O}_{rk}$, $\mathcal{O}_{re}$: The challenger responds as in Phase 1.
- $\mathcal{O}_{dec}$: On input (pk_i, C_i), if $(pk_i, C_i) = (pk^*, C^*)$, the challenger outputs `reject`; otherwise, the challenger responds as in Phase 1.

We refer to such an adversary $\mathcal{A}$ as a CCA-R adversary. We define adversary $\mathcal{A}$'s advantage in attacking `SUPRE` as the following function of the security parameter k:

$$\mathbf{Adv}_{\texttt{SUPRE}}^{\texttt{CCA-R}}(k) = |\Pr[\mathbf{b} = \mathbf{b}'] - 1/2|.$$

Using the CCA-R game, we can define CCA-R security of SUPRE.

Definition 3 (CCA-R Security). *We say that a single-use, unidirectional proxy re-encryption scheme* SUPRE *is semantically secure against an adaptive chosen ciphertext attack for the re-encrypted ciphertext if for any polynomial time CCA-R adversary $\mathcal{A}$ the function $\mathbf{Adv}^{\mathit{CCA\text{-}R}}_{\mathit{SUPRE}}(k)$ is negligible. As shorthand, we say that* PRE *is CCA-R-secure.*

Remark 3. In [2,3], the authors proposed another security notion for single-use unidirectional PRE, named collusion resistance. This security notion guarantees that the delegatee and the proxy cannot collude to get the secret key of the delegator. As mentioned in [22], the security of collusion resistance is implied by the CCA-R security. Hence, we omit the part related to collusion resistance.

4 Scheme $\prod_{sm}$ in the Standard Model

4.1 Intuition Behind the Construction

The idea behind the construction of scheme $\prod_{sm}$ begins with Cramer-Shoup public key encryption [11,12], which is proven CCA-secure in the standard model. Recall Cramer-Shoup scheme. Its private key contains two parts. One is used for ciphertexts' validity test: (x_1, x_2, y_1, y_2), called verification key; the other is used for decryption: z, called decryption key. The corresponding public key is $(g_1, g_2, c = g_1^{x_1} g_2^{x_2}, d = g_1^{y_1} g_2^{y_2}, h = g_1^{z})$, where g_1, g_2 are generators of the underlying group. The ciphertext is $(u_1 = g_1^r, u_2 = g_2^r, e = h^r m, v = c^r d^{r\alpha})$, where r is a random number, $\alpha = H(u_1, u_2, e)$, and H is a target collision resistant hash function. Before computing m, the decryptor first checks the equation $v = u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha}$. If it holds, the decryptor computes $m = e/u_1^z$; otherwise, the decryptor outputs `reject`.

Obtain CCA Security. As mentioned in [15,7], it is crucial for designing a CCA-secure PRE scheme that 1) the malicious delegatee cannot gain any advantage by using the proxy as an oracle, and 2) the malicious proxy cannot gain any advantage by using the delegatee as an oracle.

– Regarding the first point, all the existing CCA-secure PRE schemes [7,22,24,10,25,29] require that the original ciphertext could be publicly verified.
  However, Cramer-Shoup scheme does not have such property. Inspired by [5,26], to overcome this obstacle, we let the malicious delegatee generate a *random* value if the original ciphertext is *not well-formed*; the message m otherwise. The following is a key part of our proposal.
  1. The private key of the delegator is shared between the proxy and the delegatee. For example, the proxy and the delegatee know $(\hat{x}_1, \hat{x}_2, \hat{y}_1, \hat{y}_2, \hat{z})$, and $(\breve{x}_1, \breve{x}_2, \breve{y}_1, \breve{y}_2, \breve{z})$, respectively. These values satisfy $\hat{x}_1 + \breve{x}_1 = x_1$, $\hat{x}_2 + \breve{x}_2 = x_2$, $\hat{y}_1 + \breve{y}_1 = y_1$, $\hat{y}_2 + \breve{y}_2 = y_2$, and $\hat{z} + \breve{z} = z$. Furthermore, the delegator chooses two random numbers k_1, k_2 from Z_q^*, and computes $K_1 = g_1^{k_1}$, $K_2 = g_1^{k_2}$, $A = g_1^{k_1\breve{x}_1 + r_2\breve{x}_2}$, and $B = g_1^{k_1\breve{y}_1 + r_2\breve{y}_2}$. The values of (K_1, K_2, A, B) are only sent to the proxy by the delegator.
  2. The proxy computes $\hat{e} = e/u_1^{\hat{z}} \cdot (u_1^{\hat{x}_1 - \hat{x}'_1 + \hat{y}_1\alpha} u_2^{\hat{x}_2 - \hat{x}'_2 + \hat{y}_2\alpha}/v)^{\hat{r}}$, $\hat{u}_1 = K_1^{\hat{r}'} \cdot u_1^{\hat{r}}$, $\hat{u}_2 = K_2^{\hat{r}'} \cdot u_2^{\hat{r}}$, and $K = (A \cdot B^{\alpha} \cdot K_1^{\hat{x}'_1} \cdot K_2^{\hat{x}'_2})^{\hat{r}'}$, where $\hat{x}'_1, \hat{x}'_2, \hat{r}, \hat{r}'$ are random numbers from Z_q^*. The values of $(\hat{e}, \hat{u}_1, \hat{u}_2, K, \hat{x}'_1, \hat{x}'_2)$ are included in the resulting re-encrypted ciphertext;
  3. The delegatee gets m from $m = \hat{e}/u_1^{\breve{z}} \cdot (\hat{u}_1^{\breve{x}_1 + \hat{x}'_1 + \breve{y}_1\alpha} \hat{u}_2^{\breve{x}_2 + \hat{x}'_2 + \breve{y}_2\alpha})/K$. The correctness could be found in Section 4.2.

Note that $K_i^{\hat{r}'}$ $(i = 1, 2)$ are used to hide the value of $R^{\hat{r}}$, where R is the value that could be computed by the malicious delegatee before receiving the re-encrypted ciphertext. Some examples of R are u_1, u_2 and $u_1^{\breve{x}_1+\breve{y}_1\alpha} u_2^{\breve{x}_2+\breve{y}_2\alpha}$. If the malicious delegatee can generate such an R, then he/she can change the challenge ciphertext (u_1^*, u_2^*, e^*, v^*) to $(u_1', u_2', e', v') = (u_1^*, u_2^*, e^*, v^* \cdot R)$, and get $\hat{e}^* = \hat{e}' \cdot R^{\hat{r}}$. At last, the malicious delegatee gets $m_{\mathbf{b}}$ by using his/her secret key.
Clearly, if the original ciphertext is un-well-formed, the delegatee would get a random m (since $(u_1^{x_1+y_1\alpha} u_2^{x_2+y_2\alpha}/v)^{\hat{r}} \neq 1$); otherwise, the delegatee can get the message m.
Nevertheless, in order to let the delegatee compute α, the re-encrypted ciphertext should also contain (u_1, u_2, e), which causes the above construction not CCA-R-secure. Since another delegatee and his/her corresponding proxy can collude to get the message m from (u_1, u_2, e) in the re-encrypted ciphertext, which breaks the CCA-R security. We follow the method in [25]: Encrypt (u_1, u_2, e) by using the delegatee's public key (which is different from the public key used for encrypting messages. If not, the delegator and proxy can still get (u_1, u_2, e) by using the transformation property of PRE).

– Regarding the second point, all the existing CCA-secure PRE schemes [29,25] in the standard model adopt the symmetric encryption (hash function) approach. In this paper, we follow this approach but with some modification. If the delegatee receives an *un-well-formed* re-encrypted ciphertext, then he/she outputs $\perp$ in [29,25], while he/she outputs a *random* number in our proposal.

Obtain Non-Interactivity. Unfortunately, the above construction is lack of one important property: non-interactivity. In the above construction, the private key of the delegator is shared between the proxy and the delegatee. This process is interactive.

To obtain the non-interactivity, we adopt the method in [15]. That is, $(\breve{x}_1, \breve{x}_2, \breve{y}_1, \breve{y}_2, \breve{z})$ is not sent to the delegatee directly, but it is encrypted by the delegatee's public key. The resulting ciphertext is sent to the proxy, who later forwards it to the delegatee with the re-encrypted ciphertext. It is required that the public key used to encrypt $(\breve{x}_1, \breve{x}_2, \breve{y}_1, \breve{y}_2, \breve{z})$ is different from the one used to encrypt messages; otherwise, the delegatee's delegatees and proxies can get $(\breve{x}_1, \breve{x}_2, \breve{y}_1, \breve{y}_2, \breve{z})$ and then the message m, which is not allowed in single-use PRE.

Reduce the Size of the Secret. It is easy to see that the delegatee and the proxy can collude to get the decryption key z, hence the required two different public keys should be corresponding to *two* different private keys. It means that the user in our PRE scheme have to maintain much more secrets than the one in Cramer-Shoup scheme.

To solve the problem, we use the following method. For the decryption key z, we find that there is a pseudo decryption key (z_1, z_2) in the security proof of Cramer-Shoup scheme, where (z_1, z_2) can be used to decrypt (u_1, u_2, e, u) and $g_1^z = g_1^{z_1} g_2^{z_2}$. According to the security proof of Cramer-Shoup scheme, knowing (z_1, z_2) won't hurt the secrecy of z. Hence, the delegator can share (z_1, z_2) instead of z between the delegatee and the proxy, while the resulting PRE scheme can still work. On the other hand, (z_1, z_2) cannot be used to decrypt the ciphertext computed by another public key $(g_1, \bar{g}_2, \bar{c}, \bar{d}, h)$.

For the verification key (x_1, x_2, y_1, y_2), we let them be computed from the verification key corresponding to $(g_1, \bar{g}_2, \bar{c}, \bar{d}, h)$ by some hash function. Due the properties of the hash function, knowing (x_1, x_2, y_1, y_2) do not hurt the secrecy of the verification key corresponding to $(g_1, \bar{g}_2, \bar{c}, \bar{d}, h)$.

As a result, the secret of the user in our PRE scheme only includes (z_1, z_2, z) and the verification key corresponding to $(g_1, \bar{g}_2, \bar{c}, \bar{d}, h)$.

Combining the above ideas, we obtain scheme $\prod_{sm}$, which is depicted in the next subsection. Note that in our proposal, we do not use the original Cramer-Shoup scheme, but the hashed version.

4.2 The Construction of Scheme $\prod_{sm}$

The system parameters are $(q, g_1, G, F, \breve{F}, \mathbf{F}, \bar{H}, H)$. G is a finite cyclic group with prime order q, g_1 is a generator of G. $F, \breve{F}, \mathbf{F}$ are cryptographically secure pseudo-random number generators (CSPRNGs), $F : \{0,1\}^* \to G$, $\breve{F} : \{0,1\}^* \to Z_q^{*6}$, $\mathbf{F} : \{0,1\}^* \to G^8 \times Z_q^{*6} \times G \times Z_q^{*2}$, $\bar{H}, H$ are target collision resistant hash functions $\bar{H} : Z_q^* \to Z_q^*$, and $H : \{0,1\}^* \to Z_q^*$. The details of our proposal $\prod_{sm}$ are as follows.

`KeyGen`**:** Select random

$$x_1, x_2, y_1, y_2, z_1, z_2, \beta_1, \beta_2, \in Z_q^*,$$

Next, compute

$$\begin{array}{lll} g_2 = g_1^{\beta_1}, & \bar{g}_2 = g_2^{\beta_2} = g_1^{\beta_1\beta_2}, & \\ c = g_1^{\bar{H}(x_1)} g_2^{\bar{H}(x_2)}, & d = g_1^{\bar{H}(y_1)} g_2^{\bar{H}(y_2)}, & h = g_1^z = g_1^{z_1} g_2^{z_2}, \\ \bar{c} = g_1^{x_1} \bar{g}_2^{x_2}, & \bar{d} = g_1^{y_1} \bar{g}_2^{y_2}. & \end{array}$$

The public key is $pk = (g_2, \bar{g}_2, c, d, h, \bar{c}, \bar{d})$, and the private key is $sk = (x_1, x_2, y_1, y_2, z_1, z_2, z)$.

Note that $h = g_1^{z_1} \bar{g}_2^{z_2/\beta_2}$, and $z = z_1 + \beta_1 \cdot z_2 \mod q$.

`ReKeyGen`**:** On input a public key $pk' = (g_2', \bar{g}_2', c', d', h', \bar{c}', \bar{d}')$ and a secret key $sk = (x_1, x_2, y_1, y_2, z_1, z_2, z)$, output a unidirectional re-encryption key $rk = (rk^{(1)}, rk^{(2)}, rk^{(3)})$, which is computed as follows.

- Choose random $(\hat{x}_1, \breve{x}_1, \hat{x}_2, \breve{x}_2, \hat{y}_1, \breve{y}_1, \hat{y}_2, \breve{y}_2, \hat{z}_1, \breve{z}_1, \hat{z}_2, \breve{z}_2, k_1, k_2)$ from Z_q^*, such that

$$\begin{array}{ll} \bar{H}(x_1) = \hat{x}_1 + \breve{x}_1, & \bar{H}(x_2) = \hat{x}_2 + \breve{x}_2, \\ \bar{H}(y_1) = \hat{y}_1 + \breve{y}_1, & \bar{H}(y_2) = \hat{y}_2 + \breve{y}_2, \\ z_1 = \hat{z}_1 + \breve{z}_1, & z_2 = \hat{z}_2 + \breve{z}_2, \\ K_1 = g_1^{k_1}, & K_2 = g_1^{k_2}, \\ A = g_1^{k_1\breve{x}_1 + k_2\breve{x}_2}, & B = g_1^{k_1\breve{y}_1 + k_2\breve{y}_2}. \end{array}$$

- Set $rk^{(1)} = (\hat{x}_1, \hat{x}_2, \hat{y}_1, \hat{y}_2, \hat{z}_1, \hat{z}_2, K_1, K_2, A, B)$.
- Choose random $\breve{r}$ from Z_q^*.
- Set $\triangle = \breve{x}_1 || \breve{x}_2 || \breve{y}_1 || \breve{y}_2 || \breve{z}_1 || \breve{z}_2$.
- Compute

$$\begin{array}{ll} \breve{u}_1 = g_1^{\breve{r}}, & \breve{u}_2 = (\bar{g}_2')^{\breve{r}}, \\ \breve{e} = \breve{F}(h'^{\breve{r}}) \oplus \triangle, & \breve{\alpha} = H(\breve{u}_1 || \breve{u}_2 || \breve{e}), \\ \breve{v} = \bar{c}'^{\breve{r}} \bar{d}'^{\breve{r}\breve{\alpha}}. & \end{array}$$

  The above process is almost the same as the encryption of Cramer-Shoup scheme with public key $(g_1, \bar{g}_2', \bar{c}', \bar{d}', h')$, except that $\breve{e}$ is computed by $\breve{F}(h'^{\breve{r}}) \oplus m$ instead of $h'^{\breve{r}} \cdot m$ due to the different message spaces.
- Set $rk^{(2)} = (\breve{u}_1, \breve{u}_2, \breve{e}, \breve{v})$.
- Set $rk^{(3)} = (S_1, S_2, S_3, S_4) = (\bar{g}_2', \bar{c}', \bar{d}', h')$.

`Enc`: On input $pk = (g_2, \bar{g}_2, c, d, h, \bar{c}, \bar{d})$ and a message $m \in G$, do the following steps.

- Choose random r from Z_q^*.
- Compute

$$\begin{array}{ll} u_1 = g_1^r, & u_2 = g_2^r, \\ e = F(h^r) \cdot m, & \alpha = H(u_1||u_2||e), \\ v = c^r d^{r\alpha}. & \end{array}$$

  The above process is almost the same as the encryption of Cramer-Shoup scheme with public key (g_1, g_2, c, d, h), except that e is computed by $F(h^r) \cdot m$ instead of $h^r \cdot m$.
- Output $C = (u_1, u_2, e, v)$ as the original ciphertext.

`ReEnc`: On input a re-encryption key rk and an original ciphertext $C = (u_1, u_2, e, v)$ under key $pk = (g_2, \bar{g}_2, c, d, h, \bar{c}, \bar{d})$, do the following steps.

- Choose random $\hat{r}$, $\hat{r}'$, $\hat{x}'_1$, $\hat{x}'_2$ and $\mathbf{r}$ from Z_q^*.
- Compute

$$\begin{array}{ll} \hat{u}_1 = K_1^{\hat{r}'} \cdot u_1^{\hat{r}}, & \hat{u}_2 = K_2^{\hat{r}'} \cdot u_2^{\hat{r}} \\ \hat{e} = \left(\dfrac{u_1^{\hat{x}_1 - \hat{x}'_1 + \hat{y}_1 \alpha} u_2^{\hat{x}_2 - \hat{x}'_2 + \hat{y}_2 \alpha}}{v} \right)^{\hat{r}} \cdot u_1^{\hat{z}_1} u_2^{\hat{z}_2}, & K = (A \cdot B^{\alpha} \cdot K_1^{\hat{x}'_1} \cdot K_2^{\hat{x}'_2})^{\hat{r}'}. \end{array}$$

- Set $\nabla = u_1||u_2||e||\hat{u}_1||\hat{u}_2||\hat{e}||\breve{u}_1||\breve{u}_2||\breve{e}||\breve{v}||x'_1||x'_2$.
- Compute

$$\begin{array}{ll} \mathbf{u}_1 = g_1^{\mathbf{r}}, & \mathbf{u}_2 = S_1^{\mathbf{r}}, \\ \mathbf{e} = \mathbf{F}(S_4^{\mathbf{r}}) \oplus \nabla, & \hat{\alpha} = H(\mathbf{u}_1||\mathbf{u}_2||\mathbf{e}), \\ \mathbf{v} = S_2^{\mathbf{r}} S_3^{\mathbf{r}\hat{\alpha}}. & \end{array}$$

  The above process is almost the same as the encryption of Cramer-Shoup scheme with public key $(g_1, \bar{g}'_2, \bar{c}', \bar{d}', h')$, except that $\mathbf{e}$ is computed by $\mathbf{F}(h'^{\mathbf{r}}) \oplus m$ instead of $h'^{\mathbf{r}} \cdot m$ due to the different message spaces.
- Output $\mathbf{C} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$ as the re-encrypted ciphertext.

`Dec`: On input a private key and any ciphertext C, parse C,

Case $C = (u_1, u_2, e, v)$: In this case, the private key is $(x_1, x_2, y_1, y_2, z_1, z_2, z)$.

1. Compute $\alpha = H(u_1||u_2||e)$.
2. If $v = u_1^{\bar{H}(x_1) + \bar{H}(y_1)\alpha} u_2^{\bar{H}(x_2) + \bar{H}(y_2)\alpha}$ holds, u_1, u_2, e belong to G, then output $m = e/F(u_1^z)$; otherwise output `reject`.

Case $\mathbf{C} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$: In this case, the private key is $(x'_1, x'_2, y'_1, y'_2, z'_1, z'_2, z')$.

1. Compute $\hat{\alpha} = H(\mathbf{u}_1||\mathbf{u}_2||\mathbf{e})$.
2. If $\mathbf{v} = \mathbf{u}_1^{x'_1 + y'_1 \hat{\alpha}} \mathbf{u}_2^{x'_2 + y'_2 \hat{\alpha}}$ and $\mathbf{u}_1, \mathbf{u}_2$ belong to G, then compute $\nabla = \mathbf{F}(\mathbf{u}_1^{z'}) \oplus \mathbf{e}$ and do the next steps; otherwise, output `reject` and halt.
3. Parse ∇ to $u_1||u_2||e||\hat{u}_1||\hat{u}_2||\hat{e}||\breve{u}_1||\breve{u}_2||\breve{e}||\breve{v}||\hat{x}'_1||\hat{x}'_2 \in G^8 \times Z_q^{*6} \times G \times Z_q^{*2}$.
4. Compute $\breve{\alpha} = H(\breve{u}_1||\breve{u}_2||\breve{e})$.
5. If $\breve{v} = \breve{u}_1^{x'_1 + y'_1 \breve{\alpha}} \breve{u}_2^{x'_2 + y'_2 \breve{\alpha}}$ and $\breve{u}_1, \breve{u}_2$ belong to G, then compute $\mathit{\Delta} = \breve{F}(\breve{u}_1^{z'}) \oplus \breve{e}$; otherwise, output `reject` and halt.
6. Parse $\mathit{\Delta}$ to $\breve{x}_1||\breve{x}_2||\breve{y}_1||\breve{y}_2||\breve{z}_1||\breve{z}_2 \in Z_q^{*6}$.
7. Compute $\alpha = H(u_1||u_2||e)$.
8. Output

$$m = \frac{e}{F\left(\hat{e} \cdot \hat{u}_1^{\breve{x}_1 + \hat{x}'_1 + \breve{y}_1 \alpha} \hat{u}_2^{\breve{x}_2 + \hat{x}'_2 + \breve{y}_2 \alpha} \cdot u_1^{\breve{z}_1} u_2^{\breve{z}_2} / K\right)} \tag{1}$$

Correctness. We only verify the correctness of Step 8 of case $\mathbf{C} = (\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$ in `Dec`. We can get the correctness of other steps from the correctness of Cramer-Shoup scheme.

$$\hat{e} \cdot \hat{u}_1^{\breve{x}_1+\hat{x}'_1+\breve{y}_1\alpha} \hat{u}_2^{\breve{x}_2+\hat{x}'_2+\breve{y}_2\alpha} \cdot u_1^{\breve{z}_1} u_2^{\breve{z}_2} / K$$

$$= \left(\left(\frac{u_1^{\hat{x}_1-\hat{x}'_1+\hat{y}_1\alpha} u_2^{\hat{x}_2-\hat{x}'_2+\hat{y}_2\alpha}}{v} \right)^{\hat{r}} \cdot u_1^{\hat{z}_1} u_2^{\hat{z}_2} \right) \cdot \left((u_1^{\breve{x}_1+\hat{x}'_1+\breve{y}_1\alpha} u_2^{\breve{x}_2+\hat{x}'_2+\breve{y}_2\alpha})^{\hat{r}} \cdot (K_1^{\breve{x}_1+\hat{x}'_1+\breve{y}_1\alpha} K_2^{\breve{x}_2+\hat{x}'_2+\breve{y}_2\alpha})^{\hat{r}'} \right) \cdot$$
$$u_1^{\breve{z}_1} u_2^{\breve{z}_2} / K$$

$$= \left(\frac{u_1^{\bar{H}(x_1)+\bar{H}(y_1)\alpha} u_2^{\bar{H}(x_2)+\bar{H}(y_2)\alpha}}{c^r d^{r\alpha}} \right)^{\hat{r}} \cdot u_1^{z_1} u_2^{z_2} \cdot \frac{\left((g_1^{k_1})^{\breve{x}_1+\hat{x}'_1+\breve{y}_1\alpha} (g_1^{k_2})^{\breve{x}_2+\hat{x}'_2+\breve{y}_2\alpha} \right)^{\hat{r}'}}{(A \cdot B^\alpha \cdot K_1^{\hat{x}'_1} K_2^{\hat{x}'_2})^{\hat{r}'}}$$

$$= \left(\frac{u_1^{\bar{H}(x_1)+\bar{H}(y_1)\alpha} u_2^{\bar{H}(x_2)+\bar{H}(y_2)\alpha}}{\left(g_1^{\bar{H}(x_1)} g_2^{\bar{H}(x_2)} \right)^r \left(g_1^{\bar{H}(y_1)} g_2^{\bar{H}(y_2)} \right)^{r\alpha}} \right)^{\hat{r}} \cdot u_1^{z_1} u_2^{z_2} \cdot \left(\frac{(g_1^{k_1})^{\breve{x}_1+\hat{x}'_1+\breve{y}_1\alpha} (g_1^{k_2})^{\breve{x}_2+\hat{x}'_2+\breve{y}_2\alpha}}{g_1^{k_1\breve{x}_1+k_2\breve{x}_2} \cdot (g_1^{k_1\breve{y}_1+k_2\breve{y}_2})^\alpha \cdot (g_1^{k_1})^{\hat{x}'_1} (g_1^{k_2})^{\hat{x}'_2}} \right)^{\hat{r}'}$$

$$= u_1^z = h^r$$

4.3 The Security Analysis of Scheme $\prod_{sm}$

Theorem 1. *Scheme $\prod_{sm}$ is CCA-O-secure in the standard model, if the DDH assumption holds in G, $\bar{H}, H$ are target collision-resistant, and $F, \breve{F}, \mathbf{F}$ are CSPRNGs. In particular*

$$\mathbf{Adv}^{\mathit{CCA\text{-}O}}_{\mathit{SUPRE}}(k) \le \frac{18 + 6q_d + 4q_1}{q} + 4\varepsilon_{ddh} + 5\varepsilon_H^{tcr} + 9\varepsilon_{\bar{H}}^{tcr} + 3\varepsilon_F^{\mathit{CSPRNG}} + \varepsilon_{\breve{F}}^{\mathit{CSPRNG}} + \varepsilon_{\mathbf{F}}^{\mathit{CSPRNG}},$$

where ε_{ddh}, ε_H^{tcr}, $\varepsilon_{\bar{H}}^{tcr}$, $\varepsilon_F^{\mathit{CSPRNG}}$, $\varepsilon_{\breve{F}}^{\mathit{CSPRNG}}$, and $\varepsilon_{\mathbf{F}}^{\mathit{CSPRNG}}$ are the probabilities of breaking the DDH assumption, the target collision resistance of hash functions, and the security of CSPRNG, respectively. q_1, q_2 are the numbers that the adversary queries the decryption oracle in Phase 1 and Phase 2, respectively. At last, $q_1 + q_2 = q_d$.

Proof. We prove the theorem by the similar method in [12,21].

Let $\mathcal{A}$ be an adversary who breaks scheme $\prod_{sm}$ in the sense of CCA-O security. The attack game is as described in Section 3.1. Suppose that the system parameter is $(q, g_1, G, F, \breve{F}, \mathbf{F}, \bar{H}, H)$, and we denote the values related to the challenge ciphertext as starred letters. For example, the target public key is $(g_2^*, \bar{g}_2^*, c^*, d^*, h^*, \bar{c}^*, \bar{d}^*)$, the corresponding secret key is $(x_1^*, x_2^*, y_1^*, y_2^*, z_1^*, z_2^*, z^*)$, and the challenge original ciphertext (u_1^*, u_2^*, e^*, v^*).

We say that an original ciphertext or a re-encrypted ciphertext is valid, if $u_1 = g_1^r$ and $u_2 = g_2^r$ for some r; otherwise, we say that it is invalid.

Let $\beta_1^* = \log_{g_1} g_2^*$ and $\log_{g_1} = \log$. Then

$$\log c^* = \bar{H}(x_1^*) + \beta_1^* \bar{H}(x_2^*) \tag{2}$$
$$\log d^* = \bar{H}(y_1^*) + \beta_1^* \bar{H}(y_2^*) \tag{3}$$

Note that we do not know the value of β_1^*, but know the value of β_2^*, where $\bar{g}_2^* = (g_2^*)^{\beta_2^*}$.

Game G_0: Let G_0 be the original attack game, E_0 be the event that $\mathbf{b} = \mathbf{b}'$ in G_0. Hence,

$$\mathbf{Adv}^{\texttt{CCA-O}}_{\texttt{SUPRE}}(k) = |\Pr[E_0] - 1/2| \tag{4}$$

We shall define a sequence $G_1, \cdots, G_I$ of modified attack games. For any $1 \le i \le I$, we let E_i be the event that $\mathbf{b} = \mathbf{b}'$ in G_i.

Game G_1: We modify decryption oracle in game G_0, so that it applies the following special output rule: If the adversary submits a re-encrypted ciphertext $(pk, \mathbf{C})$ with $\log_{g_1} \mathbf{u}_1 \neq \log_{\bar{g}_2} \mathbf{u}_2$, where $\bar{g}_2$ is associated to pk, then it outputs `reject`. $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$ is a re-encrypted ciphertext; hence, from Theorem 2 we have that

$$|\Pr[E_1] - \Pr[E_0]| \le \frac{4 + q_d + q_1}{q} + \varepsilon_{ddh} + \varepsilon^{tcr}_{H} + \varepsilon^{tcr}_{\bar{H}} + \varepsilon^{\texttt{CSPRNG}}_{\mathbf{F}} \tag{5}$$

Game G_2: We modify decryption oracle in game G_1, so that it applies the following special output rule: If the adversary submits a re-encrypted ciphertext $(pk, \mathbf{C})$ with $\log_{g_1} \breve{u}_1 \neq \log_{\bar{g}_2} \breve{u}_2$, where $\bar{g}_2$ is associated to pk, then it outputs `reject`. $(\breve{u}_1, \breve{u}_2, \breve{e}, \breve{v})$ can be considered as a re-encrypted ciphertext; hence, from Theorem 2 we have that

$$|\Pr[E_2] - \Pr[E_1]| \le \frac{4 + q_d + q_1}{q} + \varepsilon_{ddh} + \varepsilon^{tcr}_{H} + \varepsilon^{tcr}_{\bar{H}} + \varepsilon^{\texttt{CSPRNG}}_{\breve{F}} \tag{6}$$

Game G_3: We modify Challenge phase in game G_2, so that $v^* = (c^*)^{r^*}(d^*)^{r^*\alpha^*}$ and $e^* = F(h^{*r^*}) \cdot m_{\mathbf{b}}$ are replaced by $v^* = (u_1^*)^{\bar{H}(x_1^*)+\bar{H}(y_1^*)\alpha^*}(u_2^*)^{\bar{H}(x_2^*)+\bar{H}(y_2^*)\alpha^*}$ and $e^* = F((u_1^*)^{z_1^*}(u_2^*)^{z_2^*}) \cdot m_{\mathbf{b}}$, respectively. This change is purely conceptual, hence

$$\Pr[E_3] = \Pr[E_2] \tag{7}$$

Game G_4: We modify Challenge phase in game G_3, so that (u_1^*, u_2^*) is replaced by a random pair $(g_1^{r_1^*}, (g_2^*)^{r_2^*})$, where $r_1^* \neq r_2^*$. Under the DDH assumption, $\mathcal{A}$ will hardly notice this change. We have the same proof as that of Lemma 6.3 in [12] for the following.

$$|\Pr[E_4] - \Pr[E_3]| \le \varepsilon_{ddh} + 3/q \tag{8}$$

Game G_5: We modify the decryption oracle in game G_4, so that it applies the following special rejection rule: In Phase 2, if the adversary submits an original ciphertext (pk^*, C) with $(u_1, u_2, e) \neq (u_1^*, u_2^*, e^*)$ but $\alpha = \alpha^*$, then the decryption oracle immediately outputs `reject`. It is easy to see that game G_4 and game G_5 proceed identically until that the decryption oracle in G_5 outputs `reject` by using the rule while the decryption oracle in G_4 does not. We have the same proof as that of Lemma 6.5 in [12] for the following.

$$|\Pr[E_5] - \Pr[E_4]| \le \varepsilon^{tcr}_{H} + 1/q \tag{9}$$

Game G_6: We modify the re-encryption oracle in game G_5, so that it applies the following special output rule: In Phase 2, if the adversary submits an original ciphertext (pk^*, C) with $(u_1, u_2, e) \neq (u_1^*, u_2^*, e^*)$ but $\alpha = \alpha^*$, then $\hat{e}$ is replaced by a random number T from G. Similar with game G_5, we have

$$|\Pr[E_6] - \Pr[E_5]| \leq \varepsilon_H^{tcr} + 1/q \tag{10}$$

Game G_7: We modify the decryption oracle in game G_6, so that it applies the following special output rule: In Phase 2, if the adversary submits a re-encrypted ciphertext $(pk, \mathbf{C})$ with $(u_1, u_2, e) \neq (u_1^*, u_2^*, e^*)$ but $\alpha = \alpha^*$, and the corresponding delegator is pk^*, then $\hat{e}$ is replaced by a random number T from G. Similar with game G_5, we have

$$|\Pr[E_7] - \Pr[E_6]| \leq \varepsilon_H^{tcr} + 1/q \tag{11}$$

Game G_8: We modify the decryption oracle in game G_7, so that it rejects all invalid original ciphertext (pk^*, C)'s in Phase 1. Let R_8 be the event that (pk^*, C) is rejected in G_8 that would not have been rejected in G_7. It is clear that games G_8 and G_7 proceed identically until the event R_8 occurs. Hence, we have

$$|\Pr[E_8] - \Pr[E_7]| \leq \Pr[R_8] \tag{12}$$

Lemma 1.

$$\Pr[R_8] \leq \varepsilon_{\bar{H}}^{tcr} + q_1/q \tag{13}$$

Proof of Lemma 1: From $\mathcal{A}$'s view, $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ is a random point satisfying eqs.(2) and (3), since $(x_1^*, x_2^*, y_1^*, y_2^*)$ are chosen randomly and independently, and $\bar{H}$ is target collision-resistant. Note that $\mathcal{A}$ cannot get $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ by colluding with the delegatee due to the restrictions in the CCA-O game. Suppose $\mathcal{A}$ queries an invalid original ciphertext (u_1, u_2, e, v) to the decryption oracle, where $\log u_1 = r_1$ and $\log u_2 = \beta_1^* r_2$ with $r_1 \neq r_2$. The challenger won't reject (u_1, u_2, e, v), unless the following equation holds

$$\log v = r_1 \bar{H}(x_1^*) + \beta_1^* r_2 \bar{H}(x_2^*) + \alpha r_1 \bar{H}(y_1^*) + \alpha r_2 \beta_1^* \bar{H}(y_2^*) \tag{14}$$

where $\alpha = H(u_1, u_2, e)$. However, it is clear that eqs.(2), (3) and (14) are linearly independent. Hence, we get the lemma.

Game G_9: We modify the re-encryption oracle in game G_8, so that it would output a random $\hat{e}$ from G for all invalid original ciphertext (pk^*, C)'s in Phase 1. Similar with Game G_8, we have

$$|\Pr[E_9] - \Pr[E_8]| \leq \varepsilon_{\bar{H}}^{tcr} + q_1/q \tag{15}$$

Game G_{10}: We modify the decryption oracle in Phase 1 in game G_9, so that it would output a random m from G for all invalid re-encrypted ciphertext $(pk, \mathbf{C})$'s with that the corresponding delegator is pk^*. Similar with Game G_8, we have

$$|\Pr[E_{10}] - \Pr[E_9]| \leq \varepsilon_{\bar{H}}^{tcr} + q_1/q \tag{16}$$

Game G_{11}: We modify Challenge phase in game G_{10}, so that $(u_1^*, u_2^*) = (g_1^{r_1^*}, (g_2^*)^{r_2^*})$ is randomly chosen in such a way that an event R_{11} does not occur, where R_{11} is the event that $(u_1^*, u_2^*) = (u_1, u_2)$ for some invalid ciphertext which $\mathcal{A}$ queries in Phase 1. It is clear that R_{11} happens with q_1/q at most since r_1^*, r_2^* are chosen independently and randomly. Hence, we have

$$|\Pr[E_{11}] - \Pr[E_{10}]| \leq q_1/q \tag{17}$$

Game G_{12}: We modify the decryption oracle in game G_{11}, so that it rejects all invalid original ciphertext (pk^*, C)'s in Phase 2. Let R_{12} be the event that an original ciphertext is rejected in G_{12} that would not have been rejected under the rules of G_{11}. It is clear that games G_{12} and G_{11} proceed identically until the event R_{12} occurs. Hence, we have

$$|\Pr[E_{12}] - \Pr[E_{11}]| \leq \Pr[R_{12}] \tag{18}$$

Lemma 2.

$$\Pr[R_{12}] \leq \varepsilon_{\bar{H}}^{tcr} + q_2/q \tag{19}$$

Proof of Lemma 2: From $\mathcal{A}$'s view, $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ is a random point satisfying eqs.(2), (3) and (20), since $(x_1^*, x_2^*, y_1^*, y_2^*)$ are chosen randomly and independently, and $\bar{H}$ is target collision-resistant.

$$\log v^* = r_1^* \bar{H}(x_1^*) + \beta_1^* r_2^* \bar{H}(x_2^*) + \alpha^* r_1^* \bar{H}(y_1^*) + \alpha^* r_2^* \beta_1^* \bar{H}(y_2^*) \tag{20}$$

Note that $\mathcal{A}$ cannot get $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ by colluding with the delegatee due to the restrictions in the CCA-O game. Suppose $\mathcal{A}$ queries an invalid ciphertext (u_1, u_2, e, v) to the decryption oracle, where $\log u_1 = r_1$ and $\log u_2 = \beta_1^* r_2$ with $r_1 \neq r_2$. The challenger won't reject (u_1, u_2, e, v), unless eq. (14) holds. However, it is clear that eqs.(2), (3), (14) and (20) are linearly independent. Hence, we get the lemma.

Game G_{13}: We modify the re-encryption oracle in game G_{12}, so that it will output a random $\hat{e}$ from G for all invalid original ciphertext (pk^*, C)'s in Phase 2. We have two cases of analysis for the modification.

- If $C = C^*$, then the output ciphertext cannot be queried to $\mathcal{O}_{dec}$. Furthermore, from Theorem 2, the adversary cannot modify the re-encrypted ciphertext.
- If $C \neq C^*$, we first analyze that the adversary cannot get $R^{\hat{r}^*}$ from $(\hat{u}_1^*, \hat{u}_1^*, K^*)$, where R is the value that could be computed by the adversary without using (x_1', x_2'). The result can be easily obtained, since $(\hat{u}_1^*, \hat{u}_1^*)$ are computed without using (x_1', x_2'), while K^* is computed with using (x_1', x_2').

  Now, we have the similar analysis of the proof of Lemma 2 that eq. (14) holds for a negligible probability. Hence, the output of decryption query with the re-encrypted ciphertext is a random number with a overwhelming probability.

  As a result, we have

$$|\Pr[E_{13}] - \Pr[E_{11}]| \leq \frac{4 + 2q_d}{q} + \varepsilon_{ddh} + \varepsilon_H^{tcr} + 2\varepsilon_{\bar{H}}^{tcr} + \varepsilon_{\mathbf{F}}^{\mathtt{CSPRNG}} \tag{21}$$

Game G_{14}: We modify the decryption oracle in Phase 2 in game G_{13}, so that it will output a random m from G for all invalid re-encrypted ciphertext $(pk, \mathbf{C})$'s with that the corresponding delegator is pk^*. We have two cases of analysis for the modification.

– If (u_1, u_2, e) associated to $\mathbf{C}$ is equal to (u_1^*, u_2^*, e^*), then the input of $F(\cdot)$ in eq. (1) is $(u_1^*)^{f_1(\breve{z}_1)}(u_2^*)^{f_2(\breve{z}_2)}$, where $f_1(\cdot)$ and $f_2(\cdot)$ are two polynomial functions known by the adversary, and $(\breve{z}_1, \breve{z}_2)$ are computed from $(\breve{u}_1, \breve{u}_2, \breve{e}, \breve{v})$ and unknown to the adversary. Note that we assume that $(\breve{z}_1, \breve{z}_2)$ are valid; otherwise, this decryption query is helpless for the adversary. Since the adversary has no idea about $(\breve{z}_1, \breve{z}_2)$ and $F(\cdot)$ is a CSPRNG, the output m is a random number from the view of the adversary.
– If (u_1, u_2, e) associated to $\mathbf{C}$ is not equal to (u_1^*, u_2^*, e^*), then we have the similar analysis of the proof of Lemma 2 that eq. (14) holds for a negligible probability. Hence, the output of the decryption query with this kind of re-encrypted ciphertext are random numbers with a overwhelming probability.

As a result, we have

$$|\Pr[E_{14}] - \Pr[E_{13}]| \leq \varepsilon_{\bar{H}}^{tcr} + q_2/q + \varepsilon_F^{\texttt{CSPRNG}} \tag{22}$$

Game G_{15}: We modify Challenge phase in G_{14}, so that we use a random value from G instead of $F((u_1^*)^{z_1^*}(u_2^*)^{z_2^*})$. Since u_1^* and u_2^* are computed randomly and independently, and F is a CSPRNG, we have

$$|\Pr[E_{15}] - \Pr[E_{14}]| \leq \varepsilon_F^{\texttt{CSPRNG}} \tag{23}$$

Furthermore, it is easy to see that e^* is computed by one-time pad, and the re-encryption oracle and decryption oracle output random values when the input is invalid. Hence, we have

$$\Pr[E_{15}] = 1/2 \tag{24}$$

Combining eqs.(4)— (13), (15)— (19) and (21)— (24), we have the theorem. □

Theorem 2. *Scheme $\prod_{sm}$ is CCA-R-secure in the standard model, if the DDH assumption holds in G, $\bar{H}, H$ are target collision-resistant, and $\mathbf{F}$ is a CSPRNG. In particular*

$$\mathbf{Adv}_{\textit{SUPRE}}^{\textit{CCA-R}}(k) \leq \frac{4 + q_d + q_1}{q} + \varepsilon_{ddh} + \varepsilon_H^{tcr} + \varepsilon_{\bar{H}}^{tcr} + \varepsilon_{\mathbf{F}}^{\textit{CSPRNG}},$$

where the meanings of the notations are the same as that in Theorem 1.

Proof. We prove the theorem by the similar method in [12,21].

Let $\mathcal{A}$ be an adversary who breaks scheme $\prod_{sm}$ in the sense of CCA-R security. The attack game is as described in Section 3.1. Suppose that the system parameter is $(q, g_1, G, F, \breve{F}, \mathbf{F}, \bar{H}, H)$, and we denote the values related to the challenge ciphertext as starred letters. For example, the target public key is $(g_2^*, \bar{g}_2^*, c^*, d^*, h^*, \bar{c}^*, \bar{d}^*)$, the corresponding secret key is $(x_1^*, x_2^*, y_1^*, y_2^*, z_1^*, z_2^*, z^*)$, and the challenge re-encrypted ciphertext $(\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{e}^*, \mathbf{v}^*)$.

We say that a re-encrypted ciphertext is valid, if $\mathbf{u}_1 = g_1^{\mathbf{r}}$ and $\mathbf{u}_2 = \bar{g}_2^{\mathbf{r}}$ for some $\mathbf{r}$; otherwise, we say that it is invalid.

Let $\beta^* = \beta_1^* \beta_2^* = \log_{g_1} \bar{g}_2^*$ and $\log_{g_1} = \log$. Then

$$\log \bar{c}^* = x_1^* + \beta^* x_2^* \tag{25}$$
$$\log \bar{d}^* = y_1^* + \beta^* y_2^* \tag{26}$$

Note that we do not know the value of β^*, but know the value of β_2^*, where $g_2^* = (\bar{g}_2^*)^{1/\beta_2^*}$.

Game G_0: Let G_0 be the original attack game, E_0 be the event that $\mathbf{b} = \mathbf{b}'$ in G_0. Hence,

$$\mathbf{Adv}_{\mathtt{SUPRE}}^{\mathtt{CCA\text{-}R}}(k) = |\Pr[E_0] - 1/2| \tag{27}$$

We shall define a sequence $G_1, \cdots, G_I$ of modified attack games. For any $1 \leq i \leq I$, we let E_i be the event that $\mathbf{b} = \mathbf{b}'$ in G_i.

Game G_1: We modify Challenge phase in game G_1, so that $\mathbf{v}^* = (\bar{c}^*)^{\mathbf{r}^*} (\bar{d}^*)^{\mathbf{r}^* \hat{\alpha}^*}$ and $\mathbf{e}^* = \mathbf{F}(h^{*\mathbf{r}^*}) \oplus \nabla$ are replaced by $\mathbf{v}^* = (\mathbf{u}_1^*)^{x_1^* + y_1^* \hat{\alpha}^*} (\mathbf{u}_2^*)^{x_2^* + y_2^* \hat{\alpha}^*}$ and $\mathbf{e}^* = \mathbf{F}((\mathbf{u}_1^*)^{z_1^*} (\mathbf{u}_2^*)^{z_2^*/\beta_2^*}) \oplus \nabla$, respectively. This change is purely conceptual, hence

$$\Pr[E_1] = \Pr[E_0] \tag{28}$$

Game G_2: We modify Challenge phase in game G_1, so that $(\mathbf{u}_1^*, \mathbf{u}_2^*)$ is replaced by a random pair $(g_1^{\mathbf{r}_1^*}, (\bar{g}_2^*)^{\mathbf{r}_2^*})$, where $\mathbf{r}_1^* \neq \mathbf{r}_2^*$. Due to the target collision resistance of $\bar{H}$, knowing $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ does not hurt the secrecy of $(x_1^*, x_2^*, y_1^*, y_2^*)$. Hence, under the DDH assumption, $\mathcal{A}$ will hardly notice this change. We have the same proof as that of Lemma 6.3 in [12] for the following.

$$|\Pr[E_2] - \Pr[E_1]| \leq \varepsilon_{ddh} + 3/q + \varepsilon_{\bar{H}}^{tcr} \tag{29}$$

Game G_3: We modify the decryption oracle in game G_2, so that it applies the following special rejection rule: In Phase 2, if the adversary submits a re-encrypted ciphertext $(pk^*, \mathbf{C})$ with $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}) \neq (\mathbf{u}_1^*, \mathbf{u}_2^*, \mathbf{e}^*)$ but $\hat{\alpha} = \hat{\alpha}^*$, then the decryption oracle immediately outputs `reject`. It is easy to see that game G_2 and game G_3 proceed identically until that the decryption oracle in G_3 outputs `reject` by using the rule while the decryption oracle in G_2 does not.

We have the same proof as that of Lemma 6.5 in [12] for the following.

$$|\Pr[E_3] - \Pr[E_2]| \leq \varepsilon_{H}^{tcr} + 1/q \tag{30}$$

Game G_4: We modify the decryption oracle in game G_3, so that it rejects all invalid re-encrypted ciphertext $(pk^*, \mathbf{C})$'s in Phase 1. Let R_4 be the event that $(pk^*, \mathbf{C})$ is rejected in G_4 that would not have been rejected in G_3. It is clear that games G_4 and G_3 proceed identically until the event R_4 occurs. Hence, we have

$$|\Pr[E_4] - \Pr[E_3]| \leq \Pr[R_4] \tag{31}$$

Lemma 3.

$$\Pr[R_4] \leq \varepsilon_{H}^{tcr} + q_1/q \tag{32}$$

Proof of Lemma 3: $\mathcal{A}$ cannot get $(x_1^*, x_2^*, y_1^*, y_2^*)$ from $(\bar{H}(x_1^*), \bar{H}(x_2^*), \bar{H}(y_1^*), \bar{H}(y_2^*))$ due to the target collision resistance of $\bar{H}$. Hence, from $\mathcal{A}$'s view, $(x_1^*, x_2^*, y_1^*, y_2^*)$ is a random point satisfying eqs.(25) and (26). Suppose $\mathcal{A}$ queries an invalid ciphertext (u_1, u_2, e, v) to the decryption oracle, where $\log \mathbf{u}_1 = \mathbf{r}_1$ and $\log \mathbf{u}_2 = \beta^* \mathbf{r}_2$ with $\mathbf{r}_1 \neq \mathbf{r}_2$. The challenger won't reject $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$, unless the following equation holds

$$\log \mathbf{v} = \mathbf{r}_1 x_1^* + \beta^* \mathbf{r}_2 x_2^* + \hat{\alpha} \mathbf{r}_1 y_1^* + \hat{\alpha} \mathbf{r}_2 \beta^* y_2^* \tag{33}$$

where $\hat{\alpha} = H(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e})$. However, it is clear that eqs.(25), (26) and (33) are linearly independent. Hence, we get the lemma.

Game G_5: We modify Challenge phase in game G_4, so that $(\mathbf{u}_1^*, \mathbf{u}_2^*) = (g_1^{\mathbf{r}_1^*}, (\bar{g}_2^*)^{\mathbf{r}_2^*})$ is randomly chosen in such a way that an event R_5 does not occur, where R_5 is the event that $(\mathbf{u}_1^*, \mathbf{u}_2^*) = (\mathbf{u}_1, \mathbf{u}_2)$ for some invalid ciphertext which $\mathcal{A}$ queries in Phase 1. It is clear that R_5 happens with q_1/q at most since $\mathbf{r}_1^*, \mathbf{r}_2^*$ are chosen independently and randomly. Hence, we have

$$|\Pr[E_5] - \Pr[E_4]| \leq q_1/q \tag{34}$$

Game G_6: We modify the decryption oracle in game G_5, so that it rejects all invalid re-encrypted ciphertext $(pk^*, \mathbf{C})$'s in Phase 2. Let R_6 be the event that an original ciphertext is rejected in G_6 that would not have been rejected under the rules of G_5. It is clear that games G_6 and G_5 proceed identically until the event R_6 occurs. Hence, we have

$$|\Pr[E_6] - \Pr[E_5]| \leq \Pr[R_6] \tag{35}$$

Lemma 4.

$$\Pr[R_6] \leq \varepsilon_{\bar{H}}^{tcr} + q_2/q \tag{36}$$

Proof of Lemma 2: From $\mathcal{A}$'s view, due to the target collision resistance of $\bar{H}$, $(x_1^*, x_2^*, y_1^*, y_2^*)$ is a random point satisfying eqs.(25), (26) and (37).

$$\log \mathbf{v}^* = \mathbf{r}_1^* x_1^* + \beta^* \mathbf{r}_2^* x_2^* + \hat{\alpha}^* \mathbf{r}_1^* y_1^* + \hat{\alpha}^* \mathbf{r}_2^* \beta^* y_2^* \tag{37}$$

Suppose $\mathcal{A}$ queries an invalid re-encrypted ciphertext $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$ to the decryption oracle, where $\log \mathbf{u}_1 = \mathbf{r}_1$ and $\log \mathbf{u}_2 = \beta^* \mathbf{r}_2$ with $\mathbf{r}_1 \neq \mathbf{r}_2$. The challenger won't reject $(\mathbf{u}_1, \mathbf{u}_2, \mathbf{e}, \mathbf{v})$, unless eq. (33) holds. However, it is clear that eqs.(25), (26), (33) and (37) are linearly independent. Hence, we get the lemma.

Game G_7: We modify Challenge phase in G_6, so that we use a random value from G instead of $\mathbf{F}((\mathbf{u}_1^*)^{z_1^*}(\mathbf{u}_2^*)^{z_2^*/\beta_2^*})$. Since $\mathbf{u}_1^*$ and $\mathbf{u}_2^*$ are computed randomly and independently, and F is a CSPRNG, we have

$$|\Pr[E_7] - \Pr[E_6]| \leq \varepsilon_F^{\mathtt{CSPRNG}} \tag{38}$$

Furthermore, it is easy to see that e^* is computed by one-time pad, and decryption oracle outputs `reject` when the input is invalid. Hence, we have

$$\Pr[E_7] = 1/2 \tag{39}$$

Combining eqs.(27)— (32), (34)— (36) and (38)— (39), we have the theorem. $\square$

5 Comparison

In this section, we compare our proposal with scheme WCY$^+$10 [29] in terms of the computational cost and ciphertext size. In Table 2, we denote t_p, t_{me}, and t_{re} as the timings of a bilinear pairing, a multi(=sequential)-exponentiation, and a regular exponentiation, respectively. We omit other timings. We also denote $\ell_{\mathbb{G}_1}$, $\ell_{\mathbb{G}_2}$, ℓ_G, ℓ_q and ℓ as the bit lengths of elements in bilinear groups $\mathbb{G}_1$, $\mathbb{G}_2$ with prime order q, elements in a finite cyclic group G with prime order q, elements in Z_q^*, and the security parameter, respectively.

Following the relations in [20], we have that bilinear pairings$\approx 3-5$, multi(=sequential)-exponentiation≈ 1.2, and regular exponentiation$= 1$. From Table 2, we can see that scheme WCY$^+$10 needs less storage for the re-encrypted ciphertext and less time for algorithm `ReKeyGen` than our proposal, while our proposal are more efficient in algorithms `Enc`, `ReEnc`, `Dec`, which are used more frequently than `ReKeyGen`.

Table 2. Comparison between our proposal and scheme WCY$^+$10.

			WCY$^+$10 [29]	Ours
Comput. Cost	`ReKeyGen`		$1t_{re}$	$1t_{me}+7t_{re}$
	`Enc`		$2t_{me}+3t_{re}$	$1t_{me}+3t_{re}$
	`ReEnc`		$3t_p+3t_{me}+1t_{re}$	$8t_{me}+4t_{re}$
	`Dec`	Original	$3t_p+3t_{me}+2t_{re}$	$1t_{me}+1t_{re}$
		Re-encrypted	$2t_p+1t_{me}+1t_{re}$	$4t_{me}+2t_{re}$
Ciphertext Size	Original		$1\ell_q+3\ell_{\mathbb{G}_1}+\ell$	$4\ell_G$
	Re-encrypted		$1\ell_q+2\ell_{\mathbb{G}_1}+1\ell_{\mathbb{G}_2}+\ell$	$12\ell_G+6\ell_q$

6 Conclusion

In this paper, we have proposed a novel proxy re-encryption scheme. To the best of our knowledge, our proposal is the first unidirectional PRE scheme without pairings while proven-secure in the standard model. A special property of our proposal is without public verifiability, which is the key part in all the existing CCA-secure PRE schemes.

References

1. `http://tdt.sjtu.edu.cn/~jshao/prcbib.htm`.

2. G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage. In *Internet Society (ISOC): NDSS 2005*, pages 29–43, 2005.
3. G. Ateniese, K. Fu, M. Green, and S. Hohenberger. Improved Proxy Re-encryption Schemes with Applications to Secure Distributed Storage. *ACM Transactions on Information and System Security (TISSEC)*, 9(1):1–30, 2006.
4. M. Blaze, G. Bleumer, and M. Strauss. Divertible protocols and atomic proxy cryptography. In *EUROCRYPT 1998*, volume 1403 of *LNCS*, pages 127–144, 1998.
5. R. Canetti and S. Goldwasser. An efficient threshold public key cryptosystem secure against adaptive chosen ciphertext attack. In *EUROCRYPT 1999*, volume 1592 of *LNCS*, pages 90–106, 1999.
6. R. Canetti, S. Halevi, and J. Katz. Chosen-ciphertext security from identity-based encryption. In *EUROCRYPT 2004*, volume 3027 of *LNCS*, pages 207–222, 2004.
7. R. Canetti and S. Hohenberger. Chosen-Ciphertext Secure Proxy Re-Encryption. In *ACM CCS 2007*, 2007. Full version: Cryptology ePrint Archieve: Report 2007/171.
8. Y-P. Chiu, C-L. Lei, and C-Y. Huang. Secure multicast using proxy encryption. In *ICICS 2005*, volume 3783 of *LNCS*, pages 280–290, 2005.
9. S.S.M. Chow, J. Weng, Y. Yang, and R.H. Deng. Efficient Unidirectional Proxy Re-Encryption. In *Africacrypt 2010*, volume 6055 of *LNCS*, pages 316–332, 2010. Full version: `http://eprint.iacr.org/2009/189`.
10. C. Chu, S. S. M. Chow J. Weng, J. Zhou, and R. H. Deng. Conditional Proxy Broadcast Re-Encryption . In *ACISP 2009*, volume 5594 of *LNCS*, pages 327–342, 2009.
11. R. Cramer and V. Shoup. A practical public key cryptosystem provably secure against adaptive chosen ciphertext attack. In *CRYPTO 1998*, volume 1462 of *LNCS*, pages 13–25, 1998.
12. R. Cramer and V. Shoup. Design and analysis of practical public-key encryption schemes secure against adaptive chosen ciphertext attack. *SIAM Journal on Computing*, 33(1):167–226, 2003.
13. T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.
14. C. Gentry. Fully homomorphic encryption using ideal lattices. In *ACM STOC 2009*, pages 169–178, 2009.
15. M. Green and G. Ateniese. Identity-Based Proxy Re-encryption. In *ACNS 2007*, volume 4521 of *LNCS*, pages 288–306, 2007. Full version: Cryptology ePrint Archieve: Report 2006/473.
16. T.S. Heydt-Benjamin, H. Chae, B. Defend, and K. Fu. Privacy for public transportation. In *PET 2006*, volume 4258 of *LNCS*, pages 1–19, 2005.
17. H. Khurana and H-S. Hahm. Certified mailing lists. In *ASIACCS 2006*, pages 46–58, 2006.
18. H. Khurana and R. Koleva. Scalable security and accounting services for content-based publish subscribe systems. *International Journal of E-Business Research*, 2(3), 2006.
19. H. Khurana, A. Slagell, and R. Bonilla. Sels: A secure e-mail list service. In *ACM SAC 2005*, pages 306–313, 2005.
20. E. Kiltz. Chosen-ciphertext secure key-encapsulation based on gap hashed diffie-hellman. In *PKC 2007*, volume 4450 of *LNCS*, pages 282–297, 2007.
21. K. Kurosawa and Y. Desmedt. A new paradigm of hybrid encryption scheme. In *CRYPTO 2004*, volume 3152 of *LNCS*, pages 426–442, 2004.
22. B. Libert and D. Vergnaud. Unidirectional Chosen-Ciphertext Secure Proxy Re-Encryption. In *PKC 2008*, volume 4939 of *LNCS*, pages 360–379, 2008.
23. T. Matsuda, R. Nishimaki, and K. Tanaka. CCA Proxy Re-Encryption without Bilinear Maps in the Standard Model. In *PKC 2010*, volume 6056 of *LNCS*, pages 261–278, 2010.
24. J. Shao and Z. Cao. CCA-Secure Proxy Re-Encryption without Pairings. In *PKC 2009*, volume 5443 of *LNCS*, pages 357–376, 2009. Full version: `http://eprint.iacr.org/2009/164`.
25. J. Shao, Z. Cao, and P. Liu. CCA-Secure PRE Scheme without Random Oracles. `http://eprint.iacr.org/2010/112`.
26. J. Shao, Z. Cao, and P. Liu. SCCR: a generic approach to simultaneously achieve CCA security and collusion-resistance in proxy re-encryption. *SECURITY AND COMMUNICATION NETWORKS*, 2009.
27. G. Taban, A.A. Cárdenas, and V.D. Gligor. Towards a secure and interoperable drm architecture. In *ACM DRM 2006*, pages 69–78, 2006.
28. A. Talmy and O. Dobzinski. Abuse freedom in access control schemes. In *AINA 2006*, pages 77–86, 2006.
29. J. Weng, M. Chen, Y. Yang, R.H. Deng, K. Chen, and Feng Bao. CCA-Secure Unidirectional Proxy Re-Encryption in the Adaptive Corruption Model without Random Oracles. *Science China Information Sciences*, 53(3):593–606, 2010. Updated version: `http://eprint.iacr.org/2010/265`.
30. J. Weng and Y. Zhao. On the Security of a Bidirectional Proxy Re-Encryption Scheme from PKC 2010. `http://eprint.iacr.org/2010/319`.