A Provably-Secure Unidirectional Proxy Re-encryption Scheme Without Pairing in the Random Oracle Model

S. Sharmila Deva Selvi, Arinjita Paul(✉), and Chandrasekaran Pandurangan

Theoretical Computer Science Lab,
Department of Computer Science and Engineering,
Indian Institute of Technology Madras, Chennai, India
{sharmila,arinjita,prangan}@cse.iitm.ac.in

Abstract. Proxy re-encryption (PRE) enables delegation of decryption rights by entrusting a proxy server with special information, that allows it to transform a ciphertext under one public key into a ciphertext of the same message under a different public key, without learning anything about the underlying plaintext. In Africacrypt 2010, the first PKI-based collusion resistant CCA secure PRE scheme without pairing was proposed in the random oracle model. In this paper, we point out an important weakness in the security proof of the scheme. We also present a collusion-resistant pairing-free unidirectional PRE scheme which meets CCA security under a variant of the computational Diffie-Hellman hardness assumption in the random oracle model.

Keywords: Proxy re-encryption · Random oracle model
Chosen ciphertext security · Provably secure · Unidirectional

1 Introduction

Proxy re-encryption is an important cryptographic primitive that allows a third party termed as *proxy server*, to transform the ciphertext of a user into a ciphertext of another user without learning anything about the underlying message. As pointed out by Mambo and Okamoto in [7], this is a common situation in practice where a data encrypted under PK_{Alice} is required to be encrypted under PK_{Bob}, such as applications like encrypted email forwarding, distributed file systems and outsourced filtering of encrypted spam. Here, Alice provides a secret information to the proxy called Re-Encryption Key (but not her private key

S. Sharmila Deva Selvi—Postdoctoral researcher supported by Project No. CCE/CEP/22/VK&CP/CSE/14-15 on Information Security & Awareness(ISEA) Phase-II by Ministry of Electronics & Information Technology, Government of India.
A. Paul and C. Pandurangan—Work partially supported by Project No. CCE/CEP/22/VK&CP/CSE/14-15 on ISEA-Phase II.

S. Capkun and S. S. M. Chow (Eds.): CANS 2017, LNCS 11261, pp. 459–469, 2018.
https://doi.org/10.1007/978-3-030-02641-7_21

SK_{Alice}) allowing it to transform $E_{PK_{Alice}}(m)$ to $E_{PK_{Bob}}(m)$ without learning anything about m or SK_{Alice}.

Most of the proxy re-encryption schemes in the literature are based on costly bilinear pairing operation [1,3,6]. Despite recent advances in implementation techniques, bilinear pairing takes more than twice the time taken by modular exponentiation computation and is an expensive operation. As stated by Chow *et al.* [4], removing pairing operations from PRE constructions is one of the open problems left by [3]. Weng *et al.* [5] proposed the first CCA secure pairing-free PRE scheme, which was however shown to be vulnerable to *collusion attack* [10]. Collusion resistance, also termed as *delegator secret security* is a desirable property in many practical scenarios such as secure cloud services, which prevents a colluding proxy and malicious delegatees from recovering the private key of the delegator. In 2010, Chow *et al.* [4] proposed the first construction of a collusion-resistant CCA secure pairing-free PRE scheme. However, in our work, we point out a major weakness in the security proof of the scheme by Chow *et al.* We also provide a construction of a CCA-secure collusion-resistant pairing-free unidirectional single-hop proxy re-encryption scheme under the Computational Diffie-Hellman (CDH) and the Divisible Computational Diffie-Hellman (DCDH) hardness assumptions in the random oracle model. Prior to our work, Canard *et al.* [2] exposed a similar flaw in the security proof of the scheme due to Chow *et al.* [4], and provided a fix to the scheme using NIZKOE (Non-interactive Zero-Knowledge proofs with Online Extractors). We show that our scheme is more efficient than the modified scheme due to Canard *et al.*, providing an efficient pairing-free unidirectional collusion-resistant PRE scheme.

Complexity Assumptions

We define the complexity assumptions used in the security proof of our scheme. Let $\mathbb{G}$ be a cyclic multiplicative group of prime order q.

Definition 1. ***Computational Diffie Hellman Assumption (CDH):*** *The CDH problem in $\mathbb{G}$ is, given $(g, g^a, g^b) \in \mathbb{G}^3$, compute g^{ab}, where $a, b \leftarrow \mathbb{Z}_q^*$.*

Definition 2. ***Divisible Computational Diffie Hellman Assumption (DCDH):*** *The DCDH problem in $\mathbb{G}$ is, given $(g, g^a, g^b) \in \mathbb{G}^3$, compute $g^{b/a}$, where $a, b \leftarrow \mathbb{Z}_q^*$.*

Definition 3. ***Discrete Logarithm Assumption (DL):*** *The DL problem in $\mathbb{G}$ is, given $(g, g^a) \in \mathbb{G}^2$, compute a, where $a \leftarrow \mathbb{Z}_q^*$.*

2 Analysis of a PRE Scheme by Chow *et al.* [4]

We review the scheme due to Chow *et al.* [4] and point out the weakness in the security proof of the scheme in this section.

2.1 Review of the Scheme

- **Setup**(λ)**:** Choose two primes p and q such that $q|p-1$ and the security parameter λ defines the bit-length of q. Let $\mathbb{G}$ be a subgroup of $\mathbb{Z}_q^*$ with order q and let g be a generator of the group $\mathbb{G}$. Choose four hash functions: $H_1 : \{0,1\}^{l_0} \times \{0,1\}^{l_1} \rightarrow \mathbb{Z}_q^*, H_2 : \mathbb{G} \rightarrow \{0,1\}^{l_0+l_1}, H_3 : \{0,1\}^* \rightarrow \mathbb{Z}_q^*, H_4 : \mathbb{G} \rightarrow \mathbb{Z}_q^*$. The hash functions H_1, H_2, H_3 are modelled as random oracles in the security proof reduction. Here l_0 and l_1 are security parameters determined by λ, and the message space $\mathcal{M}$ is $\{0,1\}^{l_0}$. Return the public parameters $PARAM = (q, \mathbb{G}, g, H_1, H_2, H_3, H_4, l_0, l_1)$.
- **KeyGen**$(U_i, PARAMS)$**:** To generate the private key (SK_i) and the corresponding public key (PK_i) of user U_i:
  - Pick $x_{i,1}, x_{i,2} \in_R \mathbb{Z}_q^*$ and set $SK_i = (x_{i,1}, x_{i,2})$.
  - Compute $PK_i = (PK_{i,1}, PK_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.
- **ReKeyGen**$(SK_i, PK_i, PK_j, PARAMS)$**:** On input of the private key SK_i and public key PK_i of user U_i and user j's public key PK_j, generate the re-encryption key $RK_{i \rightarrow j}$ as shown:
  - Pick $h \in_R \{0,1\}^{l_0}$, $\pi \in_R \{0,1\}^{l_1}$.
  - Compute $v = H_1(h, \pi)$, $V = PK_{j,2}^v$ and $W = H_2(g^v) \oplus (h||\pi)$.
  - Define $RK_{i \rightarrow j}^{\langle 1 \rangle} = \frac{h}{x_{i,1} H_4(PK_{i,2}) + x_{i,2}}$.
  - Return $RK_{i \rightarrow j} = (RK_{i \rightarrow j}^{\langle 1 \rangle}, V, W)$.
- **Encrypt**$(m, PK_i, PARAMS)$**:** To encrypt a message $m \in \mathcal{M}$ under PK_i:
  - Pick $u \in_R \mathbb{Z}_q^*$, $\omega \in_R \{0,1\}^{l_1}$.
  - Compute $D = \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^u$.
  - Compute $r = H_1(m, \omega)$.
  - Compute $E = \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^r$ and $F = H_2(g^r) \oplus (m||\omega)$.
  - Compute $s = u + r \cdot H_3(D, E, F) \mod q$.
  - Output the ciphertext $\sigma_i = (D, E, F, s)$.
- **ReEncrypt**$(\sigma_i, PK_i, PK_j, RK_{i \rightarrow j}, PARAMS)$**:** On input of an original ciphertext $\sigma_i = (D, E, F, s)$ encrypted under the public key of the delegator PK_i, the public key of the delegatee PK_j, the re-encryption key $RK_{i \rightarrow j}$, re-encrypt σ_i into a ciphertext $\hat{\sigma_j}$ under PK_j as follows:
  - Check if the following condition holds to satisfy the well-formedness of ciphertexts, otherwise return $\perp$:

$$\left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^s \stackrel{?}{=} D \cdot E^{H_3(D,E,F)} \tag{1}$$

  - If the condition holds, compute $\hat{E} = E^{RK_{i \rightarrow j}^{\langle 1 \rangle}} = g^{rh}$.
  - Output $\hat{\sigma_j} = (\hat{E}, F, V, W)$.
- **Encrypt**$_1(m, PK_i, PARAMS)$**:** To generate a non-transformable ciphertext under public key PK_i of a message $m \in \mathcal{M}$:
  - Pick $h \in_R \{0,1\}^{l_0}$ and $\pi \in_R \{0,1\}^{l_1}$.
  - Compute $v = H_1(h, \pi)$, $V = PK_{j,2}^v$ and $W = H_2(g^v) \oplus (h||\pi)$.

- Pick $\omega \in_R \{0,1\}^{l_1}$ and compute $r = H_1(m, \omega)$.
- Compute $\hat{E} = (g^r)^h$ and $F = H_2(g^r) \oplus (m||\omega)$.
- Output the non-transformable ciphertext $\hat{\sigma_j} = (\hat{E}, F, V, W)$.

– **Decrypt**$(\sigma_i, PK_i, SK_i, PARAMS)$**:** On input of a ciphertext σ_i, public key PK_i and private key $SK_i = (x_{i,1}, x_{i,2})$, decrypt according to two cases:
  - Original Ciphertext $\sigma_i = (D, E, F, s)$:
    * If Eq. (1) does not hold, return $\perp$.
    * Otherwise, compute $(m||\omega) = F \oplus H_2(E^{\frac{1}{x_{i,1}H_4(PK_{i,2})+x_{i,2}}})$.
    * Return m if $E \stackrel{?}{=} \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^{H_1(m,\omega)}$ holds; else return $\perp$.
  - Transformed /Non-transformable Ciphertext $\hat{\sigma_i} = (\hat{E}, F, V, W)$:
    * Compute $(h||\pi) = W \oplus H_2(V^{1/SK_{i,2}})$ and $(m||\omega) = F \oplus H_2(\hat{E}^{1/h})$.
    * Return m if $V \stackrel{?}{=} PK_{i,2}^{H_1(h,\pi)}$, $\hat{E} \stackrel{?}{=} g^{H_1(m,\omega)\cdot h}$ holds; else return $\perp$.

2.2 Weakness in the Security Proof of Chow *et al.*

In this section, we point out the weakness of the security proof for the PRE scheme by Chow *et al.* [4]. We show that the simulation of the oracles defined in the security proof of the scheme is not consistent with the real algorithm. This allows the adversary to distinguish the simulation run by the challenger from the real system. We demonstrate this flaw by considering the validity of the ciphertexts with respect to the *ReEncrypt* and *Decrypt* algorithm in the real system and the simulation (re-encryption oracle *OReE* and decryption oracles *ODec* respectively). To make it simple, we consider PK_T as the public key of the target user in the challenge phase and the attack is posed in **Phase-II** after the challenge phase is over. We re-encrypt a ciphertext σ_T under PK_T into a ciphertext $\hat{\sigma_j}$ under PK_j (PK_j is corrupt) and further decrypt $\hat{\sigma_j}$. All the computations hereafter are done using PK_T and PK_j.

First, we encrypt a message m under PK_T. Let us consider two forms of ciphertext $\sigma_{Real} = \langle D_{Real}, E_{Real}, F_{Real}, s_{Real}\rangle$ and $\sigma_{Fake} = \langle D_{Fake}, E_{Fake}, F_{Rand}, s_{Fake}\rangle$. σ_{Real} is the ciphertext obtained from the encryption algorithm **Encrypt** (i.e., encryption of m under PK_T by executing the **Encrypt**$(m, PK_T, PARAMS)$). σ_{Fake} is a cooked-up ciphertext that can pass the verification tests of *ReEncrypt* algorithm but not the *Decrypt* algorithm. We denote the algorithm for the construction of σ_{Fake} as **Encrypt**$_{Fake}(m, PK_T)$, which is as follows:

– Pick $u_{Fake} \in_R \mathbb{Z}_q^*$.
– Compute $D_{Fake} = \left((PK_{T,1})^{H_4(PK_{T,2})} PK_{T,2}\right)^{u_{Fake}}$.
– Pick $r_{Rand} \in_R \mathbb{Z}_q^*$. Here it should be noted that r_{Rand} does not follow the actual algorithm, instead it is picked at random from $\mathbb{Z}_q^*$.
– Pick $\omega_{Fake} \in_R \{0,1\}^{l_1}$ and compute $r_{Fake} = H_1(m, \omega_{Fake})$. Note that in the **Encrypt** algorithm, r_{Fake} is the output of H_1 oracle on giving a message and a random string (ω_{Fake}) of size $\{0,1\}^{l_1}$ as input.
– Compute $E_{Fake} = \left((PK_{T,1})^{H_4(PK_{T,2})} PK_{T,2}\right)^{r_{Rand}}$.

- Choose $F_{Rand} \in_R \{0,1\}^{l_0+l_1}$. In the **Encrypt** algorithm, F is the encryption of the message along with a random string ω_{Fake} of length $\{0,1\}^{l_1}$. But in the construction of σ_{Fake}, we note that F_{Rand} is chosen at random.
- Compute $s_{Fake} = u_{Fake} + r_{Rand} H_3(D_{Fake}, E_{Fake}, F_{Rand}) \mod q$.
- Output the ciphertext $\sigma_{Fake} = (D_{Fake}, E_{Fake}, F_{Rand}, s_{Fake})$. Note that, σ_{Fake} passes the ciphertext validation test of Eq. (1).

The important properties possessed by σ_{Real} and σ_{Fake} are:

1. Output of **Decrypt**$(\sigma_{Real}, PK_T, SK_T, PARAMS)$ is m and the output of $ODec(PK_T, \sigma_{Real})$ is m. This is because σ_{Real} is a legitimate ciphertext of m produced by *Encrypt* algorithm.
2. Output of **Decrypt**$(\sigma_{Fake}, PK_T, SK_T, PARAMS)$ and $ODec(PK_T, \sigma_{Fake})$ is $\perp$ as σ_{Fake} fails to satisfy the validity check for the obtained message.
3. σ_{Real} is a valid ciphertext and σ_{Fake} is an invalid ciphertext with respect to both **Decrypt** algorithm and *ODec* oracle. Therefore, the simulation of the decryption algorithm is perfect.
4. Both σ_{Real} and σ_{Fake} are valid ciphertexts corresponding to the **ReEncrypt** algorithm. This is because σ_{Real} is a legitimate ciphertext of m produced by the *Encrypt* algorithm. Again, σ_{Fake} passes the ciphertext verification test of Eq. (1) and the algorithm computes the re-encrypted ciphertext $\hat{\sigma}_{Fake} = (\hat{E}, F, V, W)$ as per the protocol where $\hat{E}_{Fake} = g^{r_{Fake} h}$.

Next, we re-encrypt both σ_{Real} and σ_{Fake} under the public key PK_j of a corrupt user. Let us consider the following notations.

- $\hat{\sigma}_{Real}^{(Scheme)} \leftarrow$ **ReEncrypt**$(\sigma_{Real}, PK_T, PK_j, RK_{T \to j}, PARAMS)$.
- $\hat{\sigma}_{Real}^{(Oracle)} \leftarrow OReE(PK_T, PK_j, \sigma_{Real})$.
- $\hat{\sigma}_{Fake}^{(Scheme)} \leftarrow$**ReEncrypt**$(\sigma_{Fake}, PK_T, PK_j, RK_{T \to j}, PARAMS)$.
- $\hat{\sigma}_{Fake}^{(Oracle)} \leftarrow OReE(PK_T, PK_j, \sigma_{Fake})$.

Observations on $\hat{\sigma}_{Real}^{(Scheme)}$ and $\hat{\sigma}_{Real}^{(Oracle)}$:

1. $\hat{\sigma}_{Real}^{(Scheme)} = \hat{\sigma}_{Real}^{(Oracle)}$.
2. $\hat{\sigma}_{Fake}^{(Scheme)} \neq \hat{\sigma}_{Fake}^{(Oracle)}$.

The reason for observation 1 follows directly from the fact that σ_{Real} is a valid ciphertext. The reason for the violation in observation 2 is that the **ReEncrypt** algorithm is only a function of the re-encryption key but *OReE* oracle makes use of the knowledge of r_{Fake} to generate $\hat{\sigma}_{Fake}^{(Oracle)}$. However, in the construction of σ_{Fake}, r_{Rand} is used in the generation of $\hat{\sigma}_{Fake}^{(Oracle)}$. The question here is, how will the adversary find this difference, that is $\hat{\sigma}_{Fake}^{(Scheme)} \neq \hat{\sigma}_{Fake}^{(Oracle)}$. Let us now demonstrate how the adversary captures this difference shown by the *OReE* oracle simulation and the **ReEncrypt** algorithm.

Distinguishing the Oracle from the Real Algorithm:

1. $\mathcal{C}$ provides the system parameters $PARAMS$ to $\mathcal{A}$.
2. After getting training in **Phase-I**, $\mathcal{A}$ provides two messages m_0 and m_1 of equal length and a target public key PK_T to $\mathcal{C}$.
3. $\mathcal{C}$ generates the challenge ciphertext σ_T and gives as challenge to $\mathcal{A}$.
4. $\mathcal{A}$ now does the following:
   (a) Generate $\sigma_{Fake} = \mathbf{Encrypt}_{Fake}(m_0, PK_T) = (D_{Fake}, E_{Fake}, F_{Rand}, s_{Fake})$. Here $\mathcal{A}$ knows r_{Rand} and u_{Fake}.
   (b) $\mathcal{A}$ queries $OReE(\sigma_{Fake}, PK_T, PK_j, RK_{T\to j}, PARAMS)$. It should noted that v, V, h, π, W are fixed for $T \to j$ delegation.
   (c) **Test:** If $\perp \leftarrow OReE((\sigma_{Fake}, PK_T, PK_j, RK_{T\to j}, PARAMS))$, then $ReEncrypt \neq OReE$ and $\mathcal{A}$ knows that it is not the real system and will abort. Else, $\mathcal{A}$ learns no clue about the simulation.

2.3 Fixing the Flaw

Note that modifying the re-encryption algorithm to fix the flaw is not possible since re-encryption of a valid ciphertext σ_T will always require the knowledge of $r = H_1(m, \omega)$ as no other trapdoor exists to obtain a re-encrypted ciphertext $\hat{\sigma}_j$. Again, the knowledge of the private key of the delegator is necessary to generate the re-encryption keys and re-encrypted ciphertexts. Consequently, we cannot provide a trivial fix to the scheme in order to address the problem. As a solution, we propose a new collusion-resistant unidirectional proxy re-encryption scheme without any pairing operation. We have incorporated additional information to the existing *Encrypt* algorithm along with ciphertext validity checks in both the *Re-Encrypt* and the *Decrypt* algorithm.

3 A Unidirectional Proxy Re-encryption Scheme

- **Setup**(λ)**:** Choose two primes p and q such that $q|p-1$ and the bit-length of q is the security parameter λ. Let $\mathbb{G}$ be a subgroup of $\mathbb{Z}_q^*$ with order q. g is a generator of the group $\mathbb{G}$. Choose five hash functions $H_1 : \{0,1\}^{l_0} \times \{0,1\}^{l_1} \to \mathbb{Z}_q^*, H_2 : \mathbb{G} \to \{0,1\}^{l_0+l_1}, H_3 : \{0,1\}^* \to \mathbb{Z}_q^*, H_4 : \mathbb{G} \to \mathbb{Z}_q^*, H_5 : \mathbb{G}^4 \times \{0,1\}^{l_0+l_1} \to \mathbb{G}$. The hash functions are modelled as random oracles in the security proof reduction. Here l_0 and l_1 are security parameters determined by λ, and the message space $\mathcal{M}$ is $\{0,1\}^{l_0}$. Return the public parameters $PARAM = (q, \mathbb{G}, g, H_1, H_2, H_3, H_4, H_5, l_0, l_1)$.
- **KeyGen**$(U_i, PARAMS)$**:** To generate the private key (SK_i) and the corresponding public key (PK_i) of user U_i:
  - Pick $x_{i,1}, x_{i,2} \in_R \mathbb{Z}_q^*$ and set $SK_i = (x_{i,1}, x_{i,2})$.
  - Compute $PK_i = (PK_{i,1}, PK_{i,2}) = (g^{x_{i,1}}, g^{x_{i,2}})$.
- **ReKeyGen**$(SK_i, PK_i, PK_j, PARAMS)$**:** On input of a user i's private key $SK_i = (x_{i,1}, x_{i,2})$ and public key $PK_i = (PK_{i,1}, PK_{i,2})$ and user j's public key $PK_j = (PK_{j,1}, PK_{j,2})$, generate the re-encryption key $RK_{i\to j}$ as shown:

- Pick $h \in_R \{0,1\}^{l_0}$, $\pi \in_R \{0,1\}^{l_1}$.
- Compute $v = H_1(h, \pi)$, $V = PK_{j,2}^v$ and $W = H_2(g^v) \oplus (h||\pi)$.
- Define $RK_{i\to j}^{\langle 1\rangle} = \frac{h}{x_{i,1}H_4(PK_{i,2})+x_{i,2}}$.
- Return $RK_{i\to j} = (RK_{i\to j}^{\langle 1\rangle}, V, W)$.

– **Encrypt**$(m, PK_i, PARAMS)$**:** To encrypt a message $m \in \mathcal{M}$:
- Pick $u \in_R \mathbb{Z}_q^*$, $\omega \in_R \{0,1\}^{l_1}$.
- Compute $D = \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^u$.
- Compute $\bar{D} = H_5(PK_{i,1}, PK_{i,2}, D, E, F)^u$.
- Compute $r = H_1(m, \omega)$.
- Compute $E = \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^r$.
- Compute $\bar{E} = H_5(PK_{i,1}, PK_{i,2}, D, E, F)^r$.
- Compute $F = H_2(g^r) \oplus (m||\omega)$.
- Compute $s = u + r \cdot H_3(D, \bar{E}, F) \mod q$.
- Output the ciphertext $\sigma_i = (D, \bar{E}, F, s)$.

– **ReEncrypt**$(\sigma_i, PK_i, PK_j, RK_{i\to j}, PARAMS)$**:** On input of an original ciphertext $\sigma_i = (E, \bar{E}, F, s)$ encrypted under public key $PK_i = (PK_{i,1}, PK_{i,2})$, the public keys PK_i and PK_j, a re-encryption key $RK_{i\to j} = (RK_{i\to j}^{\langle 1\rangle}, V, W)$, re-encrypt σ_i into a ciphertext $\hat{\sigma_j}$ under the public key $PK_j = (PK_{j,1}, PK_{j,2})$ as follows:
- Compute D and $\bar{D}$ as follows:

$$\begin{aligned} D &= \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^s \cdot (E^{H_3(E,\bar{E},F)})^{-1} \\ &= \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^u. \\ \bar{D} &= H_5(PK_{i,1}, PK_{i,2}, D, E, F)^s \cdot (\bar{E}^{(E,\bar{E},F)})^{-1} \\ &= H_5(PK_{i,1}, PK_{i,2}, D, E, F)^u. \end{aligned}$$

- Check the well-formedness of the ciphertext by verifying:

$$\left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^s \stackrel{?}{=} D \cdot E^{H_3(E,\bar{E},F)} \tag{2}$$

$$\left(H_5(PK_{i,1}, PK_{i,2}, D, E, F)\right)^s \stackrel{?}{=} \bar{D} \cdot \bar{E}^{H_3(E,\bar{E},F)} \tag{3}$$

- If the above checks fail, return $\perp$. Else, compute $\bar{E} = E^{RK_{i\to j}^{\langle 1\rangle}} = g^{rh}$.
- Output $\hat{\sigma_j} = (\bar{E}, F, V, W)$.

– **Encrypt**$_1(m, PK_i, PARAMS)$**:** To generate a non-transformable ciphertext under public key PK_i of a message $m \in \mathcal{M}$:
- Pick $h \in_R \{0,1\}^{l_0}$ and $\pi \in_R \{0,1\}^{l_1}$.
- Compute $v = H_1(h, \pi)$, $V = PK_{j,2}^v$ and $W = H_2(g^v) \oplus (h||\pi)$.
- Pick $\omega \in_R \{0,1\}^{l_1}$ and compute $r = H_1(m, \omega)$.
- Compute $\hat{E} = (g^r)^h$ and $F = H_2(g^r) \oplus (m||\omega)$.
- Output the non-transformable ciphertext $\hat{\sigma_j} = (\hat{E}, F, V, W)$.

– **Decrypt**$(\sigma_i, PK_i, SK_i, PARAMS)$**:** On input a ciphertext σ_i, public key PK_i and private key $SK_i = (x_{i,1}, x_{i,2})$, decrypt according to two cases:
  - Original ciphertext of the form $\sigma_i = (E, \bar{E}, F, s)$:
    * Check if the ciphertext is well-formed by computing the values of D and $\bar{D}$ and checking if Eqs. (2) and (3) holds.
    * If the conditions hold, extract $(m||\omega) = F \oplus H_2(E^{\frac{1}{x_{i,1}H_4(PK_{i,2})+x_{i,2}}})$, else return $\perp$.
    * Return m if the following checks hold, else return $\perp$.

$$E \stackrel{?}{=} \left(PK_{i,1}^{H_4(PK_{i,2})} PK_{i,2}\right)^{H_1(m,\omega)}$$
$$\bar{E} \stackrel{?}{=} H_5(PK_{i,1}, PK_{i,2}, D, E, F)^{H_1(m,\omega)}$$

  - Transformed or non-transformable ciphertext of the form $\sigma_i = (\hat{E}, F, V, W)$:
    * Compute $(h||\pi) = W \oplus H_2(V^{1/SK_{i,2}})$, extract $(m||\omega) = F \oplus H_2(\hat{E}^{1/h})$.
    * Return m if $V \stackrel{?}{=} PK_{i,2}^{H_1(h,\pi)}$, $\hat{E} \stackrel{?}{=} g^{H_1(m,\omega)\cdot h}$ holds; else return $\perp$.

3.1 Correctness

Due to space constraints, the correctness of our scheme is given in the full version of the paper [9].

3.2 Security Proof

Original Ciphertext Security:

Theorem 1. *The proposed scheme is CCA-secure for the original ciphertext under the DCDH assumption and the EUF−CMA security of Schnorr signature scheme [8]. If a (t, ϵ)IND-PRE-CCA $\mathcal{A}$ with an advantage ϵ breaks the IND-PRE-CCA security of the given scheme in time t, $\mathcal{C}$ can solve the DCDH problem with advantage ϵ' within time t' where:*

$$\epsilon' \geq \frac{1}{q_{H_2}}\left(\frac{\epsilon}{e(q_{RK}+1)} - \frac{q_{H_1}}{2^{l_1}} - \frac{q_{H_3}+q_{H_5}}{2^{l_0+l_1}} - q_d\left(\frac{q_{H_1}+q_{H_2}}{2^{l_0+l_1}} + \frac{2}{q}\right) - \epsilon_1 - \epsilon_2\right),$$
$$t' \leq t + (q_{H_1} + q_{H_2} + q_{H_3} + q_{H_4} + q_{H_5} + n_h + n_c + q_{RK} + q_{RE} + q_d)O(1)$$
$$+ (2n_h + 2n_c + 2q_{RK} + 5q_{RE} + 2q_d + q_{H_1}q_{RE} + (2q_{H_2} + 2q_{H_1})q_d)t_e,$$

We note that e is the base of natural logarithm, ϵ_1 denotes the advantage in breaking the CCA security of the hashed Elgamal encryption scheme and ϵ_2 denotes the advantage in breaking the EUF-CMA security of the Schnorr Signature scheme and t_e denotes the time taken for exponentiation in group $\mathbb{G}$.

Proof. Due to space constraints, the proof of the theorem is shown in the full version of the paper [9].

Transformed Ciphertext Security:

Theorem 2. *The proposed scheme is CCA-secure for the transformed ciphertext under the CDH assumption and the $EUF-CMA$ security of Schnorr signature scheme [8]. If a $(t,\epsilon)IND\text{-}PRE\text{-}CCA$ $\mathcal{A}$ with an advantage ϵ breaks the IND-CPRE-CCA security of the given scheme, $\mathcal{C}$ can solve the DCDH problem with advantage ϵ' within time t' where:*

$$\epsilon' \geq \frac{1}{q_{H_2}}\left(\frac{2\epsilon}{e(2+q_{RK})^2} - \frac{q_{H_1}}{2^{l_1}} - q_d\left(\frac{q_{H_1}+q_{H_2}}{2^{l_0+l_1}} + \frac{2}{q}\right) - \epsilon_2\right),$$
$$t' \leq t + (q_{H_1} + q_{H_2} + q_{H_3} + q_{H_4} + q_{H_5} + n_h + n_c + q_{RK} + q_{RE} + q_d)O(1)$$
$$+ (2n_h + 2n_c + 2q_{RK} + 3q_{RE} + 2q_d + (2q_{H_2} + 2q_{H_1})q_d)t_e,$$

Proof. The proof of the theorem is shown in the full version of the paper [9]. □

Non-transformable Ciphertext Security:

Theorem 3. *The proposed scheme is CCA-secure for the non-transformable ciphertext under the CDH assumption. If a $(t,\epsilon-\epsilon_2)IND\text{-}PRE\text{-}CCA$ $\mathcal{A}$ with an advantage $\epsilon-\epsilon_2$ breaks the IND-PRE-CCA security of the given scheme, $\mathcal{C}$ can solve the CDH problem with advantage ϵ' within time t' where:*

$$\epsilon' \geq \frac{1}{q_{H_2}}\left(\epsilon - \epsilon_2 - \frac{q_{H_1}}{2^{l_1}} - q_d\left(\frac{q_{H_1}+q_{H_2}}{2^{l_0+l_1}} + \frac{2}{q}\right)\right),$$
$$t' \leq t + (q_{H_1} + q_{H_2} + q_{H_3} + q_{H_4} + q_{H_5} + n_h + n_c + q_{RK} + q_d)O(1)$$
$$+ (2n_h + 2n_c + 2q_{RK} + 2q_d + (2q_{H_2} + 2q_{H_1})q_d)t_e,$$

Proof. The proof of the theorem is shown in the full version of the paper [9].

Delegator Secret Security:

Theorem 4. *The proposed scheme is DSK-secure under the DL assumption. If a $(t,\epsilon)DSK$ $\mathcal{A}$ with an advantage ϵ breaks the DSK security of the given scheme in time t, $\mathcal{C}$ can solve the DL problem with advantage ϵ within time t' where:*

$$t' \leq t + O(2q_{RK} + 2n_h + 2n_c)t_e,$$

Proof. The proof of the theorem is shown in the full version of the paper [9]. □

4 Efficiency Comparison

We give a comparison of our scheme with the modified scheme of Chow *et al.* by Canard *et al.* [2] in Table 1. We show the computational efficiency of our PRE scheme, and use t_e to denote the time for exponentiation operation. Note that $l = O(\log \lambda)$ denotes the number of commitments generated by the signer in the NIZK proof in the encryption protocol in [2]. The comparison shows that our proposed design is more efficient than the existing fix to the pairing-free unidirectional PRE scheme of Chow *et al.* constructed by Canard *et al.* [2].

Table 1. Comparative analysis of the modified pairing-free PRE scheme due to Canard *et al.* and our scheme. Note that $l = O(\log \lambda)$.

Algorithm	[2]	Our scheme
KeyGen	$2t_e$	$2t_e$
ReKeyGen	$2t_e$	$2t_e$
Encrypt_1	$7t_e$	$4t_e$
Encrypt	$(3 + l)t_e$	$5t_e$
ReEncrypt	$4t_e$	$6t_e$
Decrypt (original)	$7t_e$	$4t_e$
Decrypt (transformed)	$3t_e$	$8t_e$

5 Conclusion

Although pairing is an expensive operation, only one scheme due to Chow *et al.* [4] reported the pairing-free unidirectional property with collusion-resistance. In this paper, we point out that the security proof in the scheme is flawed, where the adversary is able to determine that the simulation provided by challenger is not consistent with real system. Also, we present a construction of unidirectional proxy re-encryption scheme without bilinear pairing that provides collusion-resistance, and show that our scheme is more efficient than the modified scheme of Chow *et al.* constructed by Canard *et al.* Our scheme is proven CCA-secure under a variant of the computational Diffie-Hellman assumption in the random oracle model.

References

1. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: Proceedings of the Network and Distributed System Security Symposium, NDSS 2005, San Diego, California, USA (2005)
2. Canard, S., Devigne, J., Laguillaumie, F.: Improving the security of an efficient unidirectional proxy re-encryption scheme. J. Internet Serv. Inf. Secur. **1**(2/3), 140–160 (2011)
3. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: Proceedings of the 2007 ACM Conference on Computer and Communications Security, CCS 2007, Alexandria, Virginia, USA, 28–31 October 2007, pp. 185–194 (2007)
4. Chow, S.S.M., Weng, J., Yang, Y., Deng, R.H.: Efficient unidirectional proxy re-encryption. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 316–332. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12678-9_19
5. Deng, R.H., Weng, J., Liu, S., Chen, K.: Chosen-ciphertext secure proxy re-encryption without pairings. In: Franklin, M.K., Hui, L.C.K., Wong, D.S. (eds.) CANS 2008. LNCS, vol. 5339, pp. 1–17. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-89641-8_1

6. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. IEEE Trans. Inf. Theory **57**(3), 1786–1802 (2011)
7. Mambo, M., Okamoto, E.: Proxy cryptosystems: Delegation of the power to decrypt ciphertexts. IEICE Trans. Fundam. Electron. Commun. Comput. Sci. **80**(1), 54–63 (1997)
8. Schnorr, C.-P.: Efficient signature generation by smart cards. J. Cryptol. **4**(3), 161–174 (1991)
9. Sharmila Deva Selvi, S., Paul, A., Rangan, C.P.: A provably-secure unidirectional proxy re-encryption scheme without pairing in the random oracle model (full version). Cryptology ePrint Archive, October 2017
10. Weng, J., Deng, R.H., Liu, S., Chen, K.: Chosen-ciphertext secure bidirectional proxy re-encryption schemes without pairings. Inf. Sci. **180**(24), 5077–5089 (2010)