Direct Constructions of Bidirectional Proxy Re-Encryption with Alleviated Trust in Proxy

Jian Weng[1], Yunlei Zhao[2*]
[1] Department of Computer Science, Jinan University, Guangzhou, China
[2] Software School, Fudan University, Shanghai, China
cryptjweng@gmail.com ylzhao@fudan.edu.cn

Abstract

In this work, we study (the direct constructions of) bidirectional proxy re-encryption (PRE) with alleviated trust in the proxy, specifically the master secret security (MSS) and the non-transitivity (NT) security, in the standard model, and achieve the following:

- A *multi-hop MSS-secure* bidirectional PRE scheme with security against chosen plaintext attacks (CPA) in the standard model, where the ciphertext remains constant size regardless how many times it has been re-encrypted. To the best of our knowledge, there exists previously no MSS-secure multi-hop bidirectional PRE scheme with constant size of ciphertexts (whether in the random oracle model or not).
- A *single-hop MSS-secure* and *non-transitive* bidirectional PRE scheme with security against *chosen ciphertext attacks* (CCA) in the standard model. The CCA-secure scheme is based on the CPA-secure scheme, and particularly employs a new re-encryption key (REK) generation mechanism to which each user makes equal contributions, where a *single* REK is used in both directions with the same computation so that the proxy needs not to distinguish the transform direction when it re-encrypts ciphertexts. Besides alleviated trust in proxy, single-hop non-transitive bidirectional PRE schemes also enjoy better fine-grained delegate right control (against malicious proxy).

The security analysis uses Coron's technique [Coron, Crypto 2000], which particularly allows adaptive secret-key corruption. Along the way, we also refine and clarify the security models for bidirectional PRE.

1 Introduction

Proxy re-encryption (PRE), introduced in [7], allows a proxy to transform ciphertexts computed under the public-key of Alice (the delegator) into other ciphertexts for Bob (the delegatee). The proxy, however, learns nothing about the underlying messages encrypted, and has no knowledge of the secret keys of the delegators and delegatees. PRE has many applications (for example, encrypted email forwarding, outsourced filtering of encrypted spam, law enforcement, secure file systems, and performing cryptographic operations on storage-limited devices, etc), and becomes increasingly popular as a method for managing encrypted file systems [3–5, 14, 21, 22, 24, 28–30].

According to the direction of transformation, PRE can be classified into two types: *unidirectional* and *bidirectional* [24]. In unidirectional PREs, the proxy can only transform ciphertexts from Alice to Bob. While in bidirectional PREs, the proxy can transform ciphertexts in both directions. Both types of PRE have their respective interesting applications. In this paper, we shall concentrate on bidirectional PRE. As an example to illustrate the application of bidirectional PRE, consider the following scenario: Alice and Bob collaborate to conduct a secret assignment, and they need to share their encrypted data in a fair manner (i.e., Alice needs to read all of Bob's encrypted data, and Bob needs also to read all of Alice's encrypted data). Indeed, this can be done

*Contact Author

by exchanging their secret keys. However, such a solution is highly unsatisfactory, since it places too much trust on each other. Bidirectional PRE can provide solutions in a more desirable way for such scenarios. PRE can also be categorized into *multi-hop* PRE, in which the ciphertexts can be transformed from Alice to Bob and then to Charlie and so on, and *single-hop* PRE, in which the ciphertexts can only be transformed once [24]. In this work, as is typical in this literature, we only consider *single-key* PRE schemes, where both the re-encryption key (REK) and the secret-key of each user (delegator or delegatee) consist of only a *single* group element.

Up to now, the only known CCA-secure bidirectional PRE scheme in the standard model is the one proposed by Canetti and Hohenberger [14] (referred to as CH07 scheme), which is *multi-hop* and is based on Blaze et al.'s [7] CPA-secure bidirectional PRE scheme. The generation of re-encryption key (REK) in both these schemes [7,14] uses the following strategy: for two users (say, Alice and Bob) with secret keys x_a and x_b respectively, the re-encryption key is simply computed as $rk_{a\leftrightarrow b} = \frac{x_b}{x_a}$. Actually, $\frac{x_b}{x_a}$ is used for re-encryption from Alice to Bob, and for re-encryption from Bob to Alice, the proxy uses $rk^{-1}_{a\leftrightarrow b} = \frac{x_a}{x_b}$ (that is, the proxy action in the two re-encryption directions is not identical). Unfortunately, in this way, if the proxy (who knows the re-encryption key) colludes with one of the user, it can recover the other user's secret key. It is worth noting that, in practice, a user may also use its secret-key to perform other cryptographic operations such as signing, identifications, etc (cf. [4,5]). This means that the collusion in such bidirectional PRE schemes not only surrenders the user's decryption right, but also can totally surrender the user's digital identity [4,5]. Another observation on the REK generation mechanism of [7,14] is the *transitivity* of REK, by which the proxy *alone* can create delegation rights between two entities that have never agreed on this. Specifically, from the REK $rk_{a\leftrightarrow b} = \frac{x_b}{x_a}$ between Alice and Bob and the REK $rk_{b\leftrightarrow c} = \frac{x_c}{x_b}$ between Bob and Charlie, the proxy alone can compute the REK $rk_{a\leftrightarrow c} = rk_{a\leftrightarrow b} \cdot rk_{b\leftrightarrow c} = \frac{x_c}{x_a}$ between Alice and Charlie. We note that *multi-hop* PRE schemes intrinsically suffer from such delegation right uncontrol against a malicious proxy, as what can be done by the proxy with the created REK $rk_{a\to c}$ can actually be done with the existing REKs $rk_{a\to b}$ and $rk_{b\to c}$ (though with higher computational complexity). That is, for *multi-hop* PRE, users may have to rely upon the trust of the proxy to not generate delegation rights to unnegotiated users, *even if the underlying REK generation is non-transitive*. But, for *single-hop* PRE, where a transformed ciphertext generated by the proxy cannot be further transformed, if the underlying REK generation is non-transitive a malicious proxy itself may be unable to delegate rights among unnegotiated users. This indicates that, suppose the underlying REK generation is non-transitive, *single-hop* PRE can enjoy better fine-grained delegation right control (against malicious proxy) than *multi-hop* PRE. Motivated by the above observations and aimed for alleviating the trust in the proxy, Ateniese et al. [4,5] introduced for PRE the notions of non-transitivity (NT) security (which informally says that given re-encryption keys $rk_{a\to b}$ and $rk_{b\to c}$ no efficient algorithm can compute $rk_{a\to c}$) and master secret security (MSS) (which informally says that the delegator's secret-key cannot be derived even if the proxy and the delegatee collude).

Several CPA-secure *single-hop unidirectional* PRE schemes with the MSS security and NT security in the standard model were proposed by Ateniese et al. [4,5]. Based on the schemes of [4,5], Libert and Vergnaud [29,30] proposed more advanced *single-hop unidirectional* PRE schemes with the MSS security and NT security in the standard model. The PRE scheme proposed in [29,30] is secure against *replayable chosen ciphertext attacks* (RCCA), which is a weaker variant of the CCA security in the sense that the re-randomizing of the challenge ciphertext is tolerated.

Canetti and Hohenberger [14] ever mentioned a generic method for transforming any unidirectional PRE into a bidirectional one by generating two unidirectional REKs between Alice and Bob (i.e., $rk_{a\to b}$ and $rk_{b\to a}$) and setting the bidirectional REK to be $rk_{a\leftrightarrow b} = (rk_{a\to b}, rk_{b\to a})$. However, since in unidirectional PRE schemes the REK of one direction (e.g., $rk_{a\to b}$) cannot be computed from the REK of the opposite direction (e.g., $rk_{b\to a}$), the complexity of REK generation can be doubled with this generic method (as already noted in [14]). As REK generation is the key component for a PRE scheme, REK generation usually assumes *secure* and *authenticated* communication channels in practice (the actual implementations of such channels can themselves cause much overload to the PRE system in reality) or some (usually impractical) secure multi-party computation protocols (particularly for some exisiting *bidirectional* PRE schemes). Also, for proxy cryptosystems, the workload of the proxy is usually the bottle-neck of the whole system. Thus, it is naturally desirable to devise direct constructions of bidirectional PRE schemes with alleviated trust in the proxy (say, the MSS and NT security).

Still no direct constructions of MSS-secure or NT-secure (whether CPA-secure or CCA-secure) bidirectional PRE are known (whether in the random oracle model or not). Note that, even with the above generic transformation method [14], still no fully CCA secure bidirectional PRE with the MSS and NT security in the standard model is known. Moreover, even with the above generic transformation method, up to now there still exists no MSS-secure *multi-hop* bidirectional PRE schemes with constant size of ciphertexts. So, a further interesting question is how to construct MSS-secure multi-hop PRE schemes with constant size of ciphertexts, even with only chosen plaintext security.

Our contributions. In this paper, we deal with direct constructions of (single-key) bidirectional PRE schemes with the MSS security and NT security in the standard model, and achieve the following contributions:

1. We present the first *MSS-secure multi-hop* (bidirectional) PRE scheme with constant size of ciphertexts. We prove its CPA security under the DBDH assumption, and its MSS security under the discrete logarithm (DL) assumption.
2. We present (direct construction of) a *CCA-secure single-hop* bidirectional PRE scheme *with both the MSS security and the NT security.* The CCA-security of transformed ciphertexts is based upon the 2-weak decisional bilinear Diffie-Hellman inverse (2-wDBDHI) assumption [9]. The CCA-security of original ciphertexts (resp., the NT-security that is stronger than the CCA-security of original ciphertexts) is based upon a newly introduced assumption, called 1-augmented wDBDHI (1-AwDBDHI) (resp., 2-AwDBDHI), with complexity of only about twice (resp., thrice) of that in solving the decisional bilinear Diffie-Hellman (DBDH) assumption in the generic bilinear group model.

Our PRE constructions are novel: the CPA-secure scheme is new; the CCA-secure scheme is based on the CPA-secure scheme, and particularly employs a new re-encryption key generation mechanism, similar to Diffie-Hellman key-exchange (DHKE) but w.r.t. a different generator, to which each user makes equal contributions. Besides rendering the MSS security and the NT security, another particular advantage of this DHKE-like re-encryption key generation is that, when queried for re-encrypting ciphertexts, (unlike in all previous bidirectional PRE schemes) the proxy does not need to distinguish the re-encryption direction (i.e., from Alice to Bob, or vice versa) as its actions for both directions are the same. The security analysis uses Coron's technique [Coron, Crypto 2000], which particularly allows adaptive secret-key corruption. Along the way, we refine the security models for (particularly, single-hop) bidirectional PRE, including CCA security, MSS security and NT security, and clarify the subtleties surrounding security formulations.

Compared with the CPA-secure bidirectional PRE scheme proposed by Blaze, Bleumer and Strauss in Eurocrypt'98 [7], our CPA-secure PRE scheme gains the advantage of MSS-security. In comparison with the only known CCA-secure bidirectional PRE scheme in the standard model [14], our CCA-secure scheme adds the MSS and NT security, the property of single-key re-encryption with the same computation in both directions, and is also essentially more efficient and has shorter ciphertexts. Also, NT-secure single-hop PRE enjoys better fine-grained delegation right control (against malicious proxy) than multi-hop bidirectional PRE. The CH07 scheme is *multi-hop* and is based on the DBDH assumption; our CCA-secure scheme is *single-hop* (where both original ciphertext CCA security and transformed ciphertext CCA security need to be dealt with), and is based on relatively non-standard assumptions than DBDH. Achieving CCA-secure *single-hop* bidirectional PRE (with the MSS and NT security) under weaker assumptions is left as an interesting open problem for future exploration. Detailed comparison with the Canetti-Hohenberger scheme [14] is presented in Appendix A.

2 Preliminaries

Notations. We use $x \xleftarrow{\$} \mathcal{S}$ to denote that an element x is chosen uniformly at random from S. For a string x, $|x|$ denotes its bit-length. We use $\mathcal{A}(x_1, x_2, \cdots)$ to denote that $\mathcal{A}$ is an algorithm with the input $(x_1, x_2, \cdots)$, and use $y \leftarrow \mathcal{A}(x_1, x_2, \cdots)$ to denote the running of $\mathcal{A}(x_1, x_2, \cdots)$ with the output y. For $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x_1, x_2, \cdots)$, it means that $\mathcal{A}$ is an algorithm with the input $(x_1, x_2, \cdots)$ and can access to oracles $\mathcal{O}_1, \mathcal{O}_2, \cdots$. By $y \leftarrow \mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x_1, x_2, \cdots)$, we denote the running of $\mathcal{A}^{\mathcal{O}_1, \mathcal{O}_2, \cdots}(x_1, x_2, \cdots)$ with the output y. Let $\mathbb{G}$ and $\mathbb{G}_T$ be two cyclic multiplicative groups with the same prime order p. A *bilinear pairing* is a map $e : \mathbb{G} \times \mathbb{G} \to \mathbb{G}_T$ with the following properties: (1) Bilinearity: $\forall g_1, g_2 \in \mathbb{G}, \forall a, b \in \mathbb{Z}_p$, we have $e(g_1^a, g_2^b) = e(g_1, g_2)^{ab}$; (2) Non-degeneracy: There exist $g_1, g_2 \in \mathbb{G}$ such that $e(g_1, g_2) \neq 1_{\mathbb{G}_T}$, where $1_{\mathbb{G}_T}$ is the identity element in group $\mathbb{G}_T$; (3) Computability: There exists an efficient algorithm to compute $e(g_1, g_2)$ for $\forall g_1, g_2 \in \mathbb{G}$. *Throughout this paper, we denote by τ the maximum over the time to compute an exponentiation, a multi-exponentiation and a pairing. To distinguish from the notation of pairing operation, we use $\dot{e}$ to denote the base of the natural logarithm.*

Definition 1. *The decisional bilinear Diffie-Hellman (*DBDH*) problem [10] in groups $(\mathbb{G}, \mathbb{G}_T)$ is, given $(g, g^a, g^b, g^c, Z) \in \mathbb{G}^4 \times \mathbb{G}_T$ with unknown $a, b, c \xleftarrow{\$} \mathbb{Z}_p^*$, decide whether $Z = e(g, g)^{abc}$. For an adversary $\mathcal{B}$, we define its advantage in solving the* DBDH *problem in groups $(\mathbb{G}, \mathbb{G}_T)$ as*

$$\mathrm{Adv}_{\mathcal{B}}^{\mathrm{DBDH}} \triangleq \left| \Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^{abc}) = 1 | a, b, c \xleftarrow{\$} \mathbb{Z}_p^*] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, e(g,g)^z) = 1 | a, b, c, z \xleftarrow{\$} \mathbb{Z}_p^*] \right|.$$

*We say that the (t, ϵ)-*DBDH *assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if no t-time adversary $\mathcal{B}$ has advantage at least ϵ in solving the* DBDH *problem in $\mathbb{G}$.*

Definition 2. *The q-weak decisional bilinear Diffie-Hellman inversion assumption (q-wDBDHI) [9] in groups $(\mathbb{G}, \mathbb{G}_T)$ is, given $(g, g^a, \cdots, g^{a^q}, g^b, Z) \in \mathbb{G}^{q+2} \times \mathbb{G}_T$ with unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, to decide whether $Z = e(g, g)^{b/a}$. For an adversary $\mathcal{B}$, we define its advantage $\mathrm{Adv}_{\mathcal{B}}^{q\text{-wDBDHI}}$ in solving the q-*wDBDHI *problem as*

$$\left| \Pr[\mathcal{B}(g, g^a, \cdots, g^{a^q}, g^b, e(g,g)^{\frac{b}{a}}) = 1 | a, b \xleftarrow{\$} \mathbb{Z}_p^*] - \Pr[\mathcal{B}(g, g^a, \cdots, g^{a^q}, g^b, e(g,g)^z) = 1 | a, b, z \xleftarrow{\$} \mathbb{Z}_p^*] \right|,$$

*We say that the (t, ϵ)-q-*wDBDHI *assumption holds if no t-time adversary $\mathcal{B}$ has advantage at least ϵ in solving the q-*wDBDHI.

In this paper, our proposed scheme shall also use a variant of q-wDBDHI assumption named q *augmented* weak decisional bilinear Diffie-Hellman inversion assumption (q-AwDBDHI), which is almost identical to the q-wDBDHI assumption except introducing the additional term g^{1/a^2}.

Definition 3. *The q-AwDBDHI assumption in groups $(\mathbb{G}, \mathbb{G}_T)$ is, given $(g, g^{1/a^2}, g^a, \cdots, g^{a^q}, g^b, Z) \in \mathbb{G}^{q+3} \times \mathbb{G}_T$ with unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, to decide whether $Z = e(g, g)^{b/a}$. For an adversary $\mathcal{B}$, we define its advantage $\mathrm{Adv}_{\mathcal{B}}^{q\text{-AwDBDHI}}$ in solving the q-*AwDBDHI *problem as*

$$\left| \Pr[\mathcal{B}(g, g^{1/a^2}, g^a, \cdots, g^{a^q}, g^b, e(g,g)^{\frac{b}{a}}) = 1 | a, b \xleftarrow{\$} \mathbb{Z}_p^*] - \Pr[\mathcal{B}(g, g^{1/a^2}, g^a, \cdots, g^{a^q}, g^b, e(g,g)^z) = 1 | a, b, z \xleftarrow{\$} \mathbb{Z}_p^*] \right|,$$

*We say that the (t, ϵ)-q-*AwDBDHI *assumption holds in $(\mathbb{G}, \mathbb{G}_T)$ if no t-time adversary $\mathcal{B}$ has advantage at least ϵ in solving the q-*AwDBDHI.

Note that, introducing the additional term g^{1/a^2} still does not appear to ease the computation of $e(g, g)^{b/a}$, since the input vector is missing the term g^{ab}. The hardness of q-AwDBDHI is implied by that of general Diffie-Hellman problem in the generic bilinear group model [9]. In particularly, a generic attacker's advantage in solving the 1-AwDBDHI (resp. 2-AwDBDHI) assumptions (on which our CCA-secure PRE scheme is based) is only about twice (resp. thrice) of that in solving the DBDH problem. The reader is referred to Appendix C for details.

3 Security Model for Bidirectional PRE

Formally, a bidirectional PRE scheme consists of the following six algorithms [14]: (1) Setup algorithm $\mathsf{Setup}(k)$, which on input a security parameter k outputs the global parameter $param$; (2) Key generation algorithm $\mathsf{KeyGen}(param)$, by which each user i generates its public/private key pair (pk_i, sk_i); (3) REK generation algorithm $\mathsf{ReKeyGen}(sk_i, sk_j)$, which on input two private keys sk_i and sk_j outputs a re-encryption key $rk_{i\leftrightarrow j}$ to the proxy; (4) Encryption algorithm $\mathsf{Enc}(pk_i, m)$, which outputs a ciphertext CT_i of $m \in \mathcal{M}$ under pk_i (Here $\mathcal{M}$ is the message space); (5) Re-encryption algorithm $\mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathrm{CT}_i)$, which, on input a re-encryption key $rk_{i\leftrightarrow j}$ and an original ciphertext CT_i under public key pk_i, outputs a transformed ciphertext CT_j under public key pk_j; (6) Decryption algorithm $\mathsf{Dec}(sk_z, \mathrm{CT}_z)$, which, on input a private key sk_z and a ciphertext CT_z, outputs a message $m \in \mathcal{M}$ or the error symbol $\perp$ (if CT_z is invalid). We remark that for each user the secret-key used for decryption and that for REK generation are the same.

Next, we introduce the following oracles which will be used to model the abilities of the adversary: (1) Public key oracle $\mathcal{O}_{\mathrm{pk}}(\cdot)$: On input $\mathcal{O}_{\mathrm{pk}}(i)$, it runs $(pk_i, sk_i) \leftarrow \mathsf{KeyGen}(param)$ and returns pk_i. (2) Secret key oracle $\mathcal{O}_{\mathrm{sk}}(\cdot)$: On input $\mathcal{O}_{\mathrm{sk}}(pk_i)$, it returns the corresponding secret key sk_i. (3) Re-encryption key oracle $\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot)$: On input $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$, it returns $rk_{i\leftrightarrow j} \leftarrow \mathsf{ReKeyGen}(sk_i, sk_j)$. (4) Re-encryption oracle $\mathcal{O}_{\mathrm{re}}(\cdot,\cdot,\cdot)$: On input $\mathcal{O}_{\mathrm{re}}(pk_i, pk_j, \mathtt{CT}_i)$, it returns $\mathtt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$. (5) Decryption oracle $\mathcal{O}_{\mathrm{d}}(\cdot,\cdot)$: On input $\mathcal{O}_{\mathrm{d}}(pk_z, \mathtt{CT}_z)$, it returns the result of $\mathsf{Dec}(sk_z, \mathtt{CT}_z)$, where sk_z is the secret key w.r.t to public keys pk_z. Note that for the last four oracles, it is required that pk_i, pk_j and pk_z are all generated by oracle $\mathcal{O}_{\mathrm{pk}}$.

CCA security (of original ciphertexts) for bidirectional PRE. For an adversary $\mathcal{A}$ running in two stages `find` and `guess`, we define its advantage as

$$\mathrm{Adv}^{\textsf{IND-PRE-oCCA}}_{\mathrm{PRE},\mathcal{A}}(k) = \left|\Pr\left[\mathbf{Exp}^{\textsf{IND-PRE-oCCA-1}}_{\mathrm{PRE},\mathcal{A}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\textsf{IND-PRE-oCCA-0}}_{\mathrm{PRE},\mathcal{A}}(k) = 1\right]\right|,$$

or equivalently (by some standard trick [6, 20]),

$$\mathrm{Adv}^{\textsf{IND-PRE-oCCA}}_{\mathrm{PRE},\mathcal{A}}(k) = |2\Pr[\delta' = \delta] - 1|,$$

where $\mathbf{Exp}^{\textsf{IND-PRE-oCCA-}\delta}_{\mathrm{PRE},\mathcal{A}}$ is defined by the following experiment:

Experiment $\mathbf{Exp}^{\textsf{IND-PRE-oCCA-}\delta}_{\mathrm{PRE},\mathcal{A}}(k)$
$param \leftarrow \mathsf{Setup}(1^k)$;
$(m_0, m_1, pk_{i^*}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{re}}(\cdot,\cdot,\cdot),\mathcal{O}_{\mathrm{d}(\cdot,\cdot)}}(\mathtt{find}, param)$, where $|m_0| = |m_1|$;
$\mathtt{CT}^* \leftarrow \mathsf{Enc}(pk_{i^*}, m_\delta)$, where $\delta \xleftarrow{\$} \{0,1\}$;
$\delta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{re}}(\cdot,\cdot,\cdot),\mathcal{O}_{\mathrm{d}(\cdot,\cdot)}}(\mathtt{guess}, param, \mathtt{CT}^*)$;
return δ'.

During the above experiment, as in [14] it is required that the following requirements are simultaneously satisfied:

1. For any re-encryption key query $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$, it is required that either both the secret keys with respect to pk_i and pk_j are corrupted (by oracle queries to $\mathcal{O}_{\mathrm{sk}}$ by $\mathcal{A}$) or alternately both are uncorrupted;
2. pk_{i^*} is generated by oracle $\mathcal{O}_{\mathrm{pk}}$, and $\mathcal{A}$ cannot issue the secret key query $\mathcal{O}_{\mathrm{sk}}(pk_{i^*})$;
3. $\mathcal{A}$ cannot simultaneously issue $\mathcal{O}_{\mathrm{sk}}(pk_j)$ and $\mathcal{O}_{\mathrm{re}}(pk_i, pk_j, \mathtt{CT}_i)$ such that $(pk_i, \mathtt{CT}_i)$ is a derivative of $(pk_{i^*}, \mathtt{CT}^*)$. (The derivatives of $(pk_{i^*}, \mathtt{CT}^*)$ are defined below.)
4. $\mathcal{A}$ cannot issue $\mathcal{O}_{\mathrm{d}}(pk_z, \mathtt{CT}_z)$ such that $(pk_z, \mathtt{CT}_z)$ is a derivative of $(pk_{i^*}, \mathtt{CT}^*)$.

The derivatives of $(pk_i, \mathtt{CT}_i)$ are inductively defined by the following rules:

Rule-1: $(pk_i, \mathtt{CT}_i)$ is a derivative of itself.

Rule-2: If $(pk_j, \mathtt{CT}_j)$ is a derivative of $(pk_i, \mathtt{CT}_i)$ and $(pk_z, \mathtt{CT}_z)$ is a derivative of $(pk_j, \mathtt{CT}_j)$, then $(pk_z, \mathtt{CT}_z)$ is a derivative of $(pk_i, \mathtt{CT}_i)$.

Rule-3: If $\mathcal{A}$ has issued a re-encryption query $\mathcal{O}_{\mathrm{re}}(pk_i, pk_j, \mathtt{CT}_i)$ and obtained the resulting transformed ciphertext $\mathtt{CT}_j$, then $(pk_j, \mathtt{CT}_j)$ is a derivative of $(pk_i, \mathtt{CT}_i)$.

Rule-4: If $\mathcal{A}$ has issued a re-encryption key generation query $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$ or $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_i)$, *and* $\mathsf{Dec}(sk_j, \mathtt{CT}_j) \in \{m_0, m_1\}$, then $(pk_j, \mathtt{CT}_j)$ is a derivative of $(pk_i, \mathtt{CT}_i)$.

Remark 1. The above IND-PRE-oCCA security formulation for *single-hop* bidirectional PRE is essentially identical to the IND-PRE-CCA security formulated in [14] for *multi-hop* bidirectional PRE, but have the following key differences (which make IND-PRE-oCCA strictly stronger than IND-PRE-CCA):

- Adaptive vs. non-adaptive secret-key corruption. Our IND-PRE-oCCA formulation allows adaptive secret-key corruption. That is, adversary $\mathcal{A}$ can adaptively learn users' secret-keys by querying the oracle $\mathcal{O}_{\mathrm{sk}}(pk_i)$ (as long as pk_i was formerly generated by the oracle $\mathcal{O}_{\mathrm{pk}}$). In the IND-PRE-CCA formulation of [14], the oracle $\mathcal{O}_{\mathrm{sk}}$ is defined to have no input and work as follows: when queried, $\mathcal{O}_{\mathrm{sk}}$ generates a new pair (pk, sk) by running KeyGen, and returns (pk, sk) to $\mathcal{A}$. Note that $\mathcal{A}$ cannot corrupt the secret-keys corresponding to already existing public-keys. As a consequence, the simulator in the CCA proof of [14] does not need to handle queries of adaptive secret-key corruption.
- Rule-4 difference. The Rule-4 is formulated in [14] as follows: "if $\mathcal{A}$ has queried $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$ or $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_i)$, or alternatively a series of queries $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_{k_1})$, $\cdots, \mathcal{O}_{\mathrm{rk}}(pk_{k_{\tau-1}}, pk_{k_\tau})$, $\mathcal{O}_{\mathrm{rk}}(pk_{k_\tau}, pk_j)$ where $\tau \geq 2$ (which is mentioned in the Rule-4 motivation in [14]), and $\mathsf{Dec}(pk_j, \mathtt{CT}_j) \in \{m_0, m_1\}$, then $(pk_j, \mathtt{CT}_j)$ is the derivative of $(pk_i, \mathtt{CT}_i)$." In contrast, from the series of queries $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_{k_1}), \cdots, \mathcal{O}_{\mathrm{rk}}(pk_{k_{\tau-1}}, pk_{k_\tau}), \mathcal{O}_{\mathrm{rk}}(pk_{k_\tau}, pk_j)$, even if $\mathcal{A}$ can get $rk_{i\leftrightarrow j}$ *but without querying either* $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$ *or* $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_i)$, $(pk_j, \mathtt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i))$ is *not* the derivative of $(pk_i, \mathtt{CT}_i)$ according to our Rule-4 specification.

Remark 2. A further stronger formulation of Rule-4 (which renders further stronger IND-PRE-oCCA security), for bidirectional PRE with *deterministic* re-encryption algorithm ReEnc, is: If $\mathcal{A}$ has issued a re-encryption key generation query $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$ or $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_i)$ to learn $rk_{i\leftrightarrow j}$, and $\mathtt{CT}_j = \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$ then $(pk_j, \mathtt{CT}_j)$ is a derivative of $(pk_i, \mathtt{CT}_i)$. The observation is that the requirement $\mathsf{Dec}(pk_j, \mathtt{CT}_j) \in \{m_0, m_1\}$ in the Rule-4 specification of [14] may be over strict, which particularly excludes re-randomizing the challenge ciphertext $\mathtt{CT}_i^*$. Specifically, consider the following ciphertext re-randomizing attack: given $(pk_i, \mathtt{CT}_i)$ where $\mathtt{CT}_i$ is the original (resp., transformed) ciphertext under public-key pk_i, an adversary $\mathcal{A}$ first re-randomizes $\mathtt{CT}_i$ into $\mathtt{CT}_i' \neq \mathtt{CT}_i$ and then gets $\mathtt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i')$ (resp., $\mathcal{A}$ first gets $\mathtt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$ and then re-randomizes $\mathtt{CT}_j$ into $\mathtt{CT}_j' \neq \mathtt{CT}_j$). Such attacks are forbidden *by definition* within the definitional framework of [14] as $(pk_j, \mathtt{CT}_j)$ (resp., $(\mathrm{pk}_j, \mathtt{CT}_j')$) is still defined to be the derivative of $(pk_i, \mathtt{CT}_i)$, while such an attack is allowed with the above stronger Rule-4 formulation as $(pk_j, \mathtt{CT}_j)$ (resp., $(pk_j, \mathtt{CT}_j')$) is not viewed as the derivative of $(pk_i, \mathtt{CT}_i)$). We show our CCA-secure PRE scheme, which is of deterministic re-encryption, satisfies the stronger IND-PRE-oCCA security with this modified stronger Rule-4 formulation.

Definition 4. *A single-hop bidirectional PRE scheme is said to be* $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{re}}, q_{\mathrm{d}}, \epsilon)$-IND-PRE-oCCA *secure, if for any t-time adversary $\mathcal{A}$ who asks at most $q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{re}}$ and q_{d} queries to oracles $\mathcal{O}_{\mathrm{pk}}, \mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}, \mathcal{O}_{\mathrm{re}}$ and $\mathcal{O}_{\mathrm{d}}$, respectively, we have $\mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}(k) \leq \epsilon$. The probability is taken over the choice of the random bit δ, coins of adversary $\mathcal{A}$ and the oracles.*

CPA security of (multi-hop) bidirectional PRE. For the CPA security of (multi-hop) bidirectional PRE, referred to as $\textsf{IND-PRE-CPA}$, the IND-PRE-CPA experiment (denoted $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\textsf{IND-PRE-CPA}}$) amounts to a restricted version of the above IND-PRE-oCCA experiment, where adversary $\mathcal{A}$ is only allowed to make queries to $\mathcal{O}_{\mathrm{pk}}$, $\mathcal{O}_{\mathrm{sk}}$ and $\mathcal{O}_{\mathrm{rk}}$ (but oracle access to $\mathcal{O}_{\mathrm{re}}$ and $\mathcal{O}_{\mathrm{d}}$ is denied) in both the `find` stage and the `guess` stage [14].

Non-Transitivity. One might think the following definition for NT-security would be sufficient: the proxies alone cannot generate a new valid re-encryption key between two users who has never agreed on the delegations, e.g., from the re-encryption keys $rk_{i\leftrightarrow j}$ and $rk_{j\leftrightarrow k}$, the adversary cannot produce the re-encryption key $rk_{i\leftrightarrow k}$. However, such a definition is somewhat weak, since it does not prevent an adversary from computing some other information that will allow it to re-encrypt ciphertexts from i to k (which is what we actually care about). For example, suppose a PRE scheme dose not satisfy the above NT-security (i.e., a valid re-encryption key can be generated transitively), and then we can easily convert it into another PRE scheme satisfying the above "NT-security" in the following way:[1] Let $r_{i\leftrightarrow j}$ be a re-encryption key for the original scheme. In the new scheme, we define a valid re-encryption key to be the triplet $(r_{i\leftrightarrow j}, \mathsf{Sign}(sk_i, r_{i\leftrightarrow j}), \mathsf{Sign}(sk_j, r_{i\leftrightarrow j}))$, where Sign is some secure signature algorithm. It is easy to see that the proxy will not be able to generate any new valid re-encryption keys involving user i without breaking the signature scheme. However, since the original scheme is not "NT-secure", the adversary could still compute $r_{i\leftrightarrow k}$ from $r_{i\leftrightarrow j}$ and $r_{j\leftrightarrow k}$, and thus is still able to redelegate decryption rights. Therefore, a reasonable NT-security definition should satisfy: if the proxy has the re-encryption keys $rk_{i^*\leftrightarrow j}$ and $rk_{j\leftrightarrow k}$, even in collusion with user k it shouldn't be able to decrypt ciphertexts intended for user i^*. We formally define the NT-security as below:

For an adversary $\mathcal{A}$ running in two stages `find` and `guess`, we define its advantage as

$$\mathrm{Adv}^{\mathsf{NT}}_{\mathrm{PRE},\mathcal{A}}(k) = \left|\Pr\left[\mathbf{Exp}^{\mathsf{NT}\text{-}1}_{\mathrm{PRE},\mathcal{A}}(k) = 1\right] - \Pr\left[\mathbf{Exp}^{\mathsf{NT}\text{-}0}_{\mathrm{PRE},\mathcal{A}}(k) = 1\right]\right|,$$

where $\mathbf{Exp}^{\mathsf{NT}\text{-}\delta}_{\mathrm{PRE},\mathcal{A}}$ is defined by the following experiment:

Experiment $\mathbf{Exp}^{\mathsf{NT}\text{-}\delta}_{\mathrm{PRE},\mathcal{A}}(k)$
$param \leftarrow \mathsf{Setup}(1^k)$;
$(m_0, m_1, pk_{i^*}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{re}}(\cdot,\cdot,\cdot),\mathcal{O}_{\mathrm{d}}(\cdot,\cdot)}(\mathtt{find}, param)$, where $|m_0| = |m_1|$;
$\mathtt{CT}^* \leftarrow \mathsf{Enc}(pk_{i^*}, m_\delta)$, where $\delta \xleftarrow{\$} \{0,1\}$;
$\delta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{re}}(\cdot,\cdot,\cdot),\mathcal{O}_{\mathrm{d}}(\cdot,\cdot)}(\mathtt{guess}, param, \mathtt{CT}^*)$;
return δ'.

During the above experiment, it is required that the following requirements are simultaneously satisfied:

1. For a public key pk_j whose secret key is corrupted (by oracle query to $\mathcal{O}_{\mathrm{sk}}$), $\mathcal{A}$ cannot issue $\mathcal{O}_{\mathrm{rk}}(pk_{i^*}, pk_j)$ or $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_{i^*})$.
2. pk_{i^*} is generated by oracle $\mathcal{O}_{\mathrm{pk}}$, and $\mathcal{A}$ cannot issue the secret key query $\mathcal{O}_{\mathrm{sk}}(pk_{i^*})$;
3. $\mathcal{A}$ cannot simultaneously issue $\mathcal{O}_{\mathrm{sk}}(pk_j)$ and $\mathcal{O}_{\mathrm{re}}(pk_i, pk_j, \mathtt{CT}_i)$ such that $(pk_i, \mathtt{CT}_i)$ is a derivative of $(pk_{i^*}, \mathtt{CT}^*)$.
4. $\mathcal{A}$ cannot issue $\mathcal{O}_{\mathrm{d}}(pk_z, \mathtt{CT}_z)$ such that $(pk_z, \mathtt{CT}_z)$ is a derivative of $(pk_{i^*}, \mathtt{CT}^*)$.

Definition 5. *A bidirectional PRE scheme is said to be* $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{re}}, q_{\mathrm{d}}, \epsilon)$-$\mathsf{NT}$*-secure, if for any t-time adversary $\mathcal{A}$ who asks at most $q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{re}}, q_{\mathrm{d}}$ queries to oracles $\mathcal{O}_{\mathrm{pk}}, \mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}, \mathcal{O}_{\mathrm{re}}, \mathcal{O}_{\mathrm{d}}$, respectively, we have* $\mathrm{Adv}^{\mathsf{NT}}_{\mathrm{PRE},\mathcal{A}}(k) \le \epsilon$.

Remark 3. In the above definition, it would be legal for adversary $\mathcal{A}$ to issue queries $\mathcal{O}_{\mathrm{rk}}(pk_{i^*}, pk_j)$, $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_k)$ and $\mathcal{O}_{\mathrm{sk}}(pk_k)$, only if $\mathcal{A}$ does not issue $\mathcal{O}_{\mathrm{sk}}(pk_j)$. Thus the above definition can ensure that, the proxies with re-encryption keys $rk_{i^*\leftrightarrow j}$ and $rk_{j\leftrightarrow k}$ is unable to decrypt ciphertexts intended for user i^*, even in collusion with user k.

Remark 4. At first glance, one might think that the above NT-security is still too weak in the sense that, it does not secure against the collusion of the delegatees (j and k) and the proxy (who

[1] We sincerely thank the anonymous reviewers of CRYPTO'11 for pointing out this interesting example.

holds the re-encryption key between users i^* and j). However, such an attack would make no sense, since the proxy can first re-encrypt the challenge ciphertext $\mathtt{CT}^*$ into a transformed ciphertext for user j, and then decrypt it with user j's secret key. Nevertheless, as stressed in [4, 5, 29, 30], it is still an open problem to come up with a (unidirectional or bidirectional) PRE scheme secure against *transfer of delegation attacks*, i.e., the collusion of the delegatees (j and k) and the proxy (with re-encryption keys between users i^* and j) cannot produce the re-encryption key from i^* to k.

Remark 5. The NT definition is almost identical to the $\mathsf{IND\text{-}PRE\text{-}oCCA}$ definition, with the exception in Requirement 1. It can be verified that Requirement 1 in the NT definition is strictly weaker than that in the $\mathsf{IND\text{-}PRE\text{-}oCCA}$ definition. Thus the $\mathsf{IND\text{-}PRE\text{-}oCCA}$ security is implied by the NT security (as stated in Proposition 1). The reason we separate $\mathsf{IND\text{-}PRE\text{-}oCCA}$ security from NT security is that, as we shall see, the former can normally be achieved with weaker assumption than that for achieving the latter.

Proposition 1. *For a single-hop bidirectional PRE $\mathcal{E}$, the* $\mathsf{IND\text{-}PRE\text{-}oCCA}$ *security is implied by the non-transitivity security. That is, if there exists an adversary $\mathcal{A}$ who can also break the* $\mathsf{IND\text{-}PRE\text{-}oCCA}$ *security of $\mathcal{E}$, then there also exists an adversary $\mathcal{B}$ who can also break the* NT *security of $\mathcal{E}$.*

CCA Security of transformed ciphertexts for *single-hop* bidirectional PRE. Note that, for single-hop bidirectional PRE, the original and transformed ciphertexts have different forms. The above IND-PRE-oCCA definition provides $\mathcal{A}$ with an original ciphertext in the challenge phase. An orthogonal definition of security is to capture the indistinguishability of transformed ciphertexts, in which adversary $\mathcal{A}$ is given a challenge transformed ciphertext. For single-hop schemes, since transformed ciphertexts cannot be re-encrypted, we should allow the adversary to obtain *any* re-encryption keys (*including those between the target user and the corrupted users*). Consequently, the re-encryption oracle becomes useless to $\mathcal{A}$ since $\mathcal{A}$ can re-encrypt ciphertexts itself using the corresponding re-encryption keys.

Specifically, for an adversary $\mathcal{A}$ running in two stages $\mathtt{find}$ and $\mathtt{guess}$ against a bidirectional *single-hop* PRE scheme, we define its advantage in distinguishing transformed ciphertexts as

$$\mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}(k) = \left|\Pr\left[\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA\text{-}1}}(k) = 1\right] - \Pr\left[\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA\text{-}0}}(k) = 1\right]\right|,$$

where $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA\text{-}}\delta}$ is defined by the following experiment:

Experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA\text{-}}\delta}(k)$
$param \leftarrow \mathsf{Setup}(1^k)$;
$(m_0, m_1, pk_{i^*}, pk_{i'}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{d}}(\cdot,\cdot)}(\mathtt{find}, param)$, where $|m_0| = |m_1|$;
$\mathtt{CT}' \leftarrow \mathsf{Enc}(pk_{i'}, m_\delta)$, where $\delta \xleftarrow{\$} \{0,1\}$; $\mathtt{CT}^* = \mathsf{ReEnc}(rk_{i'\leftrightarrow i^*}, \mathtt{CT}')$;
$\delta' \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot),\mathcal{O}_{\mathrm{d}}(\cdot,\cdot)}(\mathtt{guess}, param, \mathtt{CT}^*)$;
return δ'.

During the above experiment, the following requirements are required to be simultaneously satisfied:

1. $pk_{i'}$ and pk_{i^*} are generated by oracle $\mathcal{O}_{\mathrm{pk}}$, and $\mathcal{A}$ has never issued the query $\mathcal{O}_{\mathrm{sk}}(pk_{i^*})$. *But, the query $\mathcal{O}_{\mathrm{sk}}(pk_{i'})$ is allowed to $\mathcal{A}$.*
2. $\mathcal{A}$ has never issued the query $\mathcal{O}_{\mathrm{d}}(pk_{i^*}, \mathtt{CT}^*)$.

Definition 6. *A single-hop bidirectional PRE scheme is said to be $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{d}}, \epsilon)$-*$\mathsf{IND\text{-}PRE\text{-}tCCA}$ *secure, if for any t-time adversary $\mathcal{A}$ who asks at most $q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}$ and q_{d} queries to oracles $\mathcal{O}_{\mathrm{pk}}, \mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}$ and $\mathcal{O}_{\mathrm{d}}$, respectively, we have $\mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}(k) \le \epsilon$.*

Experiment $\mathbf{Exp}^{\mathsf{MSS}}_{\mathrm{PRE},\mathcal{A}}(k)$

$param \leftarrow \mathsf{Setup}(1^k)$;

$(pk_{i^*}, sk_{i^*}) \leftarrow \mathcal{A}^{\mathcal{O}_{\mathrm{pk}}(\cdot),\mathcal{O}_{\mathrm{sk}}(\cdot),\mathcal{O}_{\mathrm{rk}}(\cdot,\cdot)}(param)$;

If sk_{i^*} is a valid secret key w.r.t. pk_{i^*} then return 1,
else return 0.

Master secret security. For an adversary against the MSS security of a bidirectional PRE scheme, we define its advantage as $\mathrm{Adv}^{\mathsf{MSS}}_{\mathrm{PRE},\mathcal{A}}(k) = \Pr\left[\mathbf{Exp}^{\mathsf{MSS}}_{\mathrm{PRE},\mathcal{A}}(k) = 1\right]$, where $\mathbf{Exp}^{\mathsf{MSS}}_{\mathrm{PRE},\mathcal{A}}$ is defined by the following experiment:
During the above experiment, it is required that pk_{i^*} is generated by oracle $\mathcal{O}_{\mathrm{pk}}$, and $\mathcal{A}$ has never issued the secret key query $\mathcal{O}_{\mathrm{sk}}(pk_{i^*})$.

Definition 7. *A bidirectional PRE scheme is said to be $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, \epsilon)$-MSS secure, if for any t-time adversary $\mathcal{A}$ who asks at most $q_{\mathrm{pk}}, q_{\mathrm{sk}}$ and q_{rk} queries to oracles $\mathcal{O}_{\mathrm{pk}}, \mathcal{O}_{\mathrm{sk}}$ and $\mathcal{O}_{\mathrm{rk}}$, respectively, we have $\mathrm{Adv}^{\mathsf{MSS}}_{\mathrm{PRE},\mathcal{A}}(k) \leq \epsilon$.*

Remark 6. The MSS-security is actually a refinement of the MSS-security considered in [4, 5]. But, we would like to reminder here that, for some unusual or unnatural cases, the situation is a bit subtle. One example is, in a PRE scheme, the secret-key of each user consists of two elements (sk, sk') but the element of sk' is never used in re-encryption key generation and decryption. Another example is: the user's secret-key is defined to be sk, but it actually uses $sk' = f(sk)$ for decryption and REK generation where f is a one-way function. All these unusual or unnatural protocol examples can be trivially MSS-secure, though we are unaware of any existing PRE scheme being of such unusual structures. For the general case, roughly speaking, we say a PRE scheme is MSS-secure if the adversary is infeasible to output the secret-key elements of the delegator that are actually put into effect in REK generation and ciphertext decryption. Formal formulation of MSS-security for the general case is outside the scope of this paper, and is left for future exploration.

Due to the space limitation, we present the formulation and discussion of NT-security in Appendix D, where we in particular show that, for *single-hop* bidirectional PREs, the IND-PRE-oCCA (resp., MSS) security is implied by the NT (resp., IND-PRE-tCCA) security. The reason we separate IND-PRE-oCCA (resp., MSS) security from NT (resp., IND-PRE-tCCA) security is that, as we shall see, the former can normally be achieved with weaker assumptions than those for achieving the latter.

4 CPA-Secure Multi-Hop Bidirectional PRE with MSS-Security

Let $(\mathbb{G}, \mathbb{G}_T)$ be bilinear groups with prime order $p \geq 2^k$, where k is the security parameter. Then our multi-hop bidirectional PRE scheme is specified as below:

$\mathsf{Setup}(1^k)$: Picks two generators $g, g_1 \xleftarrow{\$} \mathbb{G}$, and output the public parameter $param = (g, g_1)$.

$\mathsf{KeyGen}(param)$: User i randomly pick $x_i \xleftarrow{\$} \mathbb{Z}_p^*$, and sets its public key as $pk_i = g^{x_i}$ and private key as $sk_i = x_i$.

$\mathsf{ReKeyGen}(sk_i, sk_j)$: On input two private keys sk_i and sk_j, this algorithm outputs the bidirectional re-encryption key $rk_{i\leftrightarrow j} = g_1^{-sk_i} g_1^{sk_j} (= g_1^{-sk_i+sk_j})$.

$\mathsf{Enc}(pk_i, m)$: To encrypt a message $m \in \mathbb{G}_T$ under the public key pk_i, the sender picks $r \xleftarrow{\$} \mathbb{Z}_p^*$, and outputs the ciphertext $\mathtt{CT}_i$ as below:

$$\mathtt{CT}_i = (C_1, C_2) = (g^r, m \cdot e(pk_i, g_1)^r).$$

$\mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$: On input a re-encryption key $rk_{i\leftrightarrow j} = g_1^{-x_i} g_1^{x_j}$, a ciphertext $\mathtt{CT}_i = (C_1, C_2)$ under public key pk_i, this algorithm computes $C_2' = C_2 \cdot e(C_1, rk_{i\leftrightarrow j}) = m \cdot e(pk_i, g_1)^r \cdot e(g^r, g_1^{-sk_i} g_1^{sk_j}) = m \cdot e(pk_i, g_1)^r \cdot e(pk_i, g_1)^{-r} \cdot e(pk_j, g_1)^r = m \cdot e(pk_j, g_1)^r$, and output $\mathtt{CT}_j = (C_1, C_2')$ as the transformed ciphertext under public key pk_j.

$\mathsf{Dec}(\mathtt{CT}_z, sk_z)$: On input a private key and ciphertext $\mathtt{CT}_z$, this algorithm outputs $m \leftarrow \frac{C_2}{e(C_1, g_1)^{sk_z}}$.

Theorem 1. *Our scheme is* IND-PRE-CPA *secure, assuming the* DBDH *assumption holds in* $(\mathbb{G}, \mathbb{G}_T)$.

Proof. Given a DBDH instance $(g, g^a, g^b, g^c, Z) \in \mathbb{G}^4 \times \mathbb{G}_T$ with unknown $a, b, c \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to decide whether $Z = e(g, g)^{abc}$. $\mathcal{B}$ works by interacting with adversary $\mathcal{A}$ in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}CPA}}$ (where $\mathcal{A}$ is only allowed to make queries to $\mathcal{O}_{\mathrm{pk}}$, $\mathcal{O}_{\mathrm{sk}}$ and $\mathcal{O}_{\mathrm{rk}}$ in the `find` and `guess` stages), as follows:

`Initialize.` $\mathcal{B}$ defines $g_1 = g^b$, and returns the public parameter $param = (g, g_1)$ to $\mathcal{A}$.

`Find Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}CPA}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\mathrm{pk}}(i)$: $\mathcal{B}$ picks $x_i \xleftarrow{\$} \mathbb{Z}_p^*$. Next, using Coron's technique [15], it flips a biased coin $c_i \in \{0, 1\}$ that yields 1 with probability θ and 0 otherwise, where θ is a fixed probability that will be determined later. If $c_i = 1$, it sets $pk_i = g^{x_i}$ (meaning that $sk_i = x_i$); else $pk_i = g^{a+x_i}$ (meaning that $sk_i = a + x_i$). Next, it adds the tuple (pk_i, x_i, c_i) to L^{list} and returns pk_i to $\mathcal{A}$.
- *Secret key oracle* $\mathcal{O}_{\mathrm{sk}}(pk_i)$: $\mathcal{B}$ first recovers (pk_i, x_i, c_i) from L^{list}. If $c_i = 1$, $\mathcal{B}$ returns $sk_i = x_i$ to $\mathcal{A}$; otherwise, it outputs a random bit and **aborts**.
- *Re-encryption key oracle* $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and responds according to the following three cases:
  - $c_i = c_j = 1$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} = g_1^{-(a+x_i)+(a+x_j)} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.
  - $c_i \neq c_j$: $\mathcal{B}$ outputs a random bit and **aborts**.

`Challenge.` When $\mathcal{A}$ decides that `find` stage is over, it outputs a public key pk_{i^*} and two equal-length messages $m_0, m_1 \in \mathbb{G}_T$ with the restrictions specified in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}CPA}}$. Algorithm $\mathcal{B}$ first recovers the tuple $(pk_{i^*}, x_{i^*}, c_{i^*})$ from L^{list}. If $c_{i^*} = 1$, $\mathcal{B}$ outputs a random bit and **aborts**. Otherwise (it means that $pk_{i^*} = g^{a+x_{i^*}}$), $\mathcal{B}$ flips a random coin $\delta \xleftarrow{\$} \{0, 1\}$, defines $C_1^* = g^c, C_2^* = Z \cdot e(g^b, g^c)^{x_{i^*}} \cdot m_\delta$, and returns $\mathtt{CT}^* = (C_1^*, C_2^*)$ as the challenge ciphertext to $\mathcal{A}$.

Observe that, if $Z = e(g, g)^{abc}$, $\mathtt{CT}^*$ is a valid challenge ciphertext. To see this, letting $r^* = c$, we have

$$
\begin{aligned}
C_1^* &= g^c = g^{r^*},\\
C_2^* &= Z \cdot e(g^b, g^c)^{x_{i^*}} \cdot m_\delta = e(g, g)^{abc} \cdot e(g^b, g^c)^{x_{i^*}} \cdot m_\delta = e(g^{a+x_{i^*}}, g^b)^c \cdot m_\delta = e(pk_{i^*}, g_1)^{r^*} \cdot m_\delta.
\end{aligned}
$$

On the other hand, if Z is uniform and independent in $\mathbb{G}_T$, the challenge ciphertext $\mathtt{CT}^*$ is independent of δ from the adversary's view.

`Guess Stage.` Adversary $\mathcal{A}$ continues to issue the rest queries with the restrictions described in experiment $\mathrm{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}CPA}}$, and $\mathcal{B}$ answers these queries just as it does in the `find` stage.

`Output.` Eventually, adversary $\mathcal{A}$ returns a guess $\delta' \in \{0, 1\}$. If $\delta' = \delta$, $\mathcal{B}$ outputs 1 to guess $Z = e(g, g)^{abc}$; otherwise, $\mathcal{B}$ outputs 0 to guess that Z is a random element in $\mathbb{G}_T$.

This completes the description of the simulation. We next analyze the simulation. It is clear that the simulation of oracle $\mathcal{O}_{\mathrm{pk}}$ is perfect. Let Abort denote the event that $\mathcal{B}$ aborts (when $Z = e(g, g)^{abc}$) during the simulation of oracles $\mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}$ or in `Challenge` stage. We have

$\Pr[\neg\mathsf{Abort}] = \theta^{q_{\text{sk}}}(\theta^2 + (1-\theta)^2)^{q_{\text{rk}}}(1-\theta) \geq \theta^{2q_{\text{rk}}+q_{\text{sk}}}(1-\theta)$. By setting $\theta = 1 - \frac{1}{1+2q_{\text{rk}}+q_{\text{sk}}}$, we get $\Pr[\neg\mathsf{Abort}] \geq \dot{e}^{-1}\frac{1}{(1+2q_{\text{rk}}+q_{\text{sk}})} = \frac{1}{\dot{e}(1+2q_{\text{rk}}+q_{\text{sk}})}$. Note that, if event Abort does not occur during the simulation, the simulations for $\mathcal{O}_{\text{sk}}, \mathcal{O}_{\text{rk}}$ and `Challenge` stage all are perfect. In this case, if $\mathcal{A}$ can break the IND-PRE-CPA security of our scheme, $\mathcal{B}$ can also resolve the DBDH instance. Specifically, $\Pr[\mathcal{B}(g, g^a, g^b, g^c, Z) = 1 | Z \neq e(g,g)^{abc}] = \frac{1}{2}$ as in this case the view of $\mathcal{A}$ is independent of δ, $\Pr[\mathcal{B}(g, g^a, g^b, g^c, Z) = 1 | Z = e(g,g)^{abc})] = \frac{1}{2} \cdot \Pr[\mathsf{Abort}] + \Pr[\delta' = \delta | \neg\mathsf{Abort}] \cdot \Pr[\neg\mathsf{Abort}] = \frac{1}{2} \cdot (1 - \Pr[\neg\mathsf{Abort})] + \Pr[\delta' = \delta | \neg\mathsf{Abort}] \cdot \Pr[\neg\mathsf{Abort}] = \frac{1}{2} + \Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta | \neg\mathsf{Abort}] - \frac{1}{2})$. Thus, $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{\mathrm{DBDH}} = |\Pr[\mathcal{B}(g, g^a, g^b, g^c, Z) = 1 | Z = e(g,g)^{abc})] - \Pr[\mathcal{B}(g, g^a, g^b, g^c, Z) = 1 | Z \neq e(g,g)^{abc}]| = |\Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta | \neg\mathsf{Abort}] - \frac{1}{2})|$. As $\epsilon = \mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}CPA}} = |2\Pr[\delta' = \delta] - 1|$ and $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{\dot{e}(1+2q_{\text{rk}}+q_{\text{sk}})}$, we have $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{\mathrm{DBDH}} \geq \frac{\epsilon}{2\dot{e}(1+2q_{\text{rk}}+q_{\text{sk}})}$. It's easy to see that the running time of $\mathcal{B}$ is bounded by $t' \leq t + O(\tau(q_{\text{pk}} + q_{\text{rk}}))$. □

Theorem 2. *Our scheme is* MSS *secure, assuming the* DL *assumption holds in group* $\mathbb{G}$.

Proof. Given a DL instance $(g, g^a) \in \mathbb{G}^2$ with unknown $a \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to output the value a. $\mathcal{B}$ works by interacting with adversary $\mathcal{A}$ in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{MSS}}$ as follows:

`Initialize.` $\mathcal{B}$ picks $\alpha_1 \xleftarrow{\$} \mathbb{Z}_p^*$ and defines $g_1 = g^{\alpha_1}$, and returns $param = (g, g_1)$ to $\mathcal{A}$.

`Query Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{MSS}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\text{pk}}(i)$: Identical to that in the proof of Theorem 1.
- *Secret key oracle* $\mathcal{O}_{\text{sk}}(pk_i)$: Identical to that in the proof of Theorem 1.
- *Re-encryption key oracle* $\mathcal{O}_{\text{rk}}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and responds according to the following three cases:
  - $c_i = c_j = 1$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} = g_1^{-(a+x_i)+(a+x_j)} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.
  - $c_i = 1 \wedge c_j = 0$: It means that $sk_i = x_i$ and $sk_j = a + x_j$. $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} g^{\alpha_1 a} = g_1^{-x_i}(g^{\alpha_1})^{a+x_j} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.
  - $c_i = 0 \wedge c_j = 1$: It means that $sk_i = a + x_i$ and $sk_j = x_j$. $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{-x_i+x_j} g^{-a\alpha_1} = g_1^{-(a+x_i)} g_1^{x_j} = g_1^{-sk_i} g_1^{sk_j}$ to $\mathcal{A}$.

`Output stage.` Finally, $\mathcal{A}$ outputs a secret key sk_{i^*} w.r.t. the public key pk_{i^*} that has not been queried to $\mathcal{O}_{\text{sk}}$. $\mathcal{B}$ recovers $(pk_{i^*}, x_{i^*}, c_{i^*})$ from L^{list}. If $c_{i^*} = 1$, $\mathcal{B}$ outputs "failure" and **aborts**. Otherwise (it means $sk_{i^*} = a + x_{i^*}$), $\mathcal{B}$ outputs $a = sk_{i^*} - x_{i^*}$ as the solution to the DL instance.

Analysis. It's easy to check that the running time of $\mathcal{B}$ is bounded by $t' \leq t + O(\tau(q_{\text{pk}} + q_{\text{rk}}))$. Let Abort denote the event of $\mathcal{B}$'s aborting during the simulation of oracle $\mathcal{O}_{\text{sk}}$ or in `Output` stage. We have $\Pr[\neg\mathsf{Abort}] = \theta^{q_{\text{sk}}}(1-\theta)$. By setting $\theta = \frac{q_{\text{sk}}}{1+q_{\text{sk}}}$, we get $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{\dot{e}(1+q_{\text{sk}})}$. Note that, if event Abort does not occur, the simulation is perfect from the view of $\mathcal{A}$. Therefore, if $\mathcal{A}$ can break the MSS security with probability ϵ, $\mathcal{B}$ can solve the DL instance with probability $\epsilon' \geq \frac{\epsilon}{\dot{e}(1+q_{\text{sk}})}$. □

5 CCA-Secure Single-Hop Bidirectional PRE with both MSS Security and NT Security

Before presenting the construction, we first highlight some subtleties and rationales in designing CCA-secure PRE schemes. It is well known that for a CCA-secure public key encryption scheme, its ciphertexts should be non-malleable [1, 6, 16, 18, 31]. However, the subtlety and difficulty of achieving CCA-secure PRE scheme lies in that a PRE scheme itself requires flexibility for converting a ciphertext under one public key into another one for another user (which is referred to as *legal malleability*). For a (single-hop) PRE scheme to be CCA-secure, it must ensure that the legal malleability *from original ciphertexts to transformed ciphertexts* is the *only* way of malleating ciphertexts. In particular, a CCA-secure (single-hop) PRE scheme should guarantee both the non-malleability of original ciphertexts and the non-malleability of transformed ciphertexts.

Let $(\mathbb{G}, \mathbb{G}_T)$ be bilinear groups with prime order $p > 2^k$, where k is the security parameter. Our CCA-secure bidirectional PRE scheme is specified by the following algorithms:

$\mathsf{Setup}(1^k)$: This setup algorithm works as follows:

1. Pick generators $g, g_0, g_1, u_1, u_2, u_3 \xleftarrow{\$} \mathbb{G}$.
2. Choose a target collision-resistant hash function $H : \mathbb{G} \times \{0,1\}^\ell \to \mathbb{Z}_p^*$. Choose also a pseudo-random function (PRF) family $F : \mathbb{G}_T \times \mathbb{G} \to \{0,1\}^{\ell-\ell_1} \| \{0,1\}^{\ell_1}$ such that, given a seed in $\mathbb{G}_T$ and an input in $\mathbb{G}$, it outputs an ℓ-bit pseudorandom string. Here "$||$" denotes the operator of string concatenation, and ℓ and ℓ_1 are all polynomial in k.
3. Output the public parameter $param = (p, \mathbb{G}, \mathbb{G}_T, g, g_0, g_1, u_1, u_2, u_3, H, F, \ell_1, \ell)$.

$\mathsf{KeyGen}(param)$: User i picks $x_i \xleftarrow{\$} \mathbb{Z}_p^*$, and sets its public key as $pk_i = g^{x_i}$ and private key as $sk_i = x_i$.

$\mathsf{ReKeyGen}(sk_i, sk_j)$: On input two private keys sk_i and sk_j, this algorithm outputs the bidirectional re-encryption key $rk_{i\leftrightarrow j} = g_1^{sk_i sk_j}$.

Note that, from $rk_{i\leftrightarrow j} = g_1^{sk_i sk_j}$ and sk_i no efficient algorithm can compute sk_j, which ensures the MSS security. Also, given two REKs $rk_{i\leftrightarrow j} = g_1^{sk_i sk_j}$ and $rk_{j\leftrightarrow k} = g_1^{sk_j sk_k}$ and even also the secret key sk_k, no efficient algorithm can decrypts (original) ciphertexts generated under pk_i, which implies the infeasibility of computing $rk_{i\leftrightarrow k} = g_1^{sk_i sk_k}$ and thus ensures the NT security. Formal analysis of the MSS-security and NT-security is referred to Appendix E.

$\mathsf{Enc}(pk_i, m)$: To encrypt a message $m \in \{0,1\}^{\ell_1}$ under pk_i, the sender proceeds as follows:

1. Pick $r \xleftarrow{\$} \mathbb{Z}_p^*$, and set $C_0 = g_0^r$ and $C_1 = g^r$.
2. Compute $K = e(pk_i, g_1)^r$ and set $C_2 = [F(K, C_0)]_{\ell-\ell_1} \| \big([F(K, C_0)]^{\ell_1} \oplus m\big)$, where $[F(K, C_0)]_{\ell-\ell_1}$ (resp., $[F(K, C_0)]^{\ell_1}$) denotes the $(\ell-\ell_1)$-bit prefix (resp., the ℓ_1-bit suffix) of $[F(K, C_0)]$.
3. Pick $t \xleftarrow{\$} \mathbb{Z}_p^*$, and compute $h = H(C_0, C_2)$ and $C_3 = \big(u_1^h u_2^t u_3\big)^r$.
4. Output the ciphertext $\mathtt{CT}_i = (t, C_0, C_1, C_2, C_3)$.

Here, the use of PRF, rather than a generic authentication mechanism (e.g., encrypt-then-MAC), is to get shorter ciphertext and to have more concise security proof. Note that the value t can be viewed to serve as the randomness of a chameleon hash function in an application of the Canetti-Halevi-Katz and Boyen-Mei-Waters techniques [11, 13]. This use of a chameleon hash function with randomness enables our scheme to achieve the adaptive security, i.e., the adversary needs not to commit ahead of time which public key she wants to challenge. We note that the use of a chameleon hash function with randomness was also previously appeared in [2, 23, 27, 32].

$\mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$: On input an REK $rk_{i\leftrightarrow j} = g_1^{x_i x_j}$, an original ciphertext $\mathtt{CT}_i$ and another public key pk_j, this algorithm re-encrypts this ciphertext intended for public key pk_j as follows:

1. Compute $h = H(C_0, C_2)$, and then check the validity of $\mathtt{CT}_i$ by testing whether the following equalities hold:

$$e(C_0, u_1^h u_2^t u_3) = e(C_3, g_0), \tag{1}$$

$$e(C_0, g) = e(C_1, g_0). \tag{2}$$

Observe that Eq. (1) ensures that $C_0 = g_0^r$ and $C_3 = \big(u_1^h u_2^t u_3\big)^r$, and Eq. (2) ensures that $C_0 = g_0^r$ and $C_1 = g^r$. Note that, using the technique as in [12, 17, 25, 26], the efficiency of the above verifications can be further improved by reducing two pairings (at the cost of two more multi-exponentiations). Concretely, the verifications can be alternately done by picking $r_1, r_2 \xleftarrow{\$} \mathbb{Z}_p^*$ and testing whether

$$e\left(C_0, g^{r_1}\left(u_1^h u_2^t u_3\right)^{r_2}\right) = e\left(C_1^{r_1} C_3^{r_2}, g_0\right). \tag{3}$$

2. If the above verifications fail, output $\perp$ indicating an invalid ciphertext. Otherwise, compute $C_1' = e(C_1, rk_{i\leftrightarrow j})$, and output the transformed ciphertext under public key pk_j as $\mathtt{CT}_j = (t, C_0, C_1', C_2, C_3)$. Observe that $\mathtt{CT}_j$ is of the following forms:

$\mathtt{CT}_j = (t, C_0, C_1', C_2, C_3) = \left(t, g_0^r, K^{sk_j}, [F(K, C_0)]_{\ell-\ell_1} \| \left([F(K, C_0)]^{\ell_1} \oplus m\right), \left(u_1^h u_2^t u_3\right)^r\right)$,

where $K = e(pk_i, g_1)^r$.

$\mathsf{Dec}(sk_z, \mathtt{CT}_z)$: On input $(sk_z, \mathtt{CT}_z)$, this algorithm works according to two cases:

- $\mathtt{CT}_z$ is an original ciphertext $\mathtt{CT}_z = (t, C_0, C_1, C_2, C_3)$ (where $C_1 \in \mathbb{G}$):
  1. First check the validity of the ciphertext as in Eq. (3). If the verification fails, output "$\perp$" indicating an invalid ciphertext.
  2. Compute $K = e(C_1, g_1)^{sk_z}$, and output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$, where $[C_2]^{\ell_1}$ denotes the ℓ_1-bit suffix of C_2.
- $\mathtt{CT}_z$ is a transformed ciphertext $\mathtt{CT}_z = (t, C_0, C_1', C_2, C_3)$ (where $C_1' \in \mathbb{G}_T$):
  1. First check the validity of the ciphertext as in Eq. (1). If the verification fails, output "$\perp$" indicating an invalid ciphertext.
  2. Compute $K = C_1'^{1/sk_z}$. Output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$ if the following equality holds:

$$[F(K, C_0)]_{\ell-\ell_1} = [C_2]_{\ell-\ell_1}. \tag{4}$$

     Otherwise, output "$\perp$" indicating an invalid ciphertext.

Remark 7. In the security proofs, to answer adversary $\mathcal{A}$'s decryption queries on transformed ciphertexts, the simulator needs to know the original public key pk_i. To deal with this problem, we can slightly modify the above scheme by including public key pk_i into both original ciphertext and transformed ciphertext, and computing $h = H(pk_i, C_0, C_2)$ instead of $h = H(C_0, C_2)$ (the description of H should accordingly be $H : \mathbb{G}^2 \times \{0,1\}^\ell \to \mathbb{Z}_p^*$). The security analysis, as well as the comparison with CH07, is w.r.t. this modified version.

Remark 8. At a high level, the equalities (1) and (2) (resp., (1) and (4)) ensure the well-formedness of original (resp., transformed) ciphertexts, and thus the above PRE construction fits the aforementioned design rationales of non-malleability for achieving CCA-secure PRE schemes.

Remark 9. In algorithm $\mathsf{ReKeyGen}$, users i and j make equal contributions to the REK key $rk_{i\leftrightarrow j} = g_1^{sk_i sk_j}$. Such an advantageous feature enables the proxy to transform ciphertexts without distinguishing the re-encryption direction, since its actions for both directions are the same. To our knowledge, this is the first bidirectional PRE scheme that enjoys this advantageous feature.

5.1 Security Analysis

Theorem 3. *Our scheme is* $\mathsf{IND\text{-}PRE\text{-}oCCA}$ *secure, assuming the hash function H is target collision resistant, F is a PRF family and the* 1-AwDBDHI *assumption holds in groups* $(\mathbb{G}, \mathbb{G}_T)$.

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{re}}, q_{\mathrm{d}}, \epsilon)$-$\mathsf{IND\text{-}PRE\text{-}oCCA}$ security of our scheme, then we show how to construct another algorithm $\mathcal{B}$ that can break the 1-AwDBDHI assumption in $(\mathbb{G}, \mathbb{G}_T)$. Given a 1-AwDBDHI instance $(g, g^{\frac{1}{a^2}}, g^a, g^b, Z) \in \mathbb{G}^4 \times \mathbb{G}_T$ with unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to decide whether $Z = e(g, g)^{b/a}$. $\mathcal{B}$ works by interacting with adversary $\mathcal{A}$ in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}oCCA}}$ as follows:

`Initialize.` $\mathcal{B}$ provides $\mathcal{A}$ with public parameter including $g_0 = g^{\gamma_0}, g_1 = \left(g^{\frac{1}{a^2}}\right)^{\gamma_1}, u_1 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_1} g^{\beta_1}, u_2 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_2} g^{\beta_2}$ and $u_3 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_3} g^{\beta_3}$ for random $\gamma_0, \gamma_1, \alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3 \xleftarrow{\$} \mathbb{Z}_p^*$.

`Find Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}_{\text{PRE},\mathcal{A}}^{\text{IND-PRE-oCCA}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\text{pk}}(i)$: $\mathcal{B}$ picks $x_i \xleftarrow{\$} \mathbb{Z}_p^*$. Using the Coron's technique [15], it flips a biased coin $c_i \in \{0, 1\}$ that yields 1 with probability θ and 0 otherwise, where θ is a fixed probability to be determined later. If $c_i = 1$, it sets $pk_i = g^{x_i}$ (meaning that $sk_i = x_i$); else $pk_i = (g^a)^{x_i}$ (meaning that $sk_i = ax_i$). Next, it adds (pk_i, x_i, c_i) to L^{list} and returns pk_i to $\mathcal{A}$.
- *Secret key oracle* $\mathcal{O}_{\text{sk}}(pk_i)$: $\mathcal{B}$ first recovers (pk_i, x_i, c_i) from L^{list}. If $c_i = 1$, $\mathcal{B}$ returns $sk_i = x_i$ to $\mathcal{A}$; otherwise, it outputs a random bit and **aborts**.
- *Re-encryption key oracle* $\mathcal{O}_{\text{rk}}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and responds according to the following three cases:
  - $c_i = c_j = 1$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g^{\gamma_1 x_i x_j} = \left(\left(g^{\frac{1}{a^2}}\right)^{\gamma_1}\right)^{ax_i ax_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i \neq c_j$: $\mathcal{B}$ outputs a random bit and **aborts**.
- *Re-encryption oracle* $\mathcal{O}_{\text{re}}(pk_i, pk_j, \mathtt{CT}_i)$: $\mathcal{B}$ first parses $\mathtt{CT}_i$ as $(pk_i, t, C_0, C_1, C_2, C_3)$, computes $h = H(pk_i, C_0, C_2)$, and then checks the validity of the ciphertext as in Eq. (3). If the verification fails, it returns "$\perp$" to $\mathcal{A}$ (indicating an invalid ciphertext). Otherwise, $\mathcal{B}$ recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and works according to the following two cases:
  - $c_i = c_j$: $\mathcal{B}$ first generates the re-encryption key $rk_{i\leftrightarrow j}$ as in the response for the re-encryption key oracle $\mathcal{O}_{\text{rk}}$, and returns $\mathtt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \mathtt{CT}_i)$ to $\mathcal{A}$.
  - $c_i \neq c_j$: Without loss of generality, suppose $sk_i = ax_i$ and $sk_j = x_j$. $\mathcal{B}$ first checks whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0$ or not, where $h = H(pk_i, C_0, C_2)$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0$ then $\mathcal{B}$ outputs a random bit and **aborts**. Otherwise, from $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left(\left(g^{\frac{1}{a^2}}\right)^{\alpha_1 h + \alpha_2 t + \alpha_3} g^{\beta_1 h + \beta_2 t + \beta_3}\right)^r$, $\mathcal{B}$ can compute
$$g_1^r = \left(g^{\frac{\gamma_1}{a^2}}\right)^r = \left(\frac{C_3}{C_0^{\frac{\beta_1 h + \beta_2 t + \beta_3}{\gamma_0}}}\right)^{\frac{\gamma_1}{\alpha_1 h + \alpha_2 t + \alpha_3}}. \tag{5}$$
    Then $\mathcal{B}$ computes $C_1' = e\left(g^a, g_1^r\right)^{x_i x_j}$, and returns $\mathtt{CT}_j = (pk_i, t, C_0, C_1', C_2, C_3)$ to $\mathcal{A}$. Observe that $\mathtt{CT}_j$ is indeed a valid transformed ciphertext as required, since $c_i \neq c_j$ means $sk_i sk_j = ax_i x_j$, and hence we have $C_1' = e\left(g^a, g_1^r\right)^{x_i x_j} = e\left(g^r, g_1^{ax_i x_j}\right) = e(g^r, g_1^{sk_i sk_j}) = e(C_1, rk_{i\leftrightarrow j})$.

  Note that, in the public parameters $u_1 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_1} g^{\beta_1}$, $u_2 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_2} g^{\beta_2}$ and $u_3 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_3} g^{\beta_3}$, α_1 (α_2, α_3, resp.) is perfectly blinded by β_1 (β_2, β_3, resp.), and hence no information about α_1, α_2 and α_3 is leaked to the adversary. So, in Eq. (5), the equality $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$ information-theoretically holds with probability $1/p$.
- *Decryption oracle* $\mathcal{O}_{\text{d}}(pk_z, \mathtt{CT}_z)$: $\mathcal{B}$ first recovers tuple (pk_z, x_z, c_z) from L^{list}. If $c_z = 1$ (it means that $sk_z = x_z$), algorithm $\mathcal{B}$ returns the result of $\mathsf{Dec}(x_z, \mathtt{CT}_z)$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ proceeds according to the following two cases:
  - $\mathtt{CT}_z = (pk_z, t, C_0, C_1, C_2, C_3)$ is an original ciphertext: Compute $h = H(pk_z, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (3). If the verification fails, output "$\perp$" indicating an invalid ciphertext. Then, check whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$, output a random bit and **abort**; Otherwise, from $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left(\left(g^{\frac{1}{a^2}}\right)^{\alpha_1 h + \alpha_2 t + \alpha_3} g^{\beta_1 h + \beta_2 t + \beta_3}\right)^r$, obtain g_1^r as in Eq. (5). Then, compute $K = e(pk_z, g_1^r)$ and output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$.

– $\mathtt{CT}_z = (pk_i, t, C_0, C_1', C_2, C_3)$ is a transformed ciphertext: Compute $h = H(pk_i, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (1). If the verification fails, output "$\perp$" indicating an invalid ciphertext. Otherwise, check whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$, output a random bit and **abort**; Otherwise, continue to execute the following steps:

1. From $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left((g^{\frac{1}{a^2}})^{\alpha_1 h+\alpha_2 t+\alpha_3} g^{\beta_1 h+\beta_2 t+\beta_3}\right)^r$, obtain g_1^r as in Eq. (5).
2. Recover tuple (pk_i, x_i, c_i) from L^{list}. If $c_i = 1$ check whether $C_1' = e(pk_z^{x_i}, g_1^r)$ holds; while if $c_i = 0$ check whether $C_1' = e(g^{x_i x_z \gamma_1}, C_0^{\frac{1}{\gamma_0}})$ holds. If no, output "$\perp$" indicating an invalid ciphertext.
3. Compute $K = e(pk_i, g_1^r)$ and check whether $[F(K, C_0)]_{\ell-\ell_1} = [C_2]_{\ell-\ell_1}$. If not, output "$\perp$"; otherwise, output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$.

Note that, similar to the analysis in oracle $\mathcal{O}_{\text{re}}$, $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$ holds with probability $\frac{1}{p}$.

`Challenge.` When $\mathcal{A}$ decides that `find` stage is over, it outputs a public key pk_{i^*} and two equal-length messages $m_0, m_1 \in \{0,1\}^{\ell_1}$ with the restrictions specified in experiment $\mathbf{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}$. Algorithm $\mathcal{B}$ responds as follows:

1. Recover tuple $(pk_{i^*}, x_{i^*}, c_{i^*})$ from L^{list}. If $c_{i^*} = 1$, $\mathcal{B}$ outputs a random bit in $\{0,1\}$ and **aborts**. Otherwise, it means that $pk_{i^*} = (g^a)^{x_{i^*}}$, and $\mathcal{B}$ proceeds to execute the rest steps.
2. Define $C_0^* = (g^b)^{\gamma_0}, C_1^* = g^b, K^* = Z^{x_{i^*}\gamma_1}, C_2^* = [F(K^*, C_1^*)]_{\ell-\ell_1} \| ([F(K^*, C_1^*)]^{\ell_1} \oplus m_\delta)$ for a random bit δ, $t^* = -\frac{\alpha_1 h^* + \alpha_3}{\alpha_2}$ and $C_3^* = (g^b)^{\beta_1 h^* + \beta_2 t^* + \beta_3}$ where $h^* = H(pk_{i^*}, C_0^*, C_2^*)$.
3. Return $\mathtt{CT}^* = (pk_{i^*}, t^*, C_0^*, C_1^*, C_2^*, C_3^*)$ as the challenge ciphertext to adversary $\mathcal{A}$.

Observe that if $Z = e(g,g)^{b/a}$, the ciphertext $\mathtt{CT}^*$ is indeed a valid challenge ciphertext. To see this, letting $r^* = b$, we have

$$\begin{aligned}
C_0^* &= (g^b)^{\gamma_0} = (g^{\gamma_0})^b = g_0^{r^*}, \qquad C_1^* = g^b = g^{r^*}, \\
K^* &= Z^{x_{i^*}\gamma_1} = e(g,g)^{\frac{b x_{i^*}\gamma_1}{a}} = e\big(g^{a x_{i^*}}, g^{\frac{\gamma_1}{a^2}}\big)^b = e(pk_{i^*}, g_1)^{r^*}, \\
C_3^* &= (g^b)^{\beta_1 h^* + \beta_2 t^* + \beta_3} = \big(g^{\beta_1 h^* + \beta_2 t^* + \beta_3}\big)^b = \left((g^{\frac{1}{a^2}})^{\alpha_1 h^*} (g^{\frac{1}{a^2}})^{-\frac{\alpha_1 h^* + \alpha_3}{\alpha_2}\cdot\alpha_2} (g^{\frac{1}{a^2}})^{\alpha_3} \cdot g^{\beta_1 h^* + \beta_2 t^* + \beta_3}\right)^{r^*} \\
&= \left(\left((g^{\frac{1}{a^2}})^{\alpha_1} g^{\beta_1}\right)^{h^*} \cdot \left((g^{\frac{1}{a^2}})^{\alpha_2} g^{\beta_2}\right)^{t^*} \cdot \left((g^{\frac{1}{a^2}})^{\alpha_3} g^{\beta_3}\right)\right)^{r^*} = \left(u_1^{h^*} u_2^{t^*} u_3\right)^{r^*}.
\end{aligned}$$

On the other hand, if Z is distributed uniformly and independently over $\mathbb{G}_T$, so is K^*, and m_δ is blinded by the pseudorandom value $[F(K^*, C_1^*)]$.

`Guess Stage.` Adversary $\mathcal{A}$ continues to issue the rest queries. In this stage, whenever $\mathcal{B}$ finds a collision of $h^* = H(pk_{i^*}, C_0^*, C_2^*)$, $\mathcal{B}$ outputs a random bit and aborts, which is referred to as **collision abort** for presentation simplicity. Otherwise, $\mathcal{B}$ can respond these queries for $\mathcal{A}$ as in the `find stage` (recall that $\mathcal{A}$ has to follow the restrictions described in experiment $\text{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}$). We highlight three key observations on $\mathcal{B}$'s simulation of the guess stage:

- The reason for **collision abort** is: Provided $\mathcal{A}$ can find collisions of h^*, it can distinguish the simulation of $\mathcal{B}$ and its real interactions with the oracles in experiment $\text{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}$. For example, from $\mathtt{CT}^* = (pk_{i^*}, t^*, C_0^*, C_1^*, C_2^*, C_3^*)$, $\mathcal{A}$ sets $C_2^{*\prime} = C_2^* \oplus (0^{\ell-\ell_1} \| m_0)$, if $h^* = H(pk_{i^*}, C_0^*, C_2^*) = H(pk_{i^*}, C_0^*, C_2^{*\prime})$ then $\mathtt{CT}^{*\prime} = (pk_{i^*}, t^*, C_0^*, C_1^*, C_2^{*\prime}, C_3^*)$ is a valid ciphertext and $\mathcal{A}$ can get the plaintext by querying $\mathtt{CT}^{*\prime}$ to the decryption oracle $\mathcal{O}_{\text{d}}$. Note that for real interactions in experiment $\mathbf{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}$, the plaintext got by $\mathcal{A}$ from $\mathcal{O}_{\text{d}}(pk_{i^*}, \mathtt{CT}^{*\prime})$ will be either $m_0 \oplus m_0 = 0$ (in case $\mathtt{CT}^*$ encrypts m_0) or $m_1 \oplus m_0$ (in case $\mathtt{CT}^*$ encrypts m_1). But, for the simulation of $\mathcal{B}$ *when Z is a random value in $\mathbb{G}_T$*, $\mathcal{A}$ will get back an arbitrary value with $\mathcal{O}_{\text{d}}(pk_{i^*}, \mathtt{CT}^{*\prime})$.

- Although $\mathtt{CT}^*$ leaks the information $\alpha_1 h^* + \alpha_2 t^* + \alpha_3 = 0$ to $\mathcal{A}$, the fact that $\alpha_1 h^* + \alpha_2 t^* + \alpha_3 = 0$ still information-theoretically hides the values of $\alpha_1, \alpha_2, \alpha_3$. Thus, the equality that for some h, t, $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$ (that will cause $\mathcal{B}$ to abort in the simulation of $\mathcal{O}_{\mathrm{re}}, \mathcal{O}_{\mathrm{d}}$) in the `guess` stage still information-theoretically holds with probability $\frac{1}{p}$.
- If $\mathcal{A}$ queries $\mathcal{O}_{\mathrm{rk}}(pk_{i^*}, pk_j)$ (or $\mathcal{O}_{\mathrm{rk}}(pk_j, pk_{i^*})$) and $\mathcal{O}_{\mathrm{d}}(pk_j, \mathtt{CT}_j)$ such that $\mathtt{CT}_j = (pk_{i^*}, t, C_0, C_1', C_2, C_3)$ is a re-encrypted ciphertext, $\mathcal{B}$ can check whether $\mathtt{CT}_j = \mathsf{ReEnc}(rk_{i^* \leftrightarrow j}, \mathtt{CT}^*)$ by first recovering tuple (pk_j, x_j, c_j) (note that it must be $c_j = 0$) and testing whether $t = t^*, C_0 = C_0^*, C_2 = C_2^*, C_3 = C_3^*$ and $C_1' = e(g^{x_{i^*} x_j \gamma_1}, C_0^{*\frac{1}{\gamma_0}})$ are simultaneously hold. This is why our scheme satisfies the modified stronger Rule-4 as mentioned in Remark 2.

`Output.` Eventually, adversary $\mathcal{A}$ returns a guess $\delta' \in \{0, 1\}$. If $\delta' = \delta$, $\mathcal{B}$ outputs 1 to guess $Z = e(g, g)^{b/a}$; otherwise, $\mathcal{B}$ outputs 0 to guess that Z is a random element in $\mathbb{G}_T$.

This completes the description of the simulation. Let Abort denote the event that $\mathcal{B}$ aborts (when $Z = e(g, g)^{b/a}$) during the simulation of oracles $\mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}, \mathcal{O}_{\mathrm{re}}, \mathcal{O}_{\mathrm{d}}$ or in `Challenge` stage or due to **collision abort** in `Guess` stage. We have $\Pr[\neg\mathsf{Abort}] \geq \theta^{q_{\mathrm{sk}}}(\theta^2 + (1-\theta)^2)^{q_{\mathrm{rk}}}(1 - \frac{1}{p})^{q_{\mathrm{re}} + q_{\mathrm{d}}}(1 - \theta)(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}}) \geq \theta^{2q_{\mathrm{rk}} + q_{\mathrm{sk}}}(1 - \theta)(1 - \frac{1}{p})^{q_{\mathrm{re}} + q_{\mathrm{d}}}(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$. By setting $\theta = 1 - \frac{1}{1 + 2q_{\mathrm{rk}} + q_{\mathrm{sk}}}$, we have $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{\dot{e}(1 + 2q_{\mathrm{rk}} + q_{\mathrm{sk}})}(1 - \frac{q_{\mathrm{re}} + q_{\mathrm{d}}}{p})(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$.

Observe that, if event Abort does not occur during the simulation and $Z = e(g, g)^{b/a}$, then the simulation of $\mathcal{B}$ is perfect from the view of $\mathcal{A}$. In this case, $\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^b, Z) = 1 | Z = e(g, g)^{b/a}] = \frac{1}{2}\Pr[\mathsf{Abort}] + \Pr[\delta' = \delta | \neg\mathsf{Abort}]\Pr[\neg\mathsf{Abort}] = \frac{1}{2} + \Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta | \neg\mathsf{Abort}] - \frac{1}{2})$. On the other hand, if event **Abort** does not occur during the simulation and Z is distributed uniformly and independently over $\mathbb{G}_T$, the only advantage that A can get is to break the pseudorandomness of the PRF F. In the later case, by standard probabilistic trick [6, 20] we get $\Pr[\delta' = \delta | \mathcal{B} \text{ does not abort when } Z \neq e(g, g)^{b/a}] = \frac{1}{2} \pm \frac{1}{2}\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}}$. Defining $\epsilon_{prf} = \pm\frac{1}{2}\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}} \Pr[\mathcal{B} \text{ does not abort when } Z \neq e(g, g)^{b/a}]$ (that is a negligible quantity assuming the underlying PRF is secure), we have $\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^b, Z) = 1 | Z \neq e(g, g)^{b/a}] = \frac{1}{2} + \epsilon_{prf}$. Thus, $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{1\text{-AwDBDHI}} = |\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^b, Z) = 1 | Z = e(g, g)^{b/a})] - \Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^b, Z) = 1 | Z \neq e(g, g)^{b/a}]| = |\Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta | \neg\mathsf{Abort}] - \frac{1}{2}) - \epsilon_{prf}|$. As $\epsilon = \mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}oCCA}} = |2\Pr[\delta' = \delta] - 1|$ and $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{\dot{e}(1 + 2q_{\mathrm{rk}} + q_{\mathrm{sk}})}(1 - \frac{q_{\mathrm{re}} + q_{\mathrm{d}}}{p})(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$, we have $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{1\text{-AwDBDHI}} \geq \frac{|\pm\frac{1}{2}\epsilon - \epsilon_{prf}|}{\dot{e}(1 + 2q_{\mathrm{rk}} + q_{\mathrm{sk}})}(1 - \frac{q_{\mathrm{re}} + q_{\mathrm{d}}}{p})(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$. Note that, under the assumption that the underlying H and F are secure, if ϵ is non-negligible in k, so is ϵ'. It's also easy to check that, by straightforward computation, the running time of $\mathcal{B}$ is bounded by $t' \leq t + O(\tau(q_{\mathrm{pk}} + q_{\mathrm{rk}} + 8q_{\mathrm{re}} + 8q_{\mathrm{d}}))$. □

Theorem 4. *Our scheme is* IND-PRE-tCCA *secure, assuming the hash function H is target collision resistant, F is a PRF family and the* 2-wDBDHI *assumption holds in groups* $(\mathbb{G}, \mathbb{G}_T)$.

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, q_{\mathrm{d}}, \epsilon)$-IND-PRE-tCCA security, then we show how to construct another algorithm $\mathcal{B}$ that can break the 2-wDBDHI assumption in $(\mathbb{G}, \mathbb{G}_T)$. Given a 2-wDBDHI instance $(g, g^a, g^{a^2}, g^b, Z) \in \mathbb{G}^4 \times \mathbb{G}_T$ with unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to decide whether $Z = e(g, g)^{b/a}$ by interacting with $\mathcal{A}$. Before starting the simulation, $\mathcal{B}$ first tosses a biased coin $\mathcal{COIN} \in \{0, 1\}$ that yields 0 with θ and 1 otherwise, where θ is a fixed probability to be determined later. Then, $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

`Initialize.` $\mathcal{B}$ first randomly picks $\gamma_0, \gamma_1, \alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3 \xleftarrow{\$} \mathbb{Z}_p^*$, and provides $\mathcal{A}$ with public parameters including

$$\begin{cases} g_0 = \left(g^{a^2}\right)^{\gamma_0}, g_1 = g^{\gamma_1}, u_1 = g^{\alpha_1}\left(g^{a^2}\right)^{\beta_1}, u_2 = g^{\alpha_2}\left(g^{a^2}\right)^{\beta_2}, u_3 = g^{\alpha_3}\left(g^{a^2}\right)^{\beta_3}, & \text{If } \mathcal{COIN} = 0; \\ g_0 = \left(g^{a}\right)^{\gamma_0}, g_1 = g^{\gamma_1}, u_1 = g^{\alpha_1}\left(g^{a}\right)^{\beta_1}, u_2 = g^{\alpha_2}\left(g^{a}\right)^{\beta_2}, u_3 = g^{\alpha_3}\left(g^{a}\right)^{\beta_3}, & \text{If } \mathcal{COIN} = 1. \end{cases}$$

`Find Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\mathrm{pk}}(i)$: Identical to that in the proof of Theorem 3.

- *Secret key oracle* $\mathcal{O}_{\rm sk}(pk_i)$: Identical to that in the proof of Theorem 3.
- *Re-encryption key oracle* $\mathcal{O}_{\rm rk}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from $L^{\rm list}$, and responds according to the following three cases:
  - $c_i = c_j = 1$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = g_1^{x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = \left(g^{a^2}\right)^{\gamma_1 x_i x_j} = (g^{\gamma_1})^{ax_i ax_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i \neq c_j$: $\mathcal{B}$ returns $rk_{i\leftrightarrow j} = (g^a)^{\gamma_1 x_i x_j} = (g^{\gamma_1})^{ax_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
- *Decryption oracle* $\mathcal{O}_{\rm d}(pk_z, \mathtt{CT}_z)$: $\mathcal{B}$ first recovers tuple (pk_z, x_z, c_z) from $L^{\rm list}$. If $c_z = 1$ (it means that $sk_z = x_z$), algorithm $\mathcal{B}$ returns the result of $\mathsf{Dec}(x_z, \mathtt{CT}_z)$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ proceeds to execute the following steps:
  1. If $\mathtt{CT}_z = (pk_z, t, C_0, C_1, C_2, C_3)$ is an original ciphertext, compute $h = H(pk_z, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (3). If $\mathtt{CT}_z = (pk_i, t, C_0, C_1', C_2, C_3)$ is a transformed ciphertext, compute $h = H(pk_i, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (1).
  2. If the verification fails, output "$\perp$" indicating an invalid ciphertext. Otherwise, we have
  $$\begin{cases} C_0 = g_0^r = \left(g^{a^2}\right)^{r\cdot\gamma_0}, C_3 = (u_1^h u_2^t u_3)^r = \left(g^{\alpha_1 h+\alpha_2 t+\alpha_3}\left(g^{a^2}\right)^{\beta_1 h+\beta_2 t+\beta_3}\right)^r, & \text{If } \mathcal{COIN} = 0; \\ C_0 = g_0^r = \left(g^{a}\right)^{r\cdot\gamma_0}, C_3 = (u_1^h u_2^t u_3)^r = \left(g^{\alpha_1 h+\alpha_2 t+\alpha_3}\left(g^{a}\right)^{\beta_1 h+\beta_2 t+\beta_3}\right)^r, & \text{If } \mathcal{COIN} = 1. \end{cases}$$
  3. Check whether $\alpha_1 h+\alpha_2 t+\alpha_3 = 0 \mod p$. If $\alpha_1 h+\alpha_2 t+\alpha_3 = 0 \mod p$, output a random bit and **abort**; Otherwise, obtain g_1^r by computing $g_1^r = \left(\frac{C_3}{C_0^{\frac{\beta_1 h+\beta_2 t+\beta_3}{\gamma_0}}}\right)^{\frac{\gamma_1}{\alpha_1 h+\alpha_2 t+\alpha_3}}$.
  4. If $\mathtt{CT}_z$ is an original ciphertext, compute $K = e(pk_i, g_1^r)$. If $\mathtt{CT}_z$ is a transformed ciphertext, recover tuple (pk_i, x_i, c_i) from $L^{\rm list}$, and check the validity of ciphertext component C_1' as follows: If $c_i = 1$ check whether $C_1' = e(g^{ax_i x_z}, g_1^r)$ holds; while if $c_i = 0$ check whether $C_1' = e(g^{a^2 x_i x_z}, g_1^r)$ holds. If no, output "$\perp$" indicating an invalid ciphertext, otherwise, compute $K = e(pk_j, g_1^r)$.
  5. Check whether $[F(K, C_0)]_{\ell-\ell_1} = [C_2]_{\ell-\ell_1}$. If not, output "$\perp$" indicating an invalid ciphertext; otherwise, output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$.

`Challenge.` When $\mathcal{A}$ decides that `find` stage is over, it outputs $(m_0, m_1, pk_{i^*}, pk_{i'})$ with the restrictions specified in experiment $\mathbf{Exp}^{\textsf{IND-PRE-tCCA}}_{\rm PRE,\mathcal{A}}$. Algorithm $\mathcal{B}$ responds as follows:

1. Recover tuples $(pk_{i^*}, x_{i^*}, c_{i^*})$ from $L^{\rm list}$. If $c_{i^*} = 1$, $\mathcal{B}$ outputs a random bit and **aborts**. Otherwise (it means that $pk_{i^*} = (g^a)^{x_{i^*}}$, $sk_{i^*} = ax_{i^*}$), $\mathcal{B}$ proceeds to execute the rest steps.
2. Recover tuple $(pk_{i'}, x_{i'}, c_{i'})$ from $L^{\rm list}$. If $c_{i'} \neq \mathcal{COIN}$, $\mathcal{B}$ outputs a random bit and **aborts** since it means $\mathcal{B}$ guessed the wrong $\mathcal{COIN}$. Otherwise, $\mathcal{B}$ proceeds to execute the rest steps.
3. Define $C_0^* = \left(g^b\right)^{\gamma_0}, C_1'^* = e(g, g^b)^{x_{i'} x_{i^*}\gamma_1}, K^* = Z^{x_{i'}\gamma_1}, C_2^* = [F(K^*, C_1^*)]_{\ell-\ell_1} \| ([F(K^*, C_1^*)]^{\ell_1} \oplus m_\delta)$ for a random bit δ, $t^* = -\frac{\alpha_1 h^* + \alpha_3}{\alpha_2}$ and $C_3^* = \left(g^b\right)^{\beta_1 h^* + \beta_2 t^* + \beta_3}$ where $h^* = H(pk_{i'}, C_0^*, C_2^*)$.
4. Return $\mathtt{CT}^* = (pk_{i'}, t^*, C_0^*, C_1'^*, C_2^*, C_3^*)$ as the challenge ciphertext to adversary $\mathcal{A}$.

Observe that if $Z = e(g, g)^{b/a}$, the ciphertext $\mathtt{CT}^*$ is indeed a valid challenge ciphertext (with the same distribution of transformed ciphertext from user i' as specified in experiment $\mathbf{Exp}^{\textsf{IND-PRE-tCCA}}_{\rm PRE,\mathcal{A}}$), since

- if $\mathcal{COIN} = 0$, letting $r^* = \frac{b}{a^2}$, we have

$$
\begin{aligned}
C_0^* &= \left(g^b\right)^{\gamma_0} = \left((g^{a^2})^{\gamma_0}\right)^{\frac{b}{a^2}} = g_0^{r^*},\\
C_1'^* &= e(g, g^b)^{x_{i'} x_{i^*} \gamma_1} = e\left(g^{a x_{i'}}, g^{\gamma_1}\right)^{\frac{b}{a^2} \cdot a x_{i^*}} = e(pk_{i'}, g_1)^{r^* sk_{i^*}} = K^{* sk_{i^*}},\\
K^* &= Z^{x_{i'} \gamma_1} = e(g, g)^{\frac{b}{a} x_{i'} \gamma_1} = e\left(g^{a x_{i'}}, g^{\gamma_1}\right)^{\frac{b}{a^2}} = e(pk_{i'}, g_1)^{r^*},\\
C_3^* &= (g^b)^{\beta_1 h^* + \beta_2 t^* + \beta_3} = \left((g^{a^2})^{\beta_1 h^* + \beta_2 t^* + \beta_3}\right)^{\frac{b}{a^2}} = \left(g^{\alpha_1 h^*} g^{-\frac{\alpha_1 h^* + \alpha_3}{\alpha_2} \cdot \alpha_2} g^{\alpha_3} \cdot (g^{a^2})^{\beta_1 h^* + \beta_2 t^* + \beta_3}\right)^{r^*}\\
&= \left(\left(g^{\alpha_1} (g^{a^2})^{\beta_1}\right)^{h^*} \cdot \left(g^{\alpha_2} (g^{a^2})^{\beta_2}\right)^{t^*} \cdot \left(g^{\alpha_3} (g^{a^2})^{\beta_3}\right)\right)^{r^*} = \left(u_1^{h^*} u_2^{t^*} u_3\right)^{r^*}.
\end{aligned}
$$

- if $\mathcal{COIN} = 1$, letting $r^* = \frac{b}{a}$, we have

$$
\begin{aligned}
C_0^* &= \left(g^b\right)^{\gamma_0} = ((g^a)^{\gamma_0})^{\frac{b}{a}} = g_0^{r^*},\\
C_1'^* &= e(g, g^b)^{x_{i'} x_{i^*} \gamma_1} = e\left(g^{x_{i'}}, g^{\gamma_1}\right)^{\frac{b}{a} \cdot a x_{i^*}} = e(pk_{i'}, g_1)^{r^* sk_{i^*}} = K^{* sk_{i^*}},\\
K^* &= Z^{x_{i'} \gamma_1} = e(g, g)^{\frac{b}{a} x_{i'} \gamma_1} = e\left(g^{x_{i'}}, g^{\gamma_1}\right)^{\frac{b}{a}} = e(pk_{i'}, g_1)^{r^*},\\
C_3^* &= (g^b)^{\beta_1 h^* + \beta_2 t^* + \beta_3} = \left((g^a)^{\beta_1 h^* + \beta_2 t^* + \beta_3}\right)^{\frac{b}{a}} = \left(g^{\alpha_1 h^*} g^{-\frac{\alpha_1 h^* + \alpha_3}{\alpha_2} \cdot \alpha_2} g^{\alpha_3} \cdot (g^a)^{\beta_1 h^* + \beta_2 t^* + \beta_3}\right)^{r^*}\\
&= \left(\left(g^{\alpha_1} (g^a)^{\beta_1}\right)^{h^*} \cdot \left(g^{\alpha_2} (g^a)^{\beta_2}\right)^{t^*} \cdot \left(g^{\alpha_3} (g^a)^{\beta_3}\right)\right)^{r^*} = \left(u_1^{h^*} u_2^{t^*} u_3\right)^{r^*}.
\end{aligned}
$$

On the other hand, if Z is distributed uniformly and independently over $\mathbb{G}_T$, so is K^*, and m_δ is blinded by the pseudorandom value $[F(K^*, C_1^*)]$.

`Guess Stage.` Adversary $\mathcal{A}$ continues to issue the rest queries. In this stage, whenever $\mathcal{B}$ finds a collision of $h^* = H(pk_{i'}, C_0^*, C_2^*)$, $\mathcal{B}$ outputs a random bit and aborts, which is referred to as **collision abort** for presentation simplicity. Otherwise, $\mathcal{B}$ can respond these queries for $\mathcal{A}$ as in the `find stage` (recall that $\mathcal{A}$ has to follow the restrictions described in experiment $\mathrm{Exp}_{\mathrm{PRE},\mathcal{A}}^{\textsf{IND-PRE-tCCA}}$).

`Output.` Eventually, adversary $\mathcal{A}$ returns a guess $\delta' \in \{0, 1\}$. If $\delta' = \delta$, $\mathcal{B}$ outputs 1 to guess $Z = e(g, g)^{b/a}$; otherwise, $\mathcal{B}$ outputs 0 to guess that Z is a random element in $\mathbb{G}_T$.

This completes the description of the simulations. Let Abort denote the event that $\mathcal{B}$ aborts (when $Z = e(g, g)^{b/a}$) during the simulation of oracles $\mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{d}}$ (in both the `find` state and the `guess` stage) or in `Challenge` stage or due to **collision abort** in `Guess` stage. We have $\Pr[\neg\mathsf{Abort}] \geq \theta^{1+q_{\mathrm{sk}}}(1 - \frac{1}{p})^{q_{\mathrm{d}}}(1 - \theta)^2(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\textsf{TCR}})$. By setting $\theta = 1 - \frac{2}{3+q_{\mathrm{sk}}}$, we have $\Pr[\neg\mathsf{Abort}] \geq \frac{4}{\hat{e}^2(3+q_{\mathrm{sk}})^2}(1 - \frac{q_{\mathrm{d}}}{p})(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\textsf{TCR}})$. Similarly to the analysis in Theorem 3, we can have $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{\text{2-wDBDHI}} \geq \frac{|\pm\frac{1}{2}\epsilon - \epsilon_{prf}|}{\hat{e}^2(3+q_{\mathrm{sk}})^2}(1 - \frac{q_{\mathrm{d}}}{p})(1 - \mathrm{Adv}_{H,\mathcal{A}}^{\textsf{TCR}})$. Note that, under the assumption that the underlying H and F are secure, if ϵ is non-negligible in k, so is ϵ'. It's also easy to check that, by straightforward computation, the running time of $\mathcal{B}$ is bounded by $t' \leq t + O(\tau(q_{\mathrm{pk}} + q_{\mathrm{rk}} + 8q_{\mathrm{d}}))$. □

Though the IND-PRE-oCCA (resp., IND-PRE-tCCA) security already implies the NT (resp., MSS) security of our scheme. We show, in Appendix E, that the NT (resp., MSS) security can be established *merely* upon some weaker assumption introduced and justified in this work, which might be of independent interest, without additionally relying upon the hardness assumptions of H or F.

References

[1] Michel Abdalla, Mihir Bellare, and Phillip Rogaway. The Oracle Diffie-Hellman Assumptions and an Analysis of DHIES. In *CT-RSA*, pages 143–158, 2001.

[2] Masayuki Abe, Yang Cui, Hideki Imai, and Eike Kiltz. Efficient Hybrid Encryption from ID-Based Encryption. *Des. Codes Cryptography*, 54(3):205–240, 2010.

[3] Giuseppe Ateniese, Karyn Benson, and Susan Hohenberger. Key-Private Proxy Re-encryption. In *CT-RSA*, pages 279–294, 2009.

[4] Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage. In *NDSS*. The Internet Society, 2005.

[5] Giuseppe Ateniese, Kevin Fu, Matthew Green, and Susan Hohenberger. Improved Proxy Re-Encryption Schemes with Applications to Secure Distributed Storage. *ACM Trans. Inf. Syst. Secur.*, 9(1):1–30, 2006.

[6] Mihir Bellare and Amit Sahai. Non-Malleable Encryption: Equivalence between Two Notions, and an Indistinguishability-Based Characterization. In *CRYPTO*, volume 1666 of *LNCS*. Springer, 1999.

[7] Matt Blaze, Gerrit Bleumer, and Martin Strauss. Divertible Protocols and Atomic Proxy Cryptography. In *EUROCRYPT*, pages 127–144, 1998.

[8] Dan Boneh and Xavier Boyen. Short Signatures Without Random Oracles. In *EUROCRYPT*, volume 3027 of *LNCS*, pages 56–73, 2004.

[9] Dan Boneh, Xavier Boyen, and Eu-Jin Goh. Hierarchical Identity Based Encryption with Constant Size Ciphertext. In *EUROCRYPT*, pages 440–456, 2005.

[10] Dan Boneh and Matthew Franklin. Identity-Based Encryption from the Weil Pairing. In *CRYPTO*, volume 2139 of *LNCS*. Springer, 2001.

[11] Xavier Boyen, Qixiang Mei, and Brent Waters. Direct Chosen Ciphertext Security from Identity-Based Techniques. In *ACM Conference on Computer and Communications Security*, pages 320–329, 2005.

[12] Ran Canetti, Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Adaptive Security for Threshold Cryptosystems. In *CRYPTO*, pages 98–115, 1999.

[13] Ran Canetti, Shai Halevi, and Jonathan Katz. Chosen-Ciphertext Security from Identity-Based Encryption. In *EUROCRYPT*, pages 207–222, 2004.

[14] Ran Canetti and Susan Hohenberger. Chosen-Ciphertext Secure Proxy Re-Encryption. In *ACM Conference on Computer and Communications Security*, pages 185–194, 2007.

[15] Jean-Sébastien Coron. On the Exact Security of Full Domain Hash. In *CRYPTO*, pages 229–235, 2000.

[16] Ronald Cramer and Victor Shoup. A Practical Public Key Cryptosystem Provably Secure Against Adaptive Chosen Ciphertext Attack. In *CRYPTO*, pages 13–25, 1998.

[17] Yevgeniy Dodis and Nelly Fazio. Public Key Trace and Revoke Scheme Secure against Adaptive Chosen Ciphertext Attack. In *Public Key Cryptography*, pages 100–115, 2003.

[18] Danny Dolev, Cynthia Dwork, and Moni Naor. Non-Malleable Cryptography (Extended Abstract). In *STOC*, pages 542–552. ACM, 1991.

[19] Oded Goldreich, Shafi Goldwasser, and Silvio Micali. How to Construct Random Functions. *J. ACM*, 33(4):792–807, 1986.

[20] Shafi Goldwasser and Mihir Bellare. *Lecture Notes on Cryptography, MIT*.

[21] Susan Hohenberger. *Advances in signatures, encryption, and e-cash from bilinear groups.* Ph.D Thesis, MIT, 2002.

[22] Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely Obfuscating Re-Encryption. In *TCC*, pages 233–252, 2007.

[23] Susan Hohenberger and Brent Waters. Realizing Hash-and-Sign Signatures under Standard Assumptions. In Antoine Joux, editor, *EUROCRYPT*, volume 5479 of *Lecture Notes in Computer Science*, pages 333–350. Springer, 2009.

[24] Anca-Andreea Ivan and Yevgeniy Dodis. Proxy Cryptography Revisited. In *NDSS*. The Internet Society, 2003.

[25] Eike Kiltz. Chosen-Ciphertext Secure Identity-Based Encryption in the Standard Model with short Ciphertexts. Cryptology ePrint Archive, Report 2006/122, 2006.

[26] Eike Kiltz and David Galindo. Direct Chosen-Ciphertext Secure Identity-Based Key Encapsulation without random oracles. *Theor. Comput. Sci.*, 410(47-49):5093–5111, 2009.

[27] Junzuo Lai, Robert H. Deng, Shengli Liu, and Weidong Kou. Efficient CCA-Secure PKE from Identity-Based Techniques. In *CT-RSA*, pages 132–147, 2010.

[28] Benoît Libert and Damien Vergnaud. Tracing malicious proxies in proxy re-encryption.

[29] Benoit Libert and Damien Vergnaud. Unidirectional Chosen-Ciphertext Secure Proxy Re-Encryption. *IEEE Transactions on Information Theory*, 57:1786–1802.

[30] Benoît Libert and Damien Vergnaud. Unidirectional Chosen-Ciphertext Secure Proxy Re-Encryption. In *Public Key Cryptography*, pages 360–379, 2008.

[31] Amit Sahai. Non-Malleable Non-Interactive Zero Knowledge and Adaptive Chosen-Ciphertext Security. In *FOCS*, pages 543–553, 1999.

[32] Rui Zhang. Tweaking TBE/IBE to PKE Transforms with Chameleon Hash Functions. In *ACNS*, pages 323–339, 2007.

A Comparisons with CH07 Scheme

In Table 1, we compare our CCA-secure scheme with the Canetti-Hohenberger multi-hop bidirectional PRE scheme [14] (referred to as CH07 scheme) that is CCA-secure in the standard model. We first explain some notations used in Table 1 (page 21). Here $|\mathbb{G}|$ and $|\mathbb{G}_T|$ denote the bit-length of an element in groups $\mathbb{G}$ and $\mathbb{G}_T$, respectively. $|\mathcal{M}|$ denotes the bit-length of the plaintext in CH07 scheme, and ℓ denotes the security parameter in our scheme. $|svk|$ and $|\sigma|$ denote the bit-length of the verification key and signature of one-time signature used in CH07 scheme, respectively. We use $t_p, t_e, t_{me}, t_s, t_v$ to represent the computational cost of a bilinear pairing, an exponentiation, a multi-exponentiation, one signing and one verifying a one-time signature, respectively.

B Preliminaries

Definition 8 (Target-collision resistant hash function)**.** *Let $H : X \to Y$ be a hash function. For an algorithm $\mathcal{B}$, define its advantage as*

$$\mathrm{Adv}_{H,\mathcal{B}}^{\mathsf{TCR}} = \Pr\left[x \leftarrow X, x' \leftarrow \mathcal{B}(x) : x' \neq x \wedge H(x') = H(x)\right].$$

We say that H is target-collision resistant if for any probabilistic polynomial-time (PPT) algorithm $\mathcal{B}$, its advantage $\mathrm{Adv}_{H,\mathcal{B}}^{\mathsf{TCR}}$ is negligible.

Definition 9 (Pseudorandom function (PRF) [19])**.** *Let $F : \mathcal{K} \times D \to R$ be a* polynomial-time computable *function family, where $\mathcal{K}$ is the set of keys of F, and D is the domain and R is the*

Schemes	Our Scheme	CH07 Scheme
MSS-secure?	✓	×
non-transitive?	✓	×
single-hop or multi-hop	single-hop	multi-hop
original ciphertext length	$1\lvert\mathbb{Z}_p\rvert+4\lvert\mathbb{G}\rvert+\ell$	$\lvert svk\rvert+3\lvert\mathbb{G}\rvert+1\lvert\mathbb{G}_T\rvert+1\lvert\sigma\rvert$
transformed ciphertext length	$1\lvert\mathbb{Z}_p\rvert+3\lvert\mathbb{G}\rvert+1\lvert\mathbb{G}_T\rvert+\ell$	$\lvert svk\rvert+3\lvert\mathbb{G}\rvert+1\lvert\mathbb{G}_T\rvert+1\lvert\sigma\rvert$
encryption cost	$1t_p + 1t_{me} + 2t_e$	$1t_p + 1t_{me} + 2t_e + 1t_s$
re-encryption cost	$3t_p + 2t_{me}$	$2t_p + 2t_{me} + 1t_e + 1t_v$
original decryption cost	$3t_p + 2t_{me} + t_e$	$3t_p + 2t_{me} + 1t_e + 1t_v$
transformed decryption cost	$2t_p + 2t_{me} + t_e$	$3t_p + 2t_{me} + 1t_e + 1t_v$
adaptive sk corruption?	Yes	Unknown
assumption	2-wDBDHI/1-AwDBDHI	DBDH

Table 1: Comparisons between Our CCA-Secure Scheme and CH07 Scheme

range. Let $\mathsf{Func}(D, R)$ *be the family of all functions mapping* D *to* R*. For a PPT adversary* $\mathcal{F}$*, we define its advantage as*

$$\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}} = \left|\Pr\left[\mathbf{Exp}_{F,\mathcal{F}}^{\mathsf{PRF}-1} = 1\right] - \Pr\left[\mathbf{Exp}_{F,\mathcal{F}}^{\mathsf{PRF}-0} = 1\right]\right|,$$

where experiments $\mathbf{Exp}_{F,\mathcal{F}}^{\mathrm{PRF}-1}$ *and* $\mathbf{Exp}_{F,\mathcal{F}}^{\mathrm{PRF}-0}$ *are defined as below:*

Experiment $\mathbf{Exp}_{F,\mathcal{F}}^{\mathsf{PRF}-1}$	**Experiment** $\mathbf{Exp}_{F,\mathcal{F}}^{\mathsf{PRF}-0}$
$K \xleftarrow{\$} \mathcal{K}$	$g \xleftarrow{\$} \mathsf{Func}(D, R)$
$b \xleftarrow{\$} \mathcal{F}^{F(K,\cdot)}$	$b \xleftarrow{\$} \mathcal{F}^{g(\cdot)}$
Return b	Return b

We say that F *is a pseudorandom function family, if for any PPT adversary* $\mathcal{F}$*, its advantage* $\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}}$ *is negligible.*

C Results on General Diffie-Hellman Problem and the Hardness of q-AwDBDHI Assumption

In this section, we review Boneh et al.'s results [9] on the general Diffie-Hellman problem in the generic bilinear group model, and analyze the relative hardness of the q-AwDBDHI assumption in the generic bilinear group model.

C.1 An Equivalent Definition of q-AwDBDHI Problem

For an easy analysis of the relative hardness of the q-AwDBDHI problem in the generic bilinear group model, we here first describe an equivalent variant of q-AwDBDHI problem.

Proposition 2. *The q-AwDBDHI problem in groups* $(\mathbb{G}, \mathbb{G}_T)$ *is equivalent to given* $(g, g^{a^2}, g^{a^3}, \cdots, g^{a^{q+2}}, g^{a^2 b}, Z) \in \mathbb{G}^{q+3} \times \mathbb{G}_T$ *with unknown* $a, b \xleftarrow{\$} \mathbb{Z}_p^*$*, to decide whether* $Z = e(g, g)^{a^3 b}$ *holds.*

Proof. Given input $(g, g^{a^2}, g^{a^3}, \cdots, g^{a^{q+2}}, g^{a^2 b}, Z)$, we can construct a q-AwDBDHI instance by setting $(y = g^{a^2}, y^{1/A^2} = g, y^A = g^{a^3}, \cdots, y^{A^q} = g^{a^{q+2}}, y^B = g^{a^2 b})$, which implicitly defines $A = a$ and $B = b$. Then, we have $e(y, y)^{B/A} = e(g^{a^2}, g^{a^2})^{b/a} = e(g, g)^{a^3 b}$. The converse implication is easily established and demonstrates the equivalence between both problems. □

C.2 General Diffie-Hellman Problem

Before giving the definition of the general Diffie-Hellman problem, we review some notations used in [9]. Let p be an integer and let s, n be positive integers. Let $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]^s$ be two s-

tuples of n-variate polynomials over $\mathbb{F}_p$ and let $f \in \mathbb{F}_p[X_1, \cdots, X_n]$. We write $P = (p_1, p_2, \cdots, p_s)$ and $Q = (q_1, q_2, \cdots, q_s)$. It is required that both p_1 and q_1 are constant polynomial 1. For a set Ω, a function $h : \mathbb{F}_p \to \Omega$, and a vector $(x_1, \cdots, x_n) \in \mathbb{F}_p^n$, we write

$$h(P(x_1, \cdots, x_n)) = \big(h(p_1(x_1, \cdots, x_n)), \cdots, h(p_s(x_1, \cdots, x_n))\big) \in \Omega^s,$$
$$h(Q(x_1, \cdots, x_n)) = \big(h(q_1(x_1, \cdots, x_n)), \cdots, h(q_s(x_1, \cdots, x_n))\big) \in \Omega^s.$$

For a polynomial $f \in \mathbb{F}_p[X_1, \cdots, X_n]$, we let d_f denote the total degree of f. For a set $P \subseteq \mathbb{F}_p[X_1, \cdots, X_n]^s$, we let $d_P = \max\{d_f | f \in P\}$.

Let $\mathbb{G}_0, \mathbb{G}_1$ be groups of order p and let $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$ be a non-degenerate bilinear map. Let g be a generator of $\mathbb{G}_0$ and set $g_1 = e(g, g) \in \mathbb{G}_1$.

Now the (P, Q, f)-*Diffie-Hellman problem* in $\mathbb{G}$ is depicted as below: given the vector

$$H(x_1, \cdots, x_n) = \left(g^{P(x_1,\cdots,x_n)}, g_1^{Q(x_1,\cdots,x_n)}\right) \in \mathbb{G}_0^s \times \mathbb{G}_1^s,$$

compute $g_1^{f(x_1,\cdots,x_n)} \in \mathbb{G}_1$.

Example C.1 Many assumptions, including BDH assumption and the computational version of q-AwDBDHI assumption, fall in the (P, Q, f)-Diffie-Hellman assumptions.

- BDH assumption: Set $P = (1, x, y, z)$, $Q = (1)$, $f = xyz$. Here $s = 4, n = 3, d_P = 1, d_Q = 0$ and $d_f = 3$.
- Computational version of q-AwBDHI assumption: Set $P = (1, x^2, x^3, \cdots, x^{q+2}, x^2 y)$, $Q = (1)$, $f = x^3 y$. Here $s = q + 3, n = 2, d_P = q + 2, d_Q = 0$ and $d_f = 4$.

To obtain the most general result, Boneh et al. [9] considered the decisional version of (P, Q, f)-Diffie-Hellman problem. It is said that an algorithm $\mathcal{B}$ that outputs $b \in \{0, 1\}$ has advantage ϵ in solving the decisional (P, Q, f)-Diffie-Hellman problem if

$$\Pr\Big[\mathcal{B}\big(H(x_1, \cdots, x_n), g_1^{f(x_1,\cdots,x_n)}\big) = 0\Big] - \Pr\Big[\mathcal{B}\left(H(x_1, \cdots, x_n), T\right) = 0\Big] > \epsilon,$$

where the probability is over the random choice of generator $g \in \mathbb{G}_0$, the random choice of $x_1, \cdots, x_n$ in $\mathbb{F}_p$, the random choice of $T \in \mathbb{G}_1$, and the random bits consumed by $\mathcal{B}$.

C.3 Complexity Lower Bound in Generic Bilinear Groups

We first review what is a generic bilinear group. Consider two random encodings ξ_0, ξ_1 of the additive group $\mathbb{Z}_p^+$, i.e. injective maps $\xi_0, \xi_1 : \mathbb{Z}_p^+ \to \{0, 1\}^m$. For $i = 0, 1$, denote $\mathbb{G}_i = \{\xi_i(x) | x \in \mathbb{Z}_p^+\}$. We are given oracles to compute the induced group action on $\mathbb{G}_0, \mathbb{G}_1$, and an oracle to compute a non-degenerate bilinear map $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$. Group $\mathbb{G}_0$ is called a *generic bilinear group*.

Before giving the complexity lower bound for the decisional (P, Q, f)-Diffie-Hellman problem in generic bilinear groups, we review the following useful definitions.

Let $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]^s$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$. Write $P = (p_1, \cdots, p_s)$ and $Q = (q_1, \cdots, q_s)$ where $p_1 = q_1 = 1$. We say that a polynomial $f \in \mathbb{F}_p[X_1, \cdots, X_n]$ is dependent on the sets (P, Q) if there exists $s^2 + s$ constants $\{a_{i,j}\}_{i,j=1}^s, \{b_k\}_{k=1}^s$ such that $f = \sum_{i,j=1}^s a_{i,j} p_i p_j + \sum_{k=1}^s b_k q_k$. We say that f *is independent of* (P, Q) if f is not dependent on (P, Q). It can be verified that, as to the assumptions listed in Example A.1, polynomial f is indeed independent of (P, Q).

Boneh et al. gave the following theorem, which gives a lower bound on the advantage of a generic algorithm in solving the decisional (P, Q, f)-Diffie-Hellman problem.

Theorem 5. *Let $P, Q \in \mathbb{F}_p[X_1, \cdots, X_n]^s$ be two s-tuples of n-variate polynomials over $\mathbb{F}_p$ and let $f \in \mathbb{F}_p[X_1, \cdots, X_n]$. Let $d = max(2d_P, d_Q, d_f)$. Let ξ_0, ξ_1 and $\mathbb{G}_0, \mathbb{G}_1$ be defined as above. If f is independent of (P, Q), then for any $\mathcal{A}$ that makes a total of at most q_o queries to the oracles* *computing the group operation in $\mathbb{G}_0, \mathbb{G}_1$ and the bilinear pairing $e : \mathbb{G}_0 \times \mathbb{G}_0 \to \mathbb{G}_1$, we have*

$$\left| Pr\begin{bmatrix} x_1, \cdots, x_n, y \xleftarrow{\$} \mathbb{F}_p; b \xleftarrow{\$} \{0,1\}; t_b \leftarrow f(x_1, \cdots, x_n); t_{1-b} \leftarrow y : \\ \mathcal{A}\big(p, \xi_0(P(x_1, \cdots, x_n)), \xi_1(Q(x_1, \cdots, x_n)), \xi_1(t_0), \xi_1(t_1)\big) = b \end{bmatrix} - \frac{1}{2} \right| \leq \frac{d \cdot (q_o + 2s + 2)^2}{2p}.$$

Table 2 gives the complexity lower bound for the decisional version of the assumptions listed in Example A.1 (i.e., DBDH and q-AwDBDHI).

Assumptions	DBDH	q-AwDBDHI
Complexity lower bound	$\frac{3(q_o+10)^2}{2p}$	$\frac{2(q+2)(q_o+2q+8)^2}{2p}$

Table 2: Complexity lower-bound of DBDH and q-AwDBDHI in the generic bilinear group model

As to the 1-AwDBDHI and 2-AwDBDHI assumptions, on which our CCA-secure PRE scheme is based, its complexity lower bound in the generic bilinear group is $\frac{6(q_o+10)^2}{2p}$ and $\frac{6(q_o+12)^2}{2p}$ respectively. Hence, a generic attacker's advantage in 1-AwDBDHI (resp. 2-AwDBDHI) is about twice (resp. thrice) of that in DBDH.

D More on the MSS-Security and NT-Security for Bidirectional PRE

D.1 Implication of the MSS Security

We show that, for single-hop bidirectional PRE, MSS-security as defined in Definition 7 is implied by the $\mathsf{IND\text{-}PRE\text{-}tCCA}$ security.

Proposition 3. *For a single-hop bidirectional PRE $\mathcal{E}$, the master secret security is implied by the* $\mathsf{IND\text{-}PRE\text{-}tCCA}$ *security. That is, if there exists an adversary $\mathcal{A}$ who can break the* MSS *security of $\mathcal{E}$, then there also exists an adversary $\mathcal{B}$ who can also break the* $\mathsf{IND\text{-}PRE\text{-}tCCA}$ *security of $\mathcal{E}$.*

Proof. We show how to construct an adversary $\mathcal{B}$ to break $\mathcal{E}$'s $\mathsf{IND\text{-}PRE\text{-}tCCA}$ security by interacting with $\mathcal{A}$. $\mathcal{B}$ interacts with $\mathcal{A}$ as follows:

`Initialize.` $\mathcal{B}$ obtains the public parameter *param* from its challenger in experiment $\mathbf{Exp}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}_{\mathrm{PRE},\mathcal{A}}$. Then $\mathcal{B}$ forwards *param* to $\mathcal{A}$.

`Query Stage` (corresponding to the `find` stage in $\mathbf{Exp}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}_{\mathrm{PRE},\mathcal{A}}$). $\mathcal{A}$ issues a series of queries $\mathcal{O}_{\mathrm{pk}}, \mathcal{O}_{\mathrm{sk}}, \mathcal{O}_{\mathrm{rk}}$. $\mathcal{B}$ forwards these queries to its own challenger in experiment $\mathbf{Exp}^{\mathsf{IND\text{-}PRE\text{-}tCCA}}_{\mathrm{PRE},\mathcal{A}}$, and then returns the corresponding results to $\mathcal{A}$.

`Output.` $\mathcal{A}$ returns a private key sk_{i^*}, whose public key pk_{i^*} is generated by oracle $\mathcal{O}_{\mathrm{pk}}$, to $\mathcal{B}$. $\mathcal{B}$ outputs $(pk_{i^*}, pk_{i'}, m_0, m_1)$ to its challenger, where $pk_{i'}$ is a public key generated by oracle $\mathcal{O}_{\mathrm{pk}}$, and m_0, m_1 are two equal-length messages chosen randomly by $\mathcal{B}$. Then $\mathcal{B}$ is given a challenge *transformed* ciphertext $\mathtt{CT}^* = \mathsf{ReEnc}(rk_{i' \leftrightarrow i^*}, \mathsf{Enc}(pk_{i'}, m_\delta))$. Using sk_{i^*}, $\mathcal{B}$ can decrypt $\mathtt{CT}^*$, and then recover δ. Therefore, $\mathcal{B}$ can break the $\mathsf{IND\text{-}PRE\text{-}tCCA}$ security of $\mathcal{E}$. □

Note that the MSS security is not implied by the $\mathsf{IND\text{-}PRE\text{-}oCCA}$ security, as particularly no restriction is posed on the oracle access to $\mathcal{O}_{\mathrm{rk}}$ by $\mathcal{A}$ (in particular, $\mathcal{A}$ can get any re-encryption key including those between corrupted users and uncorrupted users).

E Analysis of MSS and NT Security

We first introduce the complexity assumption to be used to prove the MSS security of the single-hop bidirectional PRE scheme proposed in Section 5.

Definition 10. *The q-strong discrete logarithm (q-*SDL*) problem in group* $\mathbb{G}$ *is, given* $(g, g^a, g^{a^2}, \cdots, g^{a^q}) \in \mathbb{G}^{q+1}$*, where* $a \xleftarrow{\$} \mathbb{Z}_p^*$*, to output* a*. For an adversary* $\mathcal{B}$*, we define its advantage in solving the q-*SDL *problem in group* $\mathbb{G}$ *as*

$$\mathrm{Adv}_{\mathcal{B}}^{q\text{-SDL}} \triangleq \Pr[\mathcal{B}(g, g^a, g^{a^2}, \cdots, g^{a^q}) = a].$$

We say that the (t, ϵ)*-q-*SDL *assumption holds in* $\mathbb{G}$ *if no t-time adversary* $\mathcal{B}$ *has advantage at least* ϵ *in solving the q-*SDL *problem in* $\mathbb{G}$*.*

The q-SDL assumption is quite mild, which is particularly weaker than the (widely used) q-strong Diffie-Hellman assumption [8] and also the q-wDBDHI assumption reviewed above. Note also that the 1-SDL assumption is just the traditional DL assumption. In this work, we only employ the traditional DL assumption and the 2-SDL assumption.

Recall that the IND-PRE-tCCA security of our scheme proposed in Section 5 is based on the 2-wDBDHI assumption and assumes target-collision resistant hash functions and pseudorandom functions. Next, we show that its MSS security actually can be established *merely* upon the weaker 2-SDL assumption and without additionally assuming collision-resistant hash functions or PRFs.

Theorem 6. *Our PRE scheme proposed in Section 5 is* MSS *secure, assuming the* 2-SDL *assumption (i.e., the q-*SDL *assumption for* $q = 2$*) holds in group* $\mathbb{G}$*. Specifically, suppose there exists an adversary* $\mathcal{A}$ *who can break the* $(t, q_{\mathrm{pk}}, q_{\mathrm{sk}}, q_{\mathrm{rk}}, \epsilon)$*-MSS security of our scheme, then there exists an algorithm* $\mathcal{B}$ *which can break the* (t', ϵ')*-2-SDL assumption in* $\mathbb{G}$ *with* $\epsilon' \geq \frac{\epsilon}{\dot{e}(1+q_{\mathrm{sk}})}$ *and* $t' \leq t + O(\tau(q_{\mathrm{pk}} + q_{\mathrm{rk}}))$*.*

Proof. Given a 2-SDL instance $(g, g^a, g^{a^2}) \in \mathbb{G}^3$ with unknown $a \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to output the value a. $\mathcal{B}$ works by interacting with adversary $\mathcal{A}$ in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{MSS}}$ as follows:

`Initialize.` $\mathcal{B}$ provides $\mathcal{A}$ with public parameter including $g_0 = g^{\gamma_0}, g_1 = g^{\gamma_1}, u_1 = g^{\beta_1}, u_2 = g^{\beta_2}$ and $u_3 = g^{\beta_3}$ for random $\gamma_0, \gamma_1, \beta_1, \beta_2, \beta_3 \xleftarrow{\$} \mathbb{Z}_p^*$.

`Query Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{MSS}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\mathrm{pk}}(i)$: $\mathcal{B}$ picks $x_i \xleftarrow{\$} \mathbb{Z}_p^*$. Next, using the Coron's technique [15], it flips a biased coin $c_i \in \{0, 1\}$ that yields 1 with probability θ and 0 otherwise, where θ is a fixed probability to be determined later. If $c_i = 1$, it sets $pk_i = g^{x_i}$ (meaning that $sk_i = x_i$); else $pk_i = (g^a)^{x_i}$ (meaning that $sk_i = ax_i$). Next, it adds the tuple (pk_i, x_i, c_i) to L^{list} and returns pk_i to $\mathcal{A}$.
- *Secret key oracle* $\mathcal{O}_{\mathrm{sk}}(pk_i)$: $\mathcal{B}$ first recovers (pk_i, x_i, c_i) from L^{list}. If $c_i = 1$, $\mathcal{B}$ returns $sk_i = x_i$ to $\mathcal{A}$; otherwise, it outputs a random element in $\mathbb{Z}_p^*$ and **aborts**.
- *Re-encryption key oracle* $\mathcal{O}_{\mathrm{rk}}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and responds according to the following three cases:
  - $c_i = c_j = 1$: $\mathcal{B}$ returns $g_1^{x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: $\mathcal{B}$ returns $rk_{i \leftrightarrow j} = (g^{a^2})^{\gamma_1 x_i x_j} = (g^{\gamma_1})^{a^2 x_i x_j} = g_1^{a x_i a x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i \neq c_j$: $\mathcal{B}$ returns $rk_{i \leftrightarrow j} = (g^a)^{\gamma_1 x_i x_j} = (g^{\gamma_1})^{a x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.

`Output stage.` Finally, $\mathcal{A}$ outputs a secret key sk_{i^*} with respect to the public key pk_{i^*} which has not been queried to oracle $\mathcal{O}_{\mathrm{sk}}$. $\mathcal{B}$ first recovers $(pk_{i^*}, x_{i^*}, c_{i^*})$ from L^{list}. If $c_{i^*} = 1$, $\mathcal{B}$ outputs a random element in $\mathbb{Z}_p^*$ and **aborts**. Otherwise (it means that $sk_{i^*} = a x_{i^*}$), $\mathcal{B}$ outputs $a = sk_{i^*}/x_{i^*}$ as the solution to the 2-SDL instance.
Analysis. The rest analysis is identical to that of Theorem 2. □

Theorem 7. *Our scheme is* NT *secure, assuming the hash function* H *is target collision resistant,* F *is a PRF family and the* 2-AwDBDHI *assumption holds in groups* $(\mathbb{G}, \mathbb{G}_T)$*.*

Proof. Suppose there exists an adversary $\mathcal{A}$ who can break the $(t, q_{\text{pk}}, q_{\text{sk}}, q_{\text{rk}}, q_{\text{re}}, q_{\text{d}}, \epsilon)$-$\mathsf{NT}$ security of our scheme, then we show how to construct another algorithm $\mathcal{B}$ that can break the 2-AwDBDHI assumption in $(\mathbb{G}, \mathbb{G}_T)$. Given a 2-AwDBDHI instance $(g, g^{\frac{1}{a^2}}, g^a, g^{a^2}, g^b, Z) \in \mathbb{G}^5 \times \mathbb{G}_T$ with unknown $a, b \xleftarrow{\$} \mathbb{Z}_p^*$, $\mathcal{B}$'s goal is to decide whether $Z = e(g, g)^{b/a}$. $\mathcal{B}$ works by interacting with adversary $\mathcal{A}$ in experiment $\mathbf{Exp}^{\mathsf{NT}}_{\text{PRE},\mathcal{A}}$ as follows:

`Initialize.` $\mathcal{B}$ provides $\mathcal{A}$ with public parameter including $g_0 = g^{\gamma_0}, g_1 = \left(g^{\frac{1}{a^2}}\right)^{\gamma_1}, u_1 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_1} g^{\beta_1}$, $u_2 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_2} g^{\beta_2}$ and $u_3 = \left(g^{\frac{1}{a^2}}\right)^{\alpha_3} g^{\beta_3}$ for random $\gamma_0, \gamma_1, \alpha_1, \alpha_2, \alpha_3, \beta_1, \beta_2, \beta_3 \xleftarrow{\$} \mathbb{Z}_p^*$.

`Find Stage.` Adversary $\mathcal{A}$ issues a series of queries as in experiment $\mathbf{Exp}^{\mathsf{NT}}_{\text{PRE},\mathcal{A}}$. $\mathcal{B}$ maintains a list L^{list}, and answers these queries for $\mathcal{A}$ as follows:

- *Public key oracle* $\mathcal{O}_{\text{pk}}(i)$: $\mathcal{B}$ picks $x_i \xleftarrow{\$} \mathbb{Z}_p^*$. Using the Coron's technique [15], it flips a biased variable $c_i \in \{0, 1, '-'\}$ such that $\Pr[c_i = 0] = \Pr[c_i = '-'] = \theta$ and $\Pr[c_i = 1] = 1 - 2\theta$, where θ is a fixed probability to be determined later. If $c_i = 1$, it sets $pk_i = g^{x_i}$ (meaning that $sk_i = x_i$); if $c_i = 0$, it sets $pk_i = (g^a)^{x_i}$ (meaning that $sk_i = ax_i$); if $c_i = '-'$, it sets $pk_i = (g^{a^2})^{x_i}$ (meaning that $sk_i = a^2 x_i$). Next, it adds (pk_i, x_i, c_i) to L^{list} and returns pk_i to $\mathcal{A}$.
- *Secret key oracle* $\mathcal{O}_{\text{sk}}(pk_i)$: $\mathcal{B}$ first recovers (pk_i, x_i, c_i) from L^{list}. If $c_i = 1$, $\mathcal{B}$ returns $sk_i = x_i$ to $\mathcal{A}$; otherwise, it outputs a random bit and **aborts**.
- *Re-encryption key oracle* $\mathcal{O}_{\text{rk}}(pk_i, pk_j)$: $\mathcal{B}$ first recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and responds according to the following three cases:
  - $c_i = c_j = 1$: Return $rk_{i\leftrightarrow j} = g_1^{x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = 0$: Return $rk_{i\leftrightarrow j} = g^{\gamma_1 x_i x_j} = \left(\left(g^{\frac{1}{a^2}}\right)^{\gamma_1}\right)^{ax_i ax_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $c_i = c_j = '-'$: Return $rk_{i\leftrightarrow j} = (g^{a^2})^{\gamma_1 x_i x_j} = \left(\left(g^{\frac{1}{a^2}}\right)^{\gamma_1}\right)^{a^2 x_i a^2 x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $(c_i = '-' \wedge c_j = 1)$ or $(c_i = 1 \wedge c_j = '-')$: Return $rk_{i\leftrightarrow j} = g^{\gamma_1 x_i x_j} = \left(\left(g^{\frac{1}{a^2}}\right)^{\gamma_1}\right)^{a^2 x_i x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $(c_i = '-' \wedge c_j = 0)$ or $(c_i = 0 \wedge c_j = '-')$: Return $rk_{i\leftrightarrow j} = (g^a)^{\gamma_1 x_i x_j} = \left(\left(g^{\frac{1}{a^2}}\right)^{\gamma_1}\right)^{a^2 x_i a x_j} = g_1^{sk_i sk_j}$ to $\mathcal{A}$.
  - $(c_i = 1 \wedge c_j = 0)$ or $(c_i = 0 \wedge c_j = 1)$: Output a random bit and **aborts**.
- *Re-encryption oracle* $\mathcal{O}_{\text{re}}(pk_i, pk_j, \mathtt{CT}_i)$: $\mathcal{B}$ first parses $\mathtt{CT}_i$ as $(pk_i, t, C_0, C_1, C_2, C_3)$, computes $h = H(pk_i, C_0, C_2)$, and then checks the validity of the ciphertext as in Eq. (3). If the verification fails, it returns "$\perp$" to $\mathcal{A}$ (indicating an invalid ciphertext). Otherwise, $\mathcal{B}$ recovers tuples (pk_i, x_i, c_i) and (pk_j, x_j, c_j) from L^{list}, and works according to the following two cases:
  - $c_i \neq c_j$: Without loss of generality, suppose $sk_i = ax_i$ and $sk_j = x_j$. $\mathcal{B}$ first checks whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0$ or not, where $h = H(pk_i, C_0, C_2)$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0$ then $\mathcal{B}$ outputs a random bit and **aborts**. Otherwise, from $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left(\left(g^{\frac{1}{a^2}}\right)^{\alpha_1 h + \alpha_2 t + \alpha_3} g^{\beta_1 h + \beta_2 t + \beta_3}\right)^r$, $\mathcal{B}$ can compute
  $$g_1^r = \left(g^{\frac{\gamma_1}{a^2}}\right)^r = \left(\frac{C_3}{C_0^{\frac{\beta_1 h + \beta_2 t + \beta_3}{\gamma_0}}}\right)^{\frac{\gamma_1}{\alpha_1 h + \alpha_2 t + \alpha_3}}. \tag{6}$$
  Then $\mathcal{B}$ computes $C_1' = e\left(g^a, g_1^r\right)^{x_i x_j}$, and returns $\mathtt{CT}_j = (pk_i, t, C_0, C_1', C_2, C_3)$ to $\mathcal{A}$. Observe that $\mathtt{CT}_j$ is indeed a valid transformed ciphertext as required, since $c_i \neq c_j$ means $sk_i sk_j = ax_i x_j$, and hence we have $C_1' = e\left(g^a, g_1^r\right)^{x_i x_j} = e\left(g^r, g_1^{ax_i x_j}\right) = e(g^r, g_1^{sk_i sk_j}) = e(C_1, rk_{i\leftrightarrow j})$. Observe also that, in Eq. (6), the equality $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$ information-theoretically holds with probability $1/p$.

– Otherwise: $\mathcal{B}$ first generates the re-encryption key $rk_{i\leftrightarrow j}$ as in the response for the re-encryption key oracle $\mathcal{O}_{\text{rk}}$, and returns $\texttt{CT}_j \leftarrow \mathsf{ReEnc}(rk_{i\leftrightarrow j}, \texttt{CT}_i)$ to $\mathcal{A}$.

- *Decryption oracle* $\mathcal{O}_{\text{d}}(pk_z, \texttt{CT}_z)$: $\mathcal{B}$ first recovers tuple (pk_z, x_z, c_z) from L^{list}. If $c_z = 1$ (it means that $sk_z = x_z$), algorithm $\mathcal{B}$ returns the result of $\mathsf{Dec}(x_z, \texttt{CT}_z)$ to $\mathcal{A}$. Otherwise, $\mathcal{B}$ proceeds according to the following two cases:
  – $\texttt{CT}_z = (pk_z, t, C_0, C_1, C_2, C_3)$ is an original ciphertext: Compute $h = H(pk_z, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (3). If the verification fails, output "$\perp$" indicating an invalid ciphertext. Then, check whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$, output a random bit and **abort**; Otherwise, from $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left((g^{\frac{1}{a^2}})^{\alpha_1 h+\alpha_2 t+\alpha_3} g^{\beta_1 h+\beta_2 t+\beta_3}\right)^r$, obtain g_1^r as in Eq. (6). Then, compute $K = e(pk_z, g_1^r)$ and output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$.
  – $\texttt{CT}_z = (pk_i, t, C_0, C_1', C_2, C_3)$ is a transformed ciphertext: Compute $h = H(pk_i, C_0, C_2)$ and then check the validity of the ciphertext as in Eq. (1). If the verification fails, output "$\perp$" indicating an invalid ciphertext. Otherwise, check whether $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$. If $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$, output a random bit and **abort**; Otherwise, continue to execute the following steps:
    1. From $C_0 = g_0^r = g^{r\cdot\gamma_0}$ and $C_3 = (u_1^h u_2^t u_3)^r = \left((g^{\frac{1}{a^2}})^{\alpha_1 h+\alpha_2 t+\alpha_3} g^{\beta_1 h+\beta_2 t+\beta_3}\right)^r$, obtain g_1^r as in Eq. (6).
    2. Recover tuple (pk_i, x_i, c_i) from L^{list}. Verify the validity of C_1' according to four cases: if $c_i = 1$ check whether $C_1' = e(pk_z^{x_i}, g_1^r)$ holds; if $c_i = c_z = 0$ check whether $C_1' = e(g^{x_i x_z \gamma_1}, C_0^{\frac{1}{\gamma_0}})$ holds; if $(c_i = 0 \wedge c_z =' -')$ or $(c_i =' -' \wedge c_z = 0)$ check whether $C_1' = e((g^a)^{x_i x_z \gamma_1}, C_0^{\frac{1}{\gamma_0}})$ holds; if $(c_i = c_z =' -')$ check whether $C_1' = e((g^{a^2})^{x_i x_z \gamma_1}, C_0^{\frac{1}{\gamma_0}})$ holds. If the above verifications fail, output "$\perp$" indicating an invalid ciphertext.
    3. Compute $K = e(pk_i, g_1^r)$ and check whether $[F(K, C_0)]_{\ell-\ell_1} = [C_2]_{\ell-\ell_1}$. If not, output "$\perp$"; otherwise, output $m = [F(K, C_0)]^{\ell_1} \oplus [C_2]^{\ell_1}$.

  Note that, similar to the analysis in oracle $\mathcal{O}_{\text{re}}$, $\alpha_1 h + \alpha_2 t + \alpha_3 = 0 \mod p$ holds with probability $\frac{1}{p}$.

`Challenge.` When $\mathcal{A}$ decides that `find` stage is over, it outputs a public key pk_{i^*} and two equal-length messages $m_0, m_1 \in \{0,1\}^{\ell_1}$ with the restrictions specified in experiment $\mathbf{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{IND-PRE-oCCA}}$. Algorithm $\mathcal{B}$ responds as follows:

1. Recover tuple $(pk_{i^*}, x_{i^*}, c_{i^*})$ from L^{list}. If $c_{i^*} \neq 0$, $\mathcal{B}$ outputs a random bit in $\{0,1\}$ and **aborts**. Otherwise, it means that $pk_{i^*} = (g^a)^{x_{i^*}}$, and $\mathcal{B}$ proceeds to execute the rest steps.
2. Define $C_0^* = \left(g^b\right)^{\gamma_0}, C_1^* = g^b, K^* = Z^{x_{i^*}\gamma_1}, C_2^* = [F(K^*, C_1^*)]_{\ell-\ell_1} \| ([F(K^*, C_1^*)]^{\ell_1} \oplus m_\delta)$ for a random bit δ, $t^* = -\frac{\alpha_1 h^* + \alpha_3}{\alpha_2}$ and $C_3^* = \left(g^b\right)^{\beta_1 h^* + \beta_2 t^* + \beta_3}$ where $h^* = H(pk_{i^*}, C_0^*, C_2^*)$.
3. Return $\texttt{CT}^* = (pk_{i^*}, t^*, C_0^*, C_1^*, C_2^*, C_3^*)$ as the challenge ciphertext to adversary $\mathcal{A}$.

Observe that if $Z = e(g,g)^{b/a}$, it can be verified that the ciphertext $\texttt{CT}^*$ is indeed a valid challenge ciphertext. On the other hand, if Z is distributed uniformly and independently over $\mathbb{G}_T$, so is K^*, and m_δ is blinded by the pseudorandom value $[F(K^*, C_1^*)]$.

`Guess Stage.` Adversary $\mathcal{A}$ continues to issue the rest queries. In this stage, whenever $\mathcal{B}$ finds a collision of $h^* = H(pk_{i^*}, C_0^*, C_2^*)$, $\mathcal{B}$ outputs a random bit and aborts, which is referred to as **collision abort** for presentation simplicity. Otherwise, $\mathcal{B}$ can respond these queries for $\mathcal{A}$ as in the `find stage` (recall that $\mathcal{A}$ has to follow the restrictions described in experiment $\text{Exp}_{\text{PRE},\mathcal{A}}^{\textsf{NT}}$).

`Output.` Eventually, adversary $\mathcal{A}$ returns a guess $\delta' \in \{0,1\}$. If $\delta' = \delta$, $\mathcal{B}$ outputs 1 to guess $Z = e(g,g)^{b/a}$; otherwise, $\mathcal{B}$ outputs 0 to guess that Z is a random element in $\mathbb{G}_T$.

This completes the description of the simulation. Let Abort denote the event that $\mathcal{B}$ aborts (when $Z = e(g,g)^{b/a}$) during the simulation of oracles $\mathcal{O}_{\text{sk}}, \mathcal{O}_{\text{rk}}, \mathcal{O}_{\text{re}}, \mathcal{O}_{\text{d}}$ or in `Challenge` stage or due to **collision abort** in Guess stage. We have $\Pr[\neg\mathsf{Abort}] \geq (1-2\theta)^{q_{\rm sk}}(1-2\theta(1-2\theta))^{q_{\rm rk}}\theta(1-\frac{1}{p})^{q_{\rm re}+q_{\rm d}}(1-\mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}}) \geq (1-2\theta)^{q_{\rm sk}+q_{\rm rk}}\theta(1-\frac{1}{p})^{q_{\rm re}+q_{\rm d}}(1-\mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$. By setting $\theta = \frac{1}{2(1+q_{\rm sk}+q_{\rm rk})}$, we have $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{2\dot{e}(1+q_{\rm sk}+q_{\rm rk})}(1-\frac{q_{\rm re}+q_{\rm d}}{p})(1-\mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$.

Observe that, if event Abort does not occur during the simulation and $Z = e(g,g)^{b/a}$, then the simulation of $\mathcal{B}$ is perfect from the view of $\mathcal{A}$. In this case, $\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^{a^2}, g^b, Z) = 1 | Z = e(g,g)^{b/a}] = \frac{1}{2}\Pr[\mathsf{Abort}] + \Pr[\delta' = \delta|\neg\mathsf{Abort}]\Pr[\neg\mathsf{Abort}] = \frac{1}{2} + \Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta|\neg\mathsf{Abort}] - \frac{1}{2})$. On the other hand, if event **Abort** does not occur during the simulation and Z is distributed uniformly and independently over $\mathbb{G}_T$, the only advantage that A can get is to break the pseudorandomness of the PRF F. In the later case, by standard probabilistic trick [6, 20] we get $\Pr[\delta' = \delta | \mathcal{B} \text{ does not abort when } Z \neq e(g,g)^{b/a}] = \frac{1}{2} \pm \frac{1}{2}\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}}$. Defining $\epsilon_{prf} = \pm\frac{1}{2}\mathrm{Adv}_{F,\mathcal{F}}^{\mathsf{PRF}}\Pr[\mathcal{B} \text{ does not abort when } Z \neq e(g,g)^{b/a}]$ (that is a negligible quantity assuming the underlying PRF is secure), we have $\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^{a^2}, g^b, Z) = 1 | Z \neq e(g,g)^{b/a}] = \frac{1}{2} + \epsilon_{prf}$. Thus, $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{\text{2-AwDBDHI}} = |\Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^{a^2}, g^b, Z) = 1 | Z = e(g,g)^{b/a})] - \Pr[\mathcal{B}(g, g^{\frac{1}{a^2}}, g^a, g^{a^2}, g^b, Z) = 1 | Z \neq e(g,g)^{b/a}]| = |\Pr[\neg\mathsf{Abort}](\Pr[\delta' = \delta|\neg\mathsf{Abort}] - \frac{1}{2}) - \epsilon_{prf}|$. As $\epsilon = \mathrm{Adv}_{\mathrm{PRE},\mathcal{A}}^{\mathsf{IND\text{-}PRE\text{-}oCCA}} = |2\Pr[\delta' = \delta] - 1|$ and $\Pr[\neg\mathsf{Abort}] \geq \frac{1}{2\dot{e}(1+q_{\rm sk}+q_{\rm rk})}(1-\frac{q_{\rm re}+q_{\rm d}}{p})(1-\mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$, we have $\epsilon' = \mathrm{Adv}_{\mathcal{B}}^{\text{2-AwDBDHI}} \geq \frac{|\pm\frac{1}{2}\epsilon - \epsilon_{prf}|}{2\dot{e}(1+q_{\rm sk}+q_{\rm rk})}(1-\frac{q_{\rm re}+q_{\rm d}}{p})(1-\mathrm{Adv}_{H,\mathcal{A}}^{\mathsf{TCR}})$. Note that, under the assumption that the underlying H and F are secure, if ϵ is non-negligible in k, so is ϵ'. It's also easy to check that, by straightforward computation, the running time of $\mathcal{B}$ is bounded by $t' \leq t + O(\tau(q_{\rm pk} + q_{\rm rk} + 8q_{\rm re} + 8q_{\rm d}))$. $\square$