Proxy Re-Encryption and Re-Signatures from Lattices

Xiong Fan[1](✉) and Feng-Hao Liu[2]

[1] Cornell University, Ithaca, NY, USA
xfan@cs.cornell.edu

[2] Florida Atlantic University, Boca Raton, FL, USA
fenghao.liu@fau.edu

Abstract. Proxy re-encryption (PRE) and Proxy re-signature (PRS) were introduced by Blaze, Bleumer and Strauss [Eurocrypt '98]. Basically, PRE allows a semi-trusted proxy to transform a ciphertext encrypted under one key into an encryption of the same plaintext under another key, without revealing the underlying plaintext. Since then, many interesting applications have been explored, and constructions in various settings have been proposed. On the other hand, PRS allows a semi-trusted proxy to transform Alice's signature on a message into Bob's signature on the same message, but the proxy cannot produce new valid signature on new messages for either Alice or Bob.

In this work, we first point out a subtle mistake in the security proof of the work by Kirshanova (PKC '14), who proposed a lattice-based CCA1 PRE. Thus, this reopens the direction of lattice-based CCA1-secure constructions, even in the single-hop setting. Then we construct a single-hop PRE scheme that is proven secure in our new tag-based CCA-PRE model. Next, we construct the *first* multi-hop PRE construction. Lastly, we also construct the *first* PRS scheme from lattices that is proved secure in our proposed unified security model.

1 Introduction

Proxy re-encryption (PRE) allows a (semi-trusted) proxy to transform an encryption of m under Alice's public key into another encryption of the same message under Bob's public key. The proxy, however, cannot learn the underlying message m, and thus both parties' privacy can be maintained. This primitive (and its variants) have various applications ranging from encrypted email forwarding [8], to securing distributed file systems [6]. In addition application-driven purposes, various works have shown connections between re-encryption (and its variants) with other cryptographic primitives, such as program obfuscation [13, 14, 23] and fully-homomorphic encryption [3, 11]. Thus studies along this line are both important and interesting for theory and practice.

Another primitive, called proxy re-signature (PRS), allows a semi-trusted proxy to transform Alice's signature σ_A on a message μ into Bob's signature σ_B on the same message μ, but the proxy cannot produce new valid signature on

© Springer Nature Switzerland AG 2019
R. H. Deng et al. (Eds.): ACNS 2019, LNCS 11464, pp. 363–382, 2019.
https://doi.org/10.1007/978-3-030-21568-2_18

new messages for either Alice or Bob. PRS is employed in various applications, such as providing a proof that a certain path in a graph is taken.

Both concepts of PRE and PRS were introduced by Blaze, Bleumer, and Strauss [8], who also gave the first construction of a CPA (i.e. chosen-plaintext attacks) secure *bi-directional multi-hop* PRE scheme under the Decisional Diffie-Hellman assumption, and a restricted PRS construction. Later on, Ateniese and Hohenberger [7] formalized security notions for PRS, and gave two PRS constructions (one is bi-directional, and the other one is uni-directional) based on bilinear maps in the random oracle model. Ateniese et al. [6] constructed the first CPA secure *uni-directional* scheme based on bilinear maps, yet their construction can only support a *single-hop* re-encryption. Hohenberger et al. [23] and Chandran et al. [14] used an obfuscation-based approach and constructed CPA secure uni-directional single-hop PRE scheme (and its variants). Chandran et al. [13], using the obfuscation-based approach, constructed the first CPA secure uni-directional multi-hop PRE scheme based on lattices assumptions.

For the PRE part, as argued that CPA security can be insufficient for some useful scenarios, Canetti and Hohenberger [10] considered a natural stronger security notion — chosen-ciphertext attacks (CCA) security where the adversary has access to a decryption oracle. Intuitively, this security notion guarantees that the underlying message of the challenge ciphertext remains hidden even if the adversary can somehow obtain decryptions of other ciphertexts. They give a meaningful security formulation of CCA secure PRE, and then constructed the first CCA-secure bidirectional multi-hop PRE scheme. Later, Shao et al. [33] constructed a CCA-secure uni-directional single-hop PRE, and Chow et al. [16] proposed another CCA-secure uni-directional scheme in random oracle model. Libert and Vergnaud [26] improved the result by constructing a CCA uni-directional single-hop PRE without random oracles, and this remains the state of the art of the current construction (for the setting of uni-directional CCA-PRE under the definition of [10]). We note that it is unclear how to extend security of the previous obfuscation-approach [13,14,23] (that are only CPA-secure) to the CCA setting. One particular technical challenge is that the re-encryption key output by the simulator might be distinguishable given the CCA decryption oracle, and thus the previous security analyses cannot go through. For CPA security, our understanding is quite well—we know how to construct PRE schemes that are uni-directional and multi-hop in the standard model. However, for CCA security, our understanding in the standard model is much limited in the following sense. First, there is no known scheme that achieves both uni-directional and multi-hop at the same time. Moreover, all currently known constructions [4,6,8,10,16,26,33] are based on Diffie-Hellman-style assumptions. Then Kirshanova [24] proposed a single-hop construction based on lattices, and argued that it is CCA1 secure[1]. However, after a careful examination of her security proof, we found a subtle mistake in the security proof. As the

[1] CCA1 security is weaker in the sense that the attacker does not have the decryption oracle after receiving the challenge ciphertext.

mistake is not easily fixable, how to construct a lattice-based PRE that achieves CCA1-security, (even for the single-hop case) remains open.

For the PRS part, Ateniese and Hohenberger [7] left some open problems such as how to construct uni-directional PRS where the proxy can only translate signatures in one direction. Can we avoid the random oracle analysis? Libert and Vergnaud [25] answered these questions positively by constructing the first *multi-use* unidirectional PRS in standard model relying on a new computational assumption in bilinear group.

In this paper, we study lattice-based PRE and PRS constructions. In particular, we make contributions in the following four folds:

- First, we point out a subtle mistake in the security proof of the work [24] (the CCA1 construction), and argue that this is not easy to fix. Briefly speaking, the re-encryption key from the challenge user to another honest user generated in the security proof is distinguishable from the real, and thus the analysis breaks down. Therefore, the construction of [24] does not achieve the CCA1 notion considered in most prior work and this paper.
- Second, we propose a new model called tag-based CCA that lies in between the CCA1 and CCA2 model. Our tag-based CCA allows the attacker to query the decryption oracle before and after the challenge ciphertext, and the honest re-encryption oracle who only re-encrypts honestly generated ciphertexts. This is a combination of CCA and a new notion – honest re-encryption (HRA) attacks proposed recently by Cohen [17].
  We then construct a lattice-based PRE scheme that achieves our tag-based CCA notion. We also describe a generic transformation from the relaxed functionality to the full-fledged one using know techniques (i.e., zero-knowledge proofs). Using a recent work that constructs NIZK from circularly secure FHE [12], we are able to achieve the full-fledged CCA-security if we further assume the required circular security on LWE.
- Third we define a selective notion of tag-based CCA security for multi-hop PRE where the attacker needs to commit to a tree structure for the challenging ciphertext at the beginning. Then we prove that our basic single-hop construction, with a slight modification, can be extended to the multi-hop setting and achieve such a security notion. This is, to our knowledge, the *first* construction of multi-hop PRE that achieves a relaxed yet meaningful notion of CCA security.
- Lastly, we propose a simpler and unified security model for PRS which captures more dynamic settings. We show that the idea of our multi-hop PRE model and the construction can be extended to construct PRS that achieves the security notion. This is the *first* (to our knowledge) multi-hop unidirectional PRS from lattices.

1.1 Technique Highlights

In the following, we highlight our technical ideas for the four contributions as described above.

Part I: The Subtle Mistake in the Work [24]**.** The subtle mistake comes in the security proof where the work [24] constructs two adjacent hybrids that are distinguishable. For clarification of exposition, we first briefly present the main idea of the construction [24]. Then we will point out where the subtlety is and explain why the problem cannot be easily fixed.

Basically, the PRE construction can be regarded as an extension of CCA-secure public key encryption scheme in [28]. For concreteness, we consider two users: User 1 has public key $\mathsf{pk}_1 = (\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \mathbf{H})$, and User 2 has public key $\mathsf{pk}_2 = (\mathbf{A}'_0, \mathbf{A}'_1, \mathbf{A}'_2, \mathbf{H}')$, where each public key consists of four matrices. The secret key of User 1 consists of low-norm matrices $\mathbf{R}_1, \mathbf{R}_2$ satisfying $\mathbf{A}_1 = -\mathbf{A}_0\mathbf{R}_1, \mathbf{A}_2 = -\mathbf{A}_0\mathbf{R}_2$, and it is similar for the case of User 2. We note that the readers here do not need to worry about the dimensions. To encrypt under pk_1, we consider an encryption matrix $\mathbf{A}_u = [\mathbf{A}_0|\mathbf{A}_1 + \mathbf{H}\mathbf{G}|\mathbf{A}_2 + \mathbf{H}_u\mathbf{G}]$, where $\mathbf{H}_u$ is a random invertible matrix (as a tag to the ciphertext), then encrypt messages using the dual-Regev style encryption [22], i.e. $\mathsf{ct} = \boldsymbol{s}^\mathsf{T}\mathbf{A}_u + \boldsymbol{e} + \mathsf{encode}(m)$. Similarly, we can encrypt under pk_2 with the same structure.

To generate a re-encryption key from User 1 to User 2, the work [24] considers a short matrix $\mathbf{X}$ satisfying the following relation:

$$[\mathbf{A}_0|\mathbf{A}_1 + \mathbf{H}\mathbf{G}|\mathbf{A}_2 + \mathbf{H}_u\mathbf{G}] \begin{bmatrix} \mathbf{X}_{00} & \mathbf{X}_{01} & \mathbf{X}_{02} \\ \mathbf{X}_{10} & \mathbf{X}_{11} & \mathbf{X}_{12} \\ 0 & 0 & \mathbf{I} \end{bmatrix} = [\mathbf{A}'_0|\mathbf{A}'_1 + \mathbf{H}'\mathbf{G}|\mathbf{A}'_2 + \mathbf{H}_u\mathbf{G}].$$

In particular, for the last column of the re-encryption key matrix, it holds that

$$[\mathbf{A}_0|\mathbf{A}_1 + \mathbf{H}\mathbf{G}] \begin{bmatrix} \mathbf{X}_{02} \\ \mathbf{X}_{12} \end{bmatrix} = \mathbf{A}'_2 - \mathbf{A}_2. \tag{1}$$

It is not hard to see that $\mathsf{ct} \cdot \mathbf{X} = \boldsymbol{s}^\mathsf{T} \cdot \mathbf{A}'_u + \tilde{\boldsymbol{e}} + \mathsf{encode}(m)$, a ciphertext of m under pk_2, so the correctness property is guaranteed.

To prove security, the work [24] uses a standard reduction argument based on the LWE assumption: suppose there exists an adversary that can break the PRE scheme, then there exists a reduction, with oracle access to the adversary, who can break the underlying LWE assumption. For this type of proofs, typically the reduction needs to embed the hard instance (LWE instance for this case), then simulates a scheme (PRE) to the adversary, and finally the reduction can use the adversary to break the underlying hardness assumption. It is **crucially important** that the simulated scheme cannot be distinguished by the adversary; otherwise, the adversary can always output $\perp$ if he detects the scheme is different from the real scheme, and such adversary is useless to the reduction. The security proof in the work [24] missed this point. At a high level, her reduction simulated a PRE scheme that *can* be distinguishable by the adversary easily, so the whole argument breaks down. Below we further elaborate on the details.

For simplicity we consider a simple case where there are only two honest users, Users 1 and 2 and the adversary only gets one re-encryption key from User 1 to User 2. The challenge ciphertext comes from an encryption of User 1, i.e. pk_1. For such case, the reduction of the work [24] pre-selects a tag matrix $\mathbf{H}_{u^*}$ (for the challenge ciphertext), matrices $\mathbf{R}_1^*, \mathbf{R}_2^*$, and then embeds an LWE instance $\mathbf{A}^*$ in the encryption matrix: $\mathbf{A}_u^* = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^* | -\mathbf{A}^*\mathbf{R}_2^* + (\mathbf{H}_u - \mathbf{H}_{u^*})\mathbf{G}]$. In this case, the reduction sets $\mathsf{pk}_1 = (\mathbf{A}_0, \mathbf{A}_1, \mathbf{A}_2, \mathbf{H})$ to be $(\mathbf{A}^*, -\mathbf{A}^*\mathbf{R}_1^* - \mathbf{H}^*\mathbf{G}, -\mathbf{A}^*\mathbf{R}_2^* - \mathbf{H}_{u^*}\mathbf{G}, \mathbf{H}^*)$ for some random invertible $\mathbf{H}^*$.

To generate re-encryption key from the challenge user 1 to User 2, the reduction first pre-samples small matrices $\mathbf{X}_{00}, \mathbf{X}_{01}, \mathbf{R}_1', \mathbf{R}_2'$, and a random invertible matrix $\mathbf{H}'$. Then it computes:

$$\mathbf{A}_0' = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^*] \begin{bmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{bmatrix}, \quad \mathbf{A}_i' = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^*] \begin{bmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{bmatrix} \cdot \mathbf{R}_i', \forall i = 1, 2$$

The reduction sets

$$\mathsf{pk}_2 = (\mathbf{A}_0', \mathbf{A}_1', \mathbf{A}_2', \mathbf{H}'), \quad \mathsf{rk}_{1\to 2} = \begin{bmatrix} \left(\begin{smallmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{smallmatrix}\right) & \left(\begin{smallmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{smallmatrix}\right) \mathbf{R}_1' & \left(\begin{smallmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{smallmatrix}\right) \mathbf{R}_2' \\ 0 & 0 & \mathbf{I} \end{bmatrix}$$

generated as above. Then obviously the matrices $\mathbf{A}_1', \mathbf{A}_2'$ can be expressed as $\mathbf{A}_1' = \mathbf{A}_0'\mathbf{R}_1', \mathbf{A}_2' = \mathbf{A}_0'\mathbf{R}_2'$, where $\mathbf{R}_1', \mathbf{R}_2'$ are small matrices and still act as secret key for User 2. Therefore, the reduction can still use the same algorithm in the real scheme to answer decryption queries for User 2.

However, if $\mathbf{A}_2'$ is generated in this way, then it is easy to check and compare with Eq. (1):

$$[\mathbf{A}_0 | \mathbf{A}_1 + \mathbf{H}\mathbf{G}] \begin{bmatrix} \mathbf{X}_{02} \\ \mathbf{X}_{12} \end{bmatrix} = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^*] \begin{bmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{bmatrix} \cdot \mathbf{R}_2' \neq \mathbf{A}_2' - \mathbf{A}_2. \tag{2}$$

This means adversary, given the simulated $\mathsf{pk}_1, \mathsf{pk}_2, \mathsf{rk}_{1\to 2}$, can easily tell whether they are from the real scheme or the simulated scheme. Thus, the security proof in this way [24] is not correct.

A straightforward fix would be to set $\mathbf{A}_2' = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^*] \begin{bmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{bmatrix} \cdot \mathbf{R}_2' + \mathbf{A}_2 = \mathbf{A}_0' \cdot \mathbf{R}_2' + \mathbf{A}_2$ so that Eqs. (1) and (2) match. But in this way it is not clear how to express $\mathbf{A}_2$ as $\mathbf{A}_0'\mathbf{R}$ for some small matrix $\mathbf{R}$, because it is not clear how to express $\mathbf{A}_2$ as $\mathbf{A}_0'\tilde{\mathbf{R}}$ for some small $\tilde{\mathbf{R}}$. Note that $\mathbf{R}$ serves as the secret key of pk_2 to simulate decryption queries. Consequently, it is not clear how the reduction can answer decryption queries as the previous approach. It seems that this construction/proof is facing a dilemma: either the reduction can answer the decryption queries but the re-encryption key can be distinguished, or the reduction can generate an indistinguishable re-encryption key but cannot answer the decryption queries.

Part II: Our New Construction for Single-Hop PRE. To overcome the dilemma, we consider a new matrix structure: the setup algorithm outputs a public matrix $\mathbf{A}$, and each user extends the previous matrix structure to be $\mathbf{A}_u = [\mathbf{A} | \mathbf{A}_1 + \mathbf{H}\mathbf{G} | \mathbf{A}_2 + \mathbf{H}_u\mathbf{G}]$, where $\mathbf{A}_1 = -\mathbf{A}\mathbf{R}_1, \mathbf{A}_2 = -\mathbf{A}\mathbf{R}_2$ and the matrices $\mathbf{R}_1, \mathbf{R}_2$ are the corresponding secret key. The shared matrix $\mathbf{A}$ offers a significant advantage for the simulation: the reduction can embed the LWE instance $\mathbf{A}^*$ as the public shared matrix, and then sets

$$\mathbf{A}_2' = [\mathbf{A}^* | -\mathbf{A}^*\mathbf{R}_1^*] \begin{bmatrix} \mathbf{X}_{00} \\ \mathbf{X}_{10} \end{bmatrix} \cdot \mathbf{R}_2' - \mathbf{A}^*\mathbf{R}_2^*.$$

This allows the reduction to express $\mathbf{A}_2'$ as $\mathbf{A}^*\mathbf{R}$ for some small and known matrix $\mathbf{R}$. Then the reduction can use this to simulate the decryption queries, while the Eq. (1) will match for the real scheme and the simulated scheme. Our modified construction achieves a relaxed re-encryption functionality in comparison to the construction proposed in [26], i.e. the re-encryption key can only transform well-formed ciphertexts into indistinguishable re-encrypted ciphertexts, but transformation of maliciously chosen cihpertexts can be distinguished if the adversary has the secret key of the target user. In Sect. 3, we present more detailed discussions and a simple transformation from the relaxed functionality to the "full-fledged" functionality using zero-knowledge proofs[2].

Part III: Extension to Multi-hop PRE. We further observe that the matrix structure in our construction can be extended to the multi-hop case with a slight modification. Interestingly, our scheme itself can support general network structures (for functionalities), yet our security proof (for CCA security), however, requires the structure of tree-structured networks (i.e. the adversary can only query re-encryption keys that form a tree among the users). If the adversary's queries form a general graph, then security of our scheme becomes unclear: we are not able to prove security under the current techniques, but there is no known attack, either. We leave it as an interesting open problem to determine whether our construction is secure under general network structures.

A technical reason for this phenomenon comes from the order of sampling for the simulation. We give a simple example for illustration: let there be three parties in the network, Users one, two, and three. It is easy for the reduction to simulate in the following order pk_1, $\mathsf{rk}_{1\to 2}$, pk_2, $\mathsf{rk}_{2\to 3}$, and then pk_3 *without* knowing a trapdoor of the LWE instance $\mathbf{A}^*$. The reduction, however, would get stuck if he needs to further generate $\mathsf{rk}_{1\to 3}$, which should be consistent with the already sampled pk_1 and pk_3. We recall that the reduction is able to check whether $\mathsf{rk}_{1\to 3}$ is consistent with pk_1 and pk_3 in both the real scheme and the simulated scheme (as Eq. (1)). Thus, the reduction must simulate such consistency as the real scheme. Even though there are techniques from the Ring-LWE [21,27] that allows sampling in the *reverse* order of pk_3, $\mathsf{rk}_{2\to 3}$, pk_2, $\mathsf{rk}_{1\to 2}$, pk_1, it does not help to solve the problem because the reduction still does not know how to generate $\mathsf{rk}_{1\to 3}$ after pk_1 and pk_3 are sampled, without a trapdoor of $\mathbf{A}^*$.

Part IV: Unified Model and Construction for Multi-hop PRS. An interesting observation from our multi-hop CCA-PRE construction is that it is also compatible with the lattice signature structure in the work of Boyen [9]. In particular, in that work, the signature scheme has the following structure: $[\mathbf{A}|\mathbf{B}_\mu]$, where is an encoded matrix for message μ. This message-dependent matrix can be extended to a similar structure similar to that in multi-hop PRE construction. Recall that prior PRS work [7,25] consider four scenarios for the security requirement. In each scenario, the adversary has access to a subset of oracles (signing, re-signing, re-key generation), and security requires that the adversary

[2] Under current techniques, zero knowledge proof systems based on pure lattices assumptions either require interactions or random oracles.

cannot forge a signature on behalf of honest users (whose secret keys are not at the adversary's hand). Our unified security model is based on the approach of multi-hop PRE model with necessary modifications to fit into the signature framework.

1.2 Related Works

Proxy Re-Encryption. As mentioned above, in recent years, there has been multiple PRE constructions achieving different security notions from different assumptions. In addition to the bi-directional PRE-CPA constructions [8,10], there is also some work [6,23] about building uni-directional PRE-CPA from various assumptions. For CCA-PRE construction, we only know how to construct single-hop scheme from bilinear group assumption as shown in work [26], and single-hop scheme from LWE assumption in the random oracle model as shown in [4]. Besides the above mentioned work, recently Nuñez et al. [31] proposed a nice framework capturing more fine-grained CCA-security of PRE, corresponding to the adversary's ability in the security experiment. Our multi-hop tag-based CCA-secure PRE construction described in the full version [19] can be categorized as $\mathsf{CCA}_{1,2}$ model in their paper regarding a special structure (trees).

Proxy Re-Signature. Bi-directional PRS was considered in the literature [7, 15]. The generation of re-key algorithm needs to take inputs both users' secret key. The more fine-grained notion, uni-directional PRS scheme was proposed in [25]. Shao et al. [34] cooked up a bilinear group based scheme (in random oracle model) that is insecure but proven secure in prior PRS model [7,25], but their result cannot be extended to the lattice setting.

2 Preliminaries

Notations. Let PPT denote probabilistic polynomial time. We use bold uppercase letters to denote matrices, and bold lowercase letters for vectors. We let λ be the security parameter and $[n]$ denote the set $\{1, ..., n\}$. We use $[\cdot|\cdot]$ to denote the concatenation of vectors or matrices, and use ℓ_∞ norm for the norms of all vectors and matrices used in our paper. We say a function $f(n)$ is *negligible* if it is $O(n^{-c})$ for all $c > 0$, and we $\mathsf{negl}(n)$ to denote a negligible function of n. Let X and Y be two random variables taking values in Ω. Define the statistical distance, denoted as $\Delta(X, Y)$ as

$$\Delta(X, Y) := \frac{1}{2} \sum_{s \in \Omega} |\mathbf{Pr}[X = s] - \mathbf{Pr}[Y = s]|$$

Let $X(\lambda)$ and $Y(\lambda)$ be ensembles of random variables. We say that X and Y are statistically close if $d(\lambda) := \Delta(X(\lambda), Y(\lambda))$ is a negligible function of λ. We say two ensembles $X(\lambda)$ and $Y(\lambda)$ are computationally indistinguishable (denoted as $X(\lambda) \approx Y(\lambda)$) if for every PPT distinguisher D, it holds that

$$|\mathbf{Pr}[D(X(\lambda)) = 1] - \mathbf{Pr}[D(Y(\lambda)) = 1]| = \mathsf{negl}(\lambda)$$

Lemma 2.1 ([1]). *Regarding the norm defined above, we have the following bounds:*

- *Let* $\mathbf{R} \in \{-1,1\}^{m\times m}$ *be chosen at random, then* $\mathbf{Pr}[||\mathbf{R}|| > 12\sqrt{2m}] < e^{-2m}$.
- *Let* $\mathbf{R}$ *be sampled from* $\mathcal{D}_{\mathbb{Z}^{m\times m},\sigma}$, *then we have* $\mathbf{Pr}[||\mathbf{R}|| > \sigma\sqrt{m}] < e^{-2m}$.

Randomness Extraction. We will use the following lemma to argue the indistinguishability of two different distributions, which is a generalization of the leftover hash lemma proposed by Dodis et al. [18].

Lemma 2.2 ([1]). *Suppose that* $m > (n+1)\log q + w(\log n)$. *Let* $\mathbf{R} \in \{-1,1\}^{m\times k}$ *be chosen uniformly at random for some polynomial* $k = k(n)$. *Let* $\mathbf{A}, \mathbf{B}$ *be matrix chosen randomly from* $\mathbb{Z}_q^{n\times m}, \mathbb{Z}_q^{n\times k}$ *respectively. Then, for all vectors* $\boldsymbol{w} \in \mathbb{Z}^m$, *the distribution* $(\mathbf{A}, \mathbf{AR}, \mathbf{R}^\mathsf{T}\boldsymbol{w})$ *is statistically close to distribution* $(\mathbf{A}, \mathbf{B}, \mathbf{R}^\mathsf{T}\boldsymbol{w})$.

Learning with Errors. The LWE problem was introduced by Regev [32], who showed that solving it *on the average* is as hard as (quantumly) solving several standard lattice problems *in the worst case*, when the error distribution is instantiated as discrete Gaussian distribution with proper parameters.

Definition 2.3 (LWE). *For an integer* $q = q(n) \geq 2$, *and an error distribution* $\chi = \chi(n)$ *over* $\mathbb{Z}_q$, *the learning with errors problem* $\mathsf{LWE}_{n,m,q,\chi}$ *is to distinguish between the distribution* $\{\mathbf{A}, \mathbf{A}^\mathsf{T}\boldsymbol{s} + \boldsymbol{x}\}$ *from distribution* $\{\mathbf{A}, \boldsymbol{u}\}$, *where* $\mathbf{A} \overset{\$}{\leftarrow} \mathbb{Z}_q^{n\times m}$, $\boldsymbol{s} \overset{\$}{\leftarrow} \mathbb{Z}_q^n$, $\boldsymbol{u} \overset{\$}{\leftarrow} \mathbb{Z}_q^m$, *and* $\boldsymbol{x} \leftarrow \chi^m$.

Small Integer Solution. The SIS problem was first suggested to be hard on average by Ajtai [2] and then formalized by Micciancio and Regev [30]. It is known to be as hard as certain worst-case problems (e.g., SIVP) in standard lattices [2,22,29,30].

Definition 2.4 (SIS). *For any* $n \in \mathbb{Z}$, *and any functions* $m = m(n), q = q(n), \beta = \beta(n)$, *the average-case Small Integer Solution problem* ($\mathsf{SIS}_{q,n,m,\beta}$) *is: Given an integer* q, *a matrix* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ *chosen uniformly at random and a real* $\beta \in \mathbb{R}$, *find a non-zero integer vector* $\boldsymbol{z} \in \mathbb{Z}^m - \{\mathbf{0}\}$, *such that* $\mathbf{A}\boldsymbol{z} = 0 \mod q$ *and* $||\boldsymbol{z}|| \leq \beta$.

G-Trapdoors and Sampling Algorithms. We briefly describe the main results in [28]: the definition of $\mathbf{G}$-trapdoor and the algorithms $\mathsf{Invert}^{\mathcal{O}}$ and $\mathsf{Sample}^{\mathcal{O}}$. Roughly speaking, a $\mathbf{G}$-trapdoor is a transformation, represented by a matrix $\mathbf{R}$ from a public matrix $\mathbf{A}$ to a special matrix $\mathbf{G}$. The formal definition is as follows:

Definition 2.5 ([28]). *Let* $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ *and* $\mathbf{G} \in \mathbb{Z}_q^{n\times w}$ *be matrices with* $m \geq w \geq n$. *A* $\mathbf{G}$*-trapdoor for* $\mathbf{A}$ *is a matrix* $\mathbf{R} \in \mathbb{Z}^{m-w} \times w$ *such that* $\mathbf{A}\left[\begin{smallmatrix}\mathbf{R}\\ \mathbf{I}\end{smallmatrix}\right] = \mathbf{HG}$ *for some invertible matrix* $\mathbf{H} \in \mathbb{Z}_q^{n\times n}$. *We refer to* $\mathbf{H}$ *as the tag or label of the trapdoor. The quality of the trapdoor is measured by its largest singular value* $s_1(\mathbf{R})$.

In order to embed matrix $\mathbf{G}$ into a uniformly looking matrix $\mathbf{A}$ together with a transformation $\mathbf{R}$, we should start with a uniform matrix $\mathbf{A}_0$ and a matrix $\mathbf{R}$, and construct $\mathbf{A} = [\mathbf{A}_0| - \mathbf{A}_0\mathbf{R} + \mathbf{HG}]$. For an appropriate chosen dimensions $(\mathbf{A}, \mathbf{AR})$ is negligible from uniformly random distribution by the Lattice-based Leftover Hash Lemma.

Following the work of Micciancio and Peikert [28], our scheme uses a special collection of elements defined over ring $\mathcal{R} = \mathbb{Z}_q[x]/(f(x))$, where $f(x) = x^n + f_{n-1}x^{n-1} + \cdots + f_0$ is a irreducible modulo every p dividing q. Since $\mathcal{R}$ is a free $\mathbb{Z}_q$-module of rank n, thus elements of $\mathcal{R}$ can be represented as vectors in $\mathbb{Z}_q$ relative to standard basis of monomials $1, x, ..., x^{n-1}$. Multiplication by any fixed element of $\mathcal{R}$ then acts as a linear transformation on $\mathbb{Z}_q^n$ according to the rule

$$x \cdot (a_0, ..., a_{n-1})^\mathsf{T} = (0, a_0, ..., a_{n-2})^\mathsf{T} - a_{n-1}(f_0, f_1, ..., f_{n-1})^\mathsf{T}$$

and so can be represented by an matrix in $\mathbb{Z}_q^{n\times n}$ relative to the standard basis. In other words, there is an injective ring homomorphism $h : \mathcal{R} \to \mathbb{Z}_q^{n\times n}$ that maps any $a \in \mathcal{R}$ to matrix $\mathbf{H} = h(a)$ representing multiplication by a. As introduced in [28], we need a very large set $\mathcal{U} = \{u_1, ..., u_l\}$ with the "unit differences" property: for any $i \neq j$, the difference $u_i - u_j \in \mathcal{R}^*$, and hence $h(u_i - u_j) = h(u_i) - h(u_j) \in \mathbb{Z}_q^{n\times n}$ is invertible.

Lemma 2.6 ([28]). *There is an efficient algorithm* $\mathsf{Sample}^{\mathcal{O}}(\mathbf{R}, \mathbf{A}', \mathbf{H}, \boldsymbol{u}, s)$, *where* $\mathbf{R}$ *is a* $\mathbf{G}$*-trapdoor for matrix* $\mathbf{A}$ *with invertible tag* $\mathbf{H}$, *a vector* $\boldsymbol{u} \in \mathbb{Z}^n$ *and an oracle* $\mathcal{O}$ *for Gaussian sampling over a desired coset* $\Lambda_q^{\boldsymbol{v}}(\mathbf{G})$. *It will output a vector drawn from a distribution within negligible statistical distance of* $\mathcal{D}_{\Lambda^{\boldsymbol{u}}(\mathbf{A}),s}$, *where* $\mathbf{A} = [\mathbf{A}'| - \mathbf{A}'\mathbf{R} + \mathbf{HG}]$.

In the following, we provide two extensions of the LWE inversion algorithms proposed by Micciancio and Peikert [28], which would be used in the security proof and scheme respectively.

- $\mathsf{Invert}^{\mathcal{O}}(\mathbf{R}_1, \mathbf{R}_2, \mathbf{A}, \boldsymbol{b})$: On input a vector $\boldsymbol{b} = \boldsymbol{s}^\mathsf{T}\mathbf{A} + \boldsymbol{e}^\mathsf{T}$, a matrix $\mathbf{A} = [\mathbf{A}_0| - \mathbf{A}_0\mathbf{R}_1 + \mathbf{H}_1\mathbf{G}| - \mathbf{A}_0\mathbf{R}_2 + \mathbf{H}_2\mathbf{G}]$ and its corresponding $\mathbf{G}$-trapdoor $\mathbf{R}_1, \mathbf{R}_2$ with invertible tag $\mathbf{H}_1, \mathbf{H}_2$, the algorithm first computes $\boldsymbol{b}' = \boldsymbol{b}^\mathsf{T}\left[\begin{smallmatrix}\mathbf{R}_1+\mathbf{R}_2\\ \mathbf{I}\\ \mathbf{I}\end{smallmatrix}\right]$, and then run the oracle $\mathcal{O}(\boldsymbol{b}')$ to get $(\boldsymbol{s}', \boldsymbol{e}')$. The algorithm outputs $\boldsymbol{s} = (\mathbf{H}_1 + \mathbf{H}_2)^{-1}\boldsymbol{s}'$ and $\boldsymbol{e} = \boldsymbol{b} - \boldsymbol{s}^\mathsf{T}\mathbf{A}$.
- $\mathsf{Invert}'^{\mathcal{O}}(\mathbf{R}_1, \mathbf{R}_2, \mathbf{A}, \boldsymbol{b})$: On input a vector $\boldsymbol{b} = \boldsymbol{s}^\mathsf{T}\mathbf{A} + \boldsymbol{e}^\mathsf{T}$, a matrix $\mathbf{A} = [\mathbf{A}_0| - \mathbf{A}_0\mathbf{R}_1| - \mathbf{A}_0\mathbf{R}_2 + \mathbf{H}_2\mathbf{G}]$ and its corresponding $\mathbf{G}$-trapdoor $\mathbf{R}_1, \mathbf{R}_2$ with invertible tag $\mathbf{H}_1, \mathbf{H}_2$, the algorithm first computes $\boldsymbol{b}' = \boldsymbol{b}^\mathsf{T}\left[\begin{smallmatrix}\mathbf{R}_1+\mathbf{R}_2\\ \mathbf{I}\\ \mathbf{I}\end{smallmatrix}\right]$, and then run the oracle $\mathcal{O}(\boldsymbol{b}')$ to get $(\boldsymbol{s}', \boldsymbol{e}')$. The algorithm outputs $\boldsymbol{s} = \mathbf{H}_2^{-1}\boldsymbol{s}'$ and $\boldsymbol{e} = \boldsymbol{b} - \boldsymbol{s}^\mathsf{T}\mathbf{A}$.

3 Proxy Re-Encryption: Syntax and Security Definitions

In this section, we first recall the syntax of single-hop PRE [26], and then we define a new variant of CCA-PRE security, i.e. tag-based CCA-PRE security that captures constructions associated with tags. However, for lattice-based constructions, our current technique cannot achieve the full-fledged PRE construction in that the re-encryption algorithm does not provide the full-fledged functionality in that it does not fully implement the regular re-encryption oracle which decrypts first, outputs $\perp$ if the decrypted value is invalid, and outputs a fresh ciphertext of the same message, otherwise. Our re-encryption algorithm guarantees the functionality when the input ciphertexts are well-formed, but if the input ciphertexts are not well-formed, the re-encryption algorithm is not able output $\perp$, yet it can only output re-encrypted ciphertexts that will be decrypted to $\perp$. This security notion is also known as security against *honest re-encryption attacks* (HRA), where the re-encryption oracle only re-encrypts honestly generated ciphertexts. The HRA model was defined in a recent work by Cohen [17], and it also identified many interesting scenarios captured by the HRA security. The work by Cohen [17] achieves the CPA + HRA security, and this work achieves a stronger notion – CCA + HRA security.

In fact, our relaxed functionality is not far from the full-fledged functionality if the input-ciphertext provider is required to prove the validity of the ciphertexts. We note that there exists an efficient lattice-based Σ protocol [5] with interaction, and we can further use the Fiat-Shamir transform [20] to achieve a NIZK proof system if a random oracle is assumed. Very recently, the work [12] constructed NIZK from FHE with circular security, which can be based on LWE with a certain circular security. Using this lattice-based NIZK, we can upgrade our security to the full-fledge CCA-PRE security. Therefore, if we further assume the required circular security on LWE, we are able to achieve the full-fledge CCA-PRE. We leave it as an interesting open problem to determine whether the circular security is inherent in achieving the full-fledge CCA-PRE.

The relaxed PRE security notion has already provided meaningful security guarantees and allowed a modular design to achieve the full-fledged functionality, e.g., the proxy additionally requests a proof of well-formness of the input ciphertexts. We believe that this notion deserves attention for the community.

3.1 Single-Hop PRE Syntax

We recall the syntax of uni-directional PRE, which can be regarded as a natural extension of bi-directional case defined in [10] and later studied in uni-directional scenario by Libert and Vergnaud [26]. The PRE scheme consists a tuple of PPT algorithms $(\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{ReKeyGen}, \mathsf{ReEnc})$, which can be defined as follows:

- $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ generates the public parameters pp.
- $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$ generates $(\mathsf{pk}, \mathsf{sk})$ for each user.
- $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}, \mu, i)$ encrypts a message μ at level $i \in \{1, 2\}$. The re-encryption can only operate on ciphertexts that are at level 1.
- $\mu' = \mathsf{Dec}(\mathsf{sk}, (\mathsf{ct}, i))$ decrypts a ciphertext ct.
- $\mathsf{rk}_{i \to j} \leftarrow \mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{pk}_j)$ computes the re-encryption key $\mathsf{rk}_{i \to j}$.

– $(\mathsf{ct}', 2) \leftarrow \mathsf{ReEnc}(\mathsf{rk}_{i\to j}, (\mathsf{ct}, 1))$ computes the re-encrypted ciphertext ct'. If the well-formedness of ciphertext ct is publicly verifiable, the algorithm should output "invalid" when ct is ill-formed.

Correctness. For correctness, we consider two cases for the PRE scheme: one for "fresh" ciphertexts generated by encryption algorithm, and the other for re-encryption ciphertexts generated by the re-encryption algorithm. We say that a single-hop PRE scheme is correct if the following holds.

– For any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, any $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, any message μ and level $i \in \{1, 2\}$, it holds that

$$\mathbf{Pr}[\mathsf{Dec}(\mathsf{sk}, \mathsf{Enc}(\mathsf{pk}, \mu, i)) = \mu] = 1 - \mathsf{negl}(\lambda)$$

– For any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$, any $(\mathsf{pk}_i, \mathsf{sk}_i), (\mathsf{pk}_j, \mathsf{sk}_j) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, any message μ, it holds that

$$\mathbf{Pr}[\mathsf{Dec}(\mathsf{sk}_j, \mathsf{ReEnc}(\mathsf{rk}_{i\to j}, \mathsf{ct})) = \mu] = 1 - \mathsf{negl}(\lambda)$$

where $\mathsf{ct} \leftarrow \mathsf{Enc}(\mathsf{pk}_i, \mu, 1), \mathsf{rk}_{i\to j} \leftarrow \mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{pk}_j)$.

3.2 Single-Hop PRE Security Definitions

In the security part, we first present the CCA-PRE definition proposed in [26] with minor modifications – in particular the definition of derivative in security model. Next, we describe a weaker security model considered in [24], whose restriction is: the re-encryption queries submitted by the adversary are only allowed between honest users. Then we propose an intermediate model, where the capability of re-encryption oracle is slightly weaker than its counterpart in [26]. Intuitively, we say a ciphertext is well-formed if it is an encryption of a message under the claimed public key. In the re-encryption oracle in [26], the well-formedness of ciphertext is public verifiable, i.e the verification only needs public keys. However, in our intermediate model, the verification needs the assistance of secret keys. Let $\mathcal{A}$ denote any PPT adversary, and Π be a PRE scheme. We define the notion of CCA-secure PRE in the uni-directional setting using the following experiment $\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA\text{-}PRE}}(1^\lambda)$, which describes the interaction between several oracles and an adversary $\mathcal{A}$. As we discussed before, we include public parameters pp in each user's public key pk and secret key sk, so we will omit pp in the description for simplicity. The experiment $\mathbf{Expt}_{\mathcal{A}}^{\mathsf{single}}(1^\lambda)$ consists of an execution of $\mathcal{A}$ with the following oracles with detail as follows:

– The challenger runs setup algorithm $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda)$ and initializes two empty sets $\mathcal{H} = \emptyset, \mathcal{C} = \emptyset$. Then he sends pp to adversary $\mathcal{A}$.
– Proceeding adaptively, adversary $\mathcal{A}$ has access to the following oracles:

Uncorrupted key generation oracle: Obtain a new key pair $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$. Send pk_i back to adversary $\mathcal{A}$, set the honest user set $\mathcal{H} = \mathcal{H} \cup \{i\}$ and pass the the tuple $(i, \mathsf{pk}_i, \mathsf{sk}_i)$ to re-encryption key generation oracle $\mathcal{O}_{\mathsf{ReKeyGen}}$ and decryption oracle $\mathcal{O}_{\mathsf{Dec}}$.

Corrupted key generation oracle: Obtain a new key pair $(\mathsf{pk}_i, \mathsf{sk}_i) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$. Send the key pair $(\mathsf{pk}_i, \mathsf{sk}_i)$ back to adversary $\mathcal{A}$, set the corrupted user set $\mathcal{C} = \mathcal{C} \cup \{i\}$ and pass the tuple $(i, \mathsf{pk}_i, \mathsf{sk}_i)$ to re-encryption key generation oracle $\mathcal{O}_{\mathsf{ReKeyGen}}$ and decryption oracle $\mathcal{O}_{\mathsf{Dec}}$.

Re-encryption key generation oracle $\mathcal{O}_{\mathsf{ReKeyGen}}$: On input an index pair (i, j) from the adversary, if the query (i, j) is made after accessing the challenge oracle, then output $\perp$ if $i = i^*$ and $j \in \mathcal{C}$. Otherwise, do the following:

- If the pair (i, j) is queried for the first time, the oracle returns a re-encryption key $\mathsf{rk}_{i \to j} \leftarrow \mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{pk}_j)$;
- else (the pair (i, j) has been queried before), the oracle returns the re-encryption key $\mathsf{rk}_{i \to j}$.

Re-encryption oracle $\mathcal{O}_{\mathsf{ReEnc}}$: On input $(i, j, (\mathsf{ct}, k))$, the oracle returns a special symbol $\perp$ if (ct, k) is not a well-formed first level ciphertext, or $j \in \mathcal{C}$ and $(i, \mathsf{ct}) = (i^*, \mathsf{ct}^*)$. Otherwise, it computes re-encrypted ciphertext $\mathsf{ct}' \leftarrow \mathsf{ReEnc}(\mathsf{rk}_{i \to j}, \mathsf{ct})$ and sends back $(\mathsf{ct}', 2)$.

Decryption oracle $\mathcal{O}_{\mathsf{Dec}}$: On input (i, ct), if $i \notin \mathcal{C} \cup \mathcal{H}$ or ct is not a valid ciphertext, then return a special symbol $\perp$. It also outputs a special symbol $\perp$ if (i, ct) is a *Derivative* (c.f. Definition 3.2) of the challenge pair (i^*, ct^*). Otherwise, it returns $\mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct})$ to adversary $\mathcal{A}$.

Challenge oracle: This oracle can be queried only once. On input (i^*, μ_0, μ_1), where $i^* \in \mathcal{H}$ and no re-encryption key from i^* to corrupted users $\mathcal{C}$ has been queried by adversary, the oracle chooses a bit $b \in \{0, 1\}$ and returns $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_{i^*}, \mu_b, 1)$ as the challenge ciphertext, and passes i^* to re-encryption key generation oracle $\mathcal{O}_{\mathsf{ReKeyGen}}$, and (i^*, ct^*) to re-encryption oracle $\mathcal{O}_{\mathsf{ReEnc}}$.

Decision oracle: This oracle can be queried only once. On input b' from adversary $\mathcal{A}$, the oracle outputs 1 if $b' = b$, and 0 otherwise.

The advantage of an adversary in the above experiment $\mathbf{Expt}_{\mathcal{A}}^{\mathsf{single}}(1^\lambda)$ is defined as $|\mathbf{Pr}[b' = b] - \frac{1}{2}|$.

Definition 3.1 (CCA-PRE Model). *A uni-directional PRE scheme is CCA-PRE secure if all* PPT *adversaries have at most a negligible advantage in experiment* $\mathbf{Expt}_{\mathcal{A}}^{single}(1^\lambda)$.

In our PRE construction, every ciphertext is associated with a tag u chosen randomly in the encryption algorithm, thus we call our security model *tag-based* CCA security. In [26], a pair (i, ct) is called *derivative* of the challenge ciphertext pair (i^*, ct^*) if $\mathsf{Dec}(\mathsf{ct}, \mathsf{sk}_i) \in \{\mu_0, \mu_1\}$, where $\{\mu_0, \mu_1\}$ are the challenge message pair. We achieve a slightly stronger notion of derivative as defined in the following

Definition 3.2 (Derivative). *A pair* $(i, (\mathsf{ct}, u))$ *is called derivative of the challenge ciphertext pair* $(i^*, (\mathsf{ct}^*, u^*))$ *if* $u = u^*$.

Remark 3.3. It is obvious to see that tag-based CCA security is stronger than CCA1 security (where the adversary cannot access the decryption oracle after the challenge ciphertext), and is slightly weaker than CCA2 security. This relaxation is meaningful and can be nearly the best we can achieve if we further require the property of *unlinkability* for re-encrypted ciphertexts. That is, if we want the re-encrypted algorithm to produce statistically indistinguishable ciphertexts, i.e. the re-encrypted ciphertexts are almost identically distributed as fresh ones, then arguably it is not possible to achieve CCA2 security, because the decryption oracle cannot distinguish a re-encryption of challenge ciphertext from a fresh ciphertext, so an adversary can easily break the security game by querying the decryption oracle with a re-encrypted ciphertext of the challenge ciphertext. For tag-based schemes, where the tag remains the same for re-encrypted ciphertexts, we can ensure that the challenge ciphertext will not be decrypted by the decryption oracle due to derivative definition (see Definition 3.2). The tag-based CCA security guarantees the challenge ciphertext remains hidden, even if the adversary can obtain decryptions of ciphertexts with other tags.

Remark 3.4. Our re-encryption oracle only re-encrypts well-form ciphertexts. This is explicitly defined as honest re-encryption attacks (HRA) by Cohen [17]. The formulation of this work is slightly different from that of the work by Cohen [17], but the two formulations have the same spirit.

The above security model only captures the CCA security of ciphertexts on the first level. We also present the CCA security of ciphertexts on the second level. Since the challenge ciphertext is on the second level, which means it cannot be further re-encrypted to ciphertext under other public keys, so there is no need to restrict the re-encryption queries regarding the challenge ciphertext. We highlight the difference comparing to security model of first level ciphertexts in the following definition.

Definition 3.5. (Second-Level Security). *The difference of experiment between second-level security and the security definition in Definition 3.1 are below:*

- *In challenge oracle: the oracle returns* $\mathsf{ct}^* \leftarrow \mathsf{Enc}(\mathsf{pk}_{i^*}, \mu_b, 2)$ *as the challenge ciphertext.*
- *The re-encryption oracle* $\mathcal{O}_{\mathsf{ReEnc}}$ *does not need to check whether the queried tuple is the same as challenge ciphertext.*

Definition 3.6. (PRE with Relaxed Functionality). *A PRE scheme with a relaxed functionality if the re-encryption algorithm outputs statistically close to the distribution of fresh ciphertexts of the second level when the input ciphertexts are well-formed. That is, if* $(\mathsf{ct}, 1)$ *is a well-formed ciphertext, then* $\mathsf{ReEnc}(\mathsf{rk}_{i \to j}, (\mathsf{ct}, 1))$ *is statistically close to* $(\mathsf{ct}', 2) \leftarrow \mathsf{Enc}(\mathsf{pk}_j, \mathsf{Dec}(\mathsf{ct}, 1), 2)$*. If the input ciphertexts are not well-formed, then only* $\mathsf{Dec}(\mathsf{sk}_j, (\mathsf{ct}', 2)) = \perp$ *is guaranteed.*

Remark 3.7. *As we argued above, the relaxed functionality does not completely implement the re-encryption oracle* $\mathcal{O}_{\mathsf{ReEnc}}$ *as in the above definition. The difference can be bridged by a crypto proof system, (either interactively or* *non-interactively) assuming the input ciphertext is associated with a proof. We present the formal description of this idea in the full version of this paper.*

In our construction, we do not allow querying the relaxed functionality directly with arbitrary input ciphertexts re-encrypted to a corrupted party, e.g., invalid input ciphertexts chosen by the adversary to some corrupted Party j. As the transformation can leak the re-encryption key, if the adversary corrupts Party j and can obtain a re-encryption key $\mathsf{rk}_{i\to j}$, then he can easily break the security of pk_i.

We note that the the CCA model of [24] is weaker than the model considered in this paper. In particular, the model [24] has the following restrictions: the re-encryption key queries (or re-encryption queries) submit by adversary $\mathcal{A}$ are restricted among honest users (we ignore the re-encryption queries within corrupted users, since adversary can generate by himself).

4 Single-Hop Tag-Based CCA-Secure PRE Construction

In this section, we present our construction of single-hop PRE. The PRE system has message space $\{0,1\}^{nk}$, which we map bijectively to the cosets of $\Lambda/2\Lambda$ for $\Lambda = \Lambda(\mathbf{G}^t)$ via some encoding function encode that is efficient to evaluate and invert. In particular, letting $\mathbf{S} \in \mathbb{Z}^{nk\times nk}$ be any basis of Λ, we can map $\boldsymbol{\mu} \in \{0,1\}^{nk}$ to $\mathsf{encode}(\boldsymbol{\mu}) = \mathbf{S}\boldsymbol{\mu} \in \mathbb{Z}^{nk}$. The PRE scheme ($\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{ReKeyGen}, \mathsf{ReEnc}$) can be described as follows:

- $\mathsf{Setup}(1^\lambda, 1^N)$: The global setup algorithm set the lattice parameter (n, k, q, s). Then it randomly selects a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times nk}$, and outputs the public parameter $\mathsf{pp} = (\mathbf{A}, n, m, q, s)$.
- $\mathsf{KeyGen}(\mathsf{pp})$: The key generation algorithm for i-th user chooses random matrices $\mathbf{R}_{i1}, \mathbf{R}_{i2} \leftarrow \mathcal{D}_{\mathbb{Z}^{nk\times nk},s}$, letting $\mathbf{A}_{i1} = \mathbf{A}\mathbf{R}_{i1} \bmod q$ and $\mathbf{A}_{i2} = \mathbf{A}\mathbf{R}_{i2} \bmod q$. The public key is $\mathsf{pk}_i = \mathbf{A}_i = [\mathbf{A}| - \mathbf{A}_{i1}| - \mathbf{A}_{i2}]$, and the secret key is $\mathsf{sk}_i = [\mathbf{R}_{i1}|\mathbf{R}_{i2}]$.
- $\mathsf{Enc}(\mathsf{pk}_i, \boldsymbol{\mu}, \ell)$: The encryption algorithm does
  - If $\ell = 1$, choose non-zero $u \leftarrow \mathcal{U}$ and let the message/level-dependent matrix
    $$\mathbf{A}_{i,u,l} = [\mathbf{A}| - \mathbf{A}_{i1} + h(\ell)\mathbf{G}| - \mathbf{A}_{i2} + h(u)\mathbf{G}]$$
    Choose $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n, \boldsymbol{e}_0, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{D}_{\mathbb{Z},s}^{nk}$. Let
    $$\boldsymbol{b}^\mathsf{T} = (\boldsymbol{b}_0, \boldsymbol{b}_1, \boldsymbol{b}_2) = 2(\boldsymbol{s}^\mathsf{T}\mathbf{A}_{i,u,\ell} \bmod q) + \boldsymbol{e}^\mathsf{T} + (0, 0, \mathsf{encode}(\boldsymbol{\mu})^\mathsf{T}) \bmod 2q$$
    where $\boldsymbol{e} = (\boldsymbol{e}_0, \boldsymbol{e}_1, \boldsymbol{e}_2)$. Output the ciphertext $\mathsf{ct} = (u, \boldsymbol{b}, 1)$.
  - If $\ell = 2$, the algorithm uses the same procedure to encrypt the message, except it chooses error $\boldsymbol{e}_0, \boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{D}_{\mathbb{Z},s'}^{nk}$, and outputs $\mathsf{ct} = (u, \boldsymbol{b}, 2)$.

- $\mathsf{Dec}(\mathsf{sk}_i, \mathsf{ct})$: The decryption algorithm
  1. If ct does not parse or $u = 0$, output $\perp$. Otherwise, reconstruct the message/level-dependent matrix $\mathbf{A}_{i,u,\ell}$
  $$\mathbf{A}_{i,u,l} = [\mathbf{A}| - \mathbf{A}_{i1} + h(\ell)\mathbf{G}| - \mathbf{A}_{i2} + h(u)\mathbf{G}]$$
  Call $\mathsf{Invert}^{\mathcal{O}}([\mathbf{R}_{i1}|\mathbf{R}_{i2}], \mathbf{A}_u, \boldsymbol{b} \bmod q)$ to get values $\boldsymbol{z} \in \mathbb{Z}_q^n$ and $\boldsymbol{e} = (\boldsymbol{e}_0, \boldsymbol{e}_1, \boldsymbol{e}_2)$ for which $\boldsymbol{b} = \boldsymbol{z} + \boldsymbol{e} \bmod q$. If the algorithm Invert fail for any reason, output $\perp$.
  2. Check the length of the obtained error vectors, namely if $||\boldsymbol{e}_0|| \geq s'\sqrt{m}$ or $||\boldsymbol{e}_i|| \geq s'^2 m$, for $i = 1, 2$, output $\perp$.
  3. Let $\boldsymbol{v} = \boldsymbol{b} - \boldsymbol{e}$, and parse $\boldsymbol{v} = (\boldsymbol{v}_0, \boldsymbol{v}_1, \boldsymbol{v}_2)$. If $\boldsymbol{v}_0 \notin 2\Lambda(\mathbf{A}^{\mathsf{T}})$, output $\perp$. Finally, output
  $$\mathsf{encode}^{-1}(\boldsymbol{v}^{\mathsf{T}} \begin{bmatrix} \mathbf{R}_{i1} & \mathbf{R}_{i2} \\ \mathbf{I} & 0 \\ 0 & \mathbf{I} \end{bmatrix} \bmod 2q) \in \{0,1\}^{nk}$$
  if it exists, otherwise output $\perp$.
- $\mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{sk}_i, \mathsf{pk}_j)$: The re-encryption key generation algorithm does:
  1. Use $\mathsf{sk}_i = [\mathbf{R}_{i1}|\mathbf{R}_{i2}]$ to run extended sampling algorithm $\mathsf{Sample}^{\mathcal{O}}$ to sample $\mathbf{X}_{01}, \mathbf{X}_{02}, \mathbf{X}_{11}, \mathbf{X}_{12} \in \mathbb{Z}^{nk \times nk}$ such that
  $$[\mathbf{A}|-\mathbf{A}_{i1} + h(1)\mathbf{G}|-\mathbf{A}_{i2} + \mathbf{B}] \begin{bmatrix} \mathbf{I} & \mathbf{X}_{01} & \mathbf{X}_{02} \\ 0 & \mathbf{X}_{11} & \mathbf{X}_{12} \\ 0 & 0 & \mathbf{I} \end{bmatrix} = [\mathbf{A}| - \mathbf{A}_{j1} + h(2)\mathbf{G}| - \mathbf{A}_{j2} + \mathbf{B}]$$
  for any matrix $\mathbf{B} \in \mathbb{Z}^{n \times nk}$.
  2. Output the re-encryption key
  $$\mathsf{rk}_{i \to j} = \{\mathbf{X}_{01}, \mathbf{X}_{02}, \mathbf{X}_{11}, \mathbf{X}_{12}\}$$
- $\mathsf{ReEnc}(\mathsf{rk}_{i \to j}, \mathsf{ct})$: First the re-encryption algorithm parses $\mathsf{ct} = (u, b, \ell)$ outputs a special symbol $\perp$ if $\ell = 2$. Otherwise, it computes
$$\boldsymbol{b}^{\mathsf{T}} \cdot \mathsf{rk}_{i \to j} = \boldsymbol{s}^{\mathsf{T}}[\mathbf{A}| - \mathbf{A}_{j1} + h(1)\mathbf{G}| - \mathbf{A}_{j2} + h(u)\mathbf{G}] + \boldsymbol{e}'^{\mathsf{T}} + \widetilde{\boldsymbol{e}^{\mathsf{T}}} + (0, 0, \mathsf{encode}(\mu)^{\mathsf{T}})$$
  where $\boldsymbol{e}' = (\boldsymbol{e}'_0, \boldsymbol{e}'_1, \boldsymbol{e}'_2), \widetilde{\boldsymbol{e}} \leftarrow \mathcal{D}_{\mathbb{Z}^{3nk}, s'}$, and
$$\boldsymbol{e}'_0 = \boldsymbol{e}_0, \qquad \boldsymbol{e}'_1 = \boldsymbol{e}_0\mathbf{X}_{01} + \boldsymbol{e}_1\mathbf{X}_{11}, \qquad \boldsymbol{e}'_2 = \boldsymbol{e}_0\mathbf{X}_{02} + \boldsymbol{e}_1\mathbf{X}_{12} + \boldsymbol{e}_2 \tag{3}$$
  Then, it outputs $\mathsf{ct}' = (u, \boldsymbol{b}', 2)$.

Parameter Setting. In this part, we set the lattice parameters used in our construction. The correctness proof of our construction can be found in full version [19]. $\mathbf{G} \in \mathbb{Z}_q^{n \times nk}$ is a gadget matrix for $q = \mathsf{poly}(n), n = \mathsf{poly}(\lambda)$ and $k = O(\log q) = O(\log n)$. For matrix $\mathbf{A} \in \mathbf{Z}_q^{n \times m}$ in the public parameters and secret keys $\mathbf{R} \leftarrow \mathcal{D}$, we set $m = O(nk)$ and $\mathcal{D} = \mathcal{D}^{m \times nk}_{\mathbf{Z}, w(\sqrt{\log n})}$ respectively. We set the deviation s for discrete Gaussian distribution used in security proof to be $s = \omega(\sqrt{\log n})\sqrt{m}$, and parameter for level 2 error is $s' = s\sqrt{m}$. For the error rate α in the LWE assumption, we set sufficiently large $1/\alpha = O(nk) \cdot w(\sqrt{\log n})$.

5 Proxy Re-Signature with Selectively Chosen Tag

In this section, we present the syntax and our construction of PRS.

5.1 Syntax and Correctness Definition

We first recall the syntax and security definition of PRS in [7,25], then propose a simpler and unified security model that captures the security requirements. Our model adapts the same spirit of the prior security model of proxy re-encryption [10,26], with necessary modifications to fit into the signature framework. We also compare our new notion with the previous security model in [7,25].

Let $L = L(\lambda)$ denotes the maximum level the PRS system supports. The scheme $\Sigma = (\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Verify}, \mathsf{ReKeyGen}, \mathsf{ReSign})$ is described as follows:

– $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$ generates the public parameter pp for the whole system.
– $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp}, i)$ generates $(\mathsf{pk}_i, \mathsf{sk}_i)$ for user i.
– $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}_i, \mu, \kappa)$ computes a signature σ for μ at level κ.
– $\mathsf{Verify}(\mathsf{pk}_i, \sigma, \mu, \kappa)$ outputs 1 (accept) or 0 (reject).
– $\mathsf{rk}^{\kappa}_{i\to j} \leftarrow \mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{pk}_j, \mathsf{sk}_j, \kappa)$ computes a re-signing key from the i-th user at level κ to the j-th user at level $\kappa + 1$.
– $\mathsf{ReSign}(\mathsf{rk}^{\kappa}_{i\to j}, \mu, \sigma, \kappa)$ computes a re-signature σ' under pk_j if $\mathsf{Verify}(\mathsf{pk}_i, \sigma, \mu, \kappa) = 1$, or $\perp$ otherwise.

Correctness. For all security parameter λ, any $\mathsf{pp} \leftarrow \mathsf{Setup}(1^\lambda, 1^L)$, all couples of secret/public key pairs $(\mathsf{sk}_i, \mathsf{pk}_i), (\mathsf{sk}_j, \mathsf{pk}_j)$ generated by $\mathsf{KeyGen}(\mathsf{pp})$, for any message μ and $\kappa \in [L]$, it holds that

$$\mathsf{Verify}(\mathsf{pk}_i, \mu, \kappa, \mathsf{Sign}(\mathsf{sk}_i, \mu, \kappa)) = 1$$

$$\mathsf{Verify}(\mathsf{pk}_j, \kappa + 1, \mu, \sigma) = 1$$

where $\sigma = \mathsf{ReSign}(\mathsf{rk}^{\kappa}_{i\to j}, \mu, \kappa, \mathsf{Sign}(\mathsf{sk}_i, \mu, \kappa))$ and $\mathsf{rk}^{\kappa}_{i\to j} \leftarrow \mathsf{ReKeyGen}(\mathsf{pk}_i, \mathsf{pk}_j, \mathsf{sk}_j, \kappa)$.

5.2 Our PRS Construction

Now we present our PRS construction and its security proof sketch. For simplicity, we first present the scheme with security regarding a selective chosen tag, where in the security experiment, the adversary needs to commit to the challenge tag before obtaining public parameters and public keys. In the full version, we also describe how to modify our construction, slightly, to achieve security for adaptively chosen tags. Let the message space be $\mathcal{M} = \mathbb{Z}_q$, and the tag space be $\mathcal{T} = \mathbb{Z}_q$. The description is the following:

– $\mathsf{Setup}(1^\lambda, 1^L)$: The setup algorithm sets the lattice parameters (n, q, m, s), then randomly chooses a matrix $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ and vectors $\boldsymbol{b}, \boldsymbol{v} \in \mathbb{Z}_q^n$. Output the public parameter $\mathsf{pp} = (\mathbf{A}, \boldsymbol{b}, \boldsymbol{v}, q, n, m)$.

- $\mathsf{KeyGen}(\mathsf{pp})$: The key generation algorithm computes $(\mathsf{pk}_i, \mathsf{sk}_i)$ as follows:
  1. Sample two small matrices $\mathbf{R}_{i1}, \mathbf{R}_{i2}$ from discrete Gaussian distribution $\mathcal{D}_{\mathbb{Z}^{m\times m},s}$.
  2. Compute $\mathbf{A}_i = \mathbf{A}\cdot\mathbf{R}_{i1} \bmod q$ and $\mathbf{A}'_i = \mathbf{A}\cdot\mathbf{R}_{i2} \bmod q$.
  3. The public key pk_i and secret key sk_i for i-th user is
$$\mathsf{pk}_i = (\mathbf{A}_i, \mathbf{A}'_i), \qquad \mathsf{sk}_i = (\mathbf{R}_{i1}, \mathbf{R}_{i2})$$
- $\mathsf{Sign}(\mathsf{pp}, \mathsf{sk}_i, \mu, \kappa)$: The signing algorithm does:
  1. Randomly select a non-zero tag $t \in \mathbb{Z}_q^*$, and define the signing matrix to be
$$\mathbf{F}_{t,i,\kappa} = [\mathbf{A}|\mathbf{A}_i + h(\kappa)\mathbf{G}|\mathbf{A}'_i + t\mathbf{G}]$$
  2. Sample a vector $\boldsymbol{r}_1 \leftarrow \mathcal{D}_{\mathbb{Z}^m,s}$, then sample vector $(\boldsymbol{r}_0, \boldsymbol{r}_2) \in \mathbb{Z}^{2m}$, using
$$(\boldsymbol{r}_0, \boldsymbol{r}_2) \leftarrow \mathsf{Sample}^{\mathcal{O}}(\mathbf{A}, t\mathbf{G}, \mathbf{R}_{i2}, \mathbf{T}_{\mathbf{G}}, \boldsymbol{b} + \mu\boldsymbol{v} - (\mathbf{A}'_i + h(\kappa)\mathbf{G})\boldsymbol{r}_1, s)$$
     Therefore, it holds that $\mathbf{F}_{t,i,\kappa}\cdot\sigma = \boldsymbol{b} + \mu\boldsymbol{v} \bmod q$, where $\sigma = (\boldsymbol{r}_0, \boldsymbol{r}_1, \boldsymbol{r}_2)$.
  3. Output the signature (σ, t, i, κ).
- $\mathsf{Verify}(\mathsf{pp}, \mathsf{pk}_i, \mu, (\sigma, t, i, \kappa))$: The verification algorithm dose:
  1. Parse the signature tuple as $\sigma = (\boldsymbol{r}_0, \boldsymbol{r}_1, \boldsymbol{r}_2)$, tag t, user index i and level index κ, then first check the norm of $|\sigma| = |(\boldsymbol{r}_0, \boldsymbol{r}_1, \boldsymbol{r}_2)|$. Output 0 if $|\sigma| \geq B$.
  2. Reconstruct the signing matrix
$$\mathbf{F}_{t,i,\kappa} = [\mathbf{A}|\mathbf{A}_i + h(\kappa)\mathbf{G}|\mathbf{A}'_i + t\mathbf{G}]$$
     and output 1 if $\mathbf{F}_{t,i,\kappa}\cdot\sigma = \boldsymbol{b} + \mu\boldsymbol{v}$, otherwise output 0.
- $\mathsf{ReKeyGen}(\mathsf{pk}_i, (\mathsf{sk}_j, \mathsf{pk}_j), \kappa)$: The re-signing key generation:
  1. Sample small matrices $(\mathbf{X}_{01}, \mathbf{X}_{11}, \mathbf{X}_{02}, \mathbf{X}_{12})$, using
$$(\mathbf{X}_{01}, \mathbf{X}_{11}) \leftarrow \mathsf{Sample}^{\mathcal{O}}(\mathbf{A}, h(\kappa+1)\mathbf{G}, \mathbf{R}_{j1}, \mathbf{T}_{\mathbf{G}}, \mathbf{A}_i + h(\kappa)\mathbf{G}, s),$$
$$(\mathbf{X}_{02}, \mathbf{X}_{12}) \leftarrow \mathsf{Sample}^{\mathcal{O}}(\mathbf{A}, h(\kappa+1)\mathbf{G}, \mathbf{R}_{j1}, \mathbf{T}_{\mathbf{G}}, \mathbf{A}'_i - \mathbf{A}'_j, s)$$
     Therefore it holds that
$$[\mathbf{A}|\mathbf{A}_j + h(\kappa+1)\mathbf{G}|\mathbf{A}'_j + t\mathbf{G}]\begin{bmatrix}\mathbf{I} & \mathbf{X}_{01} & \mathbf{X}_{02}\\ 0 & \mathbf{X}_{11} & \mathbf{X}_{12}\\ 0 & 0 & \mathbf{I}\end{bmatrix} = [\mathbf{A}|\mathbf{A}_i + h(\kappa)\mathbf{G}|\mathbf{A}'_i + t\mathbf{G}]$$
  2. Output the re-signing key $\mathsf{rk}^{\kappa}_{i\to j} = (\mathbf{X}_{01}, \mathbf{X}_{02}, \mathbf{X}_{11}, \mathbf{X}_{12})$.
- $\mathsf{ReSign}(\mathsf{rk}^{\kappa}_{i\to j}, (\sigma, t, i, \kappa), \mu, pk_i)$: The re-signing algorithm does:
  1. First parse $\sigma = (\boldsymbol{r}_0, \boldsymbol{r}_1, \boldsymbol{r}_2)$. Output $\perp$ if $\mathsf{Verify}(\mathsf{pp}, \mathsf{pk}_i, \mu, (\sigma, t, i, \kappa)) = 0$.
  2. Otherwise, output the re-signature tuple $(\sigma', t, j, \kappa+1)$, where $\sigma' = \mathsf{rk}^{\kappa}_{j\to j}\cdot\sigma$.

5.3 Parameter Setting

Let λ be the security parameter. For $L = \mathsf{polylog}(\lambda)$ maximum allowed re-signing, we set the parameters of our scheme based on standard SIS assumption as $q = n^{O(L)}, n = \mathsf{poly}(\lambda), L = \mathsf{polylog}(\lambda), m = O(n \log q)$. To ensure the SIS instance has a worst-case lattice reduction as shown in [30], i.e. $q \geq \beta\omega(\sqrt{n \log n})$, we set $\beta = \mathsf{polylog}(n)$. In order to achieve indistinguishability between real execution and reduction, the Gaussian parameter is set to be $s = \omega(\sqrt{\log n})$. As a signature produced by algorithm Sign has the size of $O(s\sqrt{m})$, and after each re-signing, the size grows at the rate of $O(sm)$, so we set parameter used in verification to be $B = \omega(2^L)$. Our PRS construction can support $L = \mathsf{poly}(\lambda)$-hop using subexponential SIS assumption.

6 Conclusion

In this work, we first point out a subtle error in work [24] and then showed how to construct single-hop PRE that is secure in our new model, tag-based CCA security. We then extend our security definition and construction to the multi-hop scenario, as elaborated in the full version [19]. Lastly, we propose a simpler and unified security model for PRS which captures more dynamic settings, then give a construction based on SIS assumption. Due to the space constrain, the security definition and proof of PRS are in the full version of this paper [19].

Acknowledgement. Xiong Fan is supported by NSF Award CNS-1561209. Feng-Hao Liu is supported by the NSF Award CNS-1657040. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the sponsors.

References

1. Agrawal, S., Boneh, D., Boyen, X.: Efficient lattice (H)IBE in the standard model. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 553–572. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_28
2. Ajtai, M.: Determinism versus non-determinism for linear time RAMs (extended abstract). In 31st ACM STOC, pp. 632–641. ACM Press, May 1999
3. Alwen, J., et al.: On the relationship between functional encryption, obfuscation, and fully homomorphic encryption. In: Stam, M. (ed.) IMACC 2013. LNCS, vol. 8308, pp. 65–84. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-45239-0_5
4. Aono, Y., Boyen, X., Phong, L.T., Wang, L.: Key-private proxy re-encryption under LWE. In: Paul, G., Vaudenay, S. (eds.) INDOCRYPT 2013. LNCS, vol. 8250, pp. 1–18. Springer, Cham (2013). https://doi.org/10.1007/978-3-319-03515-4_1
5. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold FHE. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_29

6. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. In: NDSS 2005. The Internet Society, February 2005
7. Ateniese, G., Hohenberger, S.: Proxy re-signatures: new definitions, algorithms, and applications. In: Atluri, V., Meadows, C., Juels, A. (eds.) ACM CCS 2005, pp. 310–319. ACM Press, November 2005
8. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998). https://doi.org/10.1007/BFb0054122
9. Boyen, X.: Lattice mixing and vanishing trapdoors: a framework for fully secure short signatures and more. In: Nguyen, P.Q., Pointcheval, D. (eds.) PKC 2010. LNCS, vol. 6056, pp. 499–517. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13013-7_29
10. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: Ning, P., De Capitani di Vimercati, S., Syverson, P.F. (eds.) ACM CCS 2007, pp. 185–194. ACM Press, October 2007
11. Canetti, R., Lin, H., Tessaro, S., Vaikuntanathan, V.: Obfuscation of probabilistic circuits and applications. In: Dodis, Y., Nielsen, J.B. (eds.) TCC 2015. LNCS, vol. 9015, pp. 468–497. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46497-7_19
12. Canetti, R., Lombardi, A., Wichs, D.: Non-interactive zero knowledge and correlation intractability from circular-secure FHE. Cryptology ePrint Archive, Report 2018/1248 (2018). https://eprint.iacr.org/2018/1248
13. Chandran, N., Chase, M., Liu, F.-H., Nishimaki, R., Xagawa, K.: Re-encryption, functional re-encryption, and multi-hop re-encryption: a framework for achieving obfuscation-based security and instantiations from lattices. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 95–112. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_6
14. Chandran, N., Chase, M., Vaikuntanathan, V.: Functional re-encryption and collusion-resistant obfuscation. In: Cramer, R. (ed.) TCC 2012. LNCS, vol. 7194, pp. 404–421. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-28914-9_23
15. Chow, S.S.M., Phan, R.C.-W.: Proxy re-signatures in the standard model. In: Wu, T.-C., Lei, C.-L., Rijmen, V., Lee, D.-T. (eds.) ISC 2008. LNCS, vol. 5222, pp. 260–276. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-85886-7_18
16. Chow, S.S.M., Weng, J., Yang, Y., Deng, R.H.: Efficient unidirectional proxy re-encryption. In: Bernstein, D.J., Lange, T. (eds.) AFRICACRYPT 2010. LNCS, vol. 6055, pp. 316–332. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-12678-9_19
17. Cohen, A.: What about Bob? the inadequacy of CPA security for proxy reencryption. In: Lin, D., Sako, K. (eds.) PKC 2019. LNCS, vol. 11443, pp. 287–316. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-17259-6_10
18. Dodis, Y., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. In: Cachin, C., Camenisch, J.L. (eds.) EUROCRYPT 2004. LNCS, vol. 3027, pp. 523–540. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-24676-3_31
19. Fan, X., Liu, F.-H.: Proxy re-encryption and re-signatures from lattices. IACR Cryptology ePrint Archive **2017**, 456 (2017)

20. Fiat, A., Shamir, A.: How to prove yourself: practical solutions to identification and signature problems. In: Odlyzko, A.M. (ed.) CRYPTO 1986. LNCS, vol. 263, pp. 186–194. Springer, Heidelberg (1987). https://doi.org/10.1007/3-540-47721-7_12
21. Garg, S., Gentry, C., Halevi, S.: Candidate multilinear maps from ideal lattices. In: Johansson, T., Nguyen, P.Q. (eds.) EUROCRYPT 2013. LNCS, vol. 7881, pp. 1–17. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-38348-9_1
22. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Ladner, R.E., Dwork, C. (eds.) 40th ACM STOC, pp. 197–206. ACM Press, May 2008
23. Hohenberger, S., Rothblum, G.N., Shelat, A., Vaikuntanathan, V.: Securely obfuscating re-encryption. In: Vadhan, S.P. (ed.) TCC 2007. LNCS, vol. 4392, pp. 233–252. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-70936-7_13
24. Kirshanova, E.: Proxy re-encryption from lattices. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 77–94. Springer, Heidelberg (2014). https://doi.org/10.1007/978-3-642-54631-0_5
25. Libert, B., Vergnaud, D.: Multi-use unidirectional proxy re-signatures. In: Ning, P., Syverson, P.F., Jha, S. (eds.) ACM CCS 2008, pp. 511–520. ACM Press, October 2008
26. Libert, B., Vergnaud, D.: Unidirectional chosen-ciphertext secure proxy re-encryption. In: Cramer, R. (ed.) PKC 2008. LNCS, vol. 4939, pp. 360–379. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78440-1_21
27. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Gilbert, H. (ed.) EUROCRYPT 2010. LNCS, vol. 6110, pp. 1–23. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13190-5_1
28. Micciancio, D., Peikert, C.: Trapdoors for lattices: simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29011-4_41
29. Micciancio, D., Peikert, C.: Hardness of SIS and LWE with small parameters. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013. LNCS, vol. 8042, pp. 21–39. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40041-4_2
30. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on Gaussian measures. In: 45th FOCS, pp. 372–381. IEEE Computer Society Press, October 2004
31. Nunez, D., Agudo, I., Lopez, J.: A parametric family of attack models for proxy re-encryption. In: 2015 IEEE 28th Computer Security Foundations Symposium (CSF), pp. 290–301. IEEE (2015)
32. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) 37th ACM STOC, pp. 84–93. ACM Press, May 2005
33. Shao, J., Cao, Z., Liu, P.: CCA-Secure PRE scheme without random oracles. Cryptology ePrint Archive, Report 2010/112 (2010). http://eprint.iacr.org/2010/112
34. Shao, J., Feng, M., Zhu, B., Cao, Z., Liu, P.: The security model of unidirectional proxy re-signature with private re-signature key. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 216–232. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14081-5_14