PREs with HRA Security and Key Privacy Based on Standard LWE Assumptions

Yang Wang[1,2]✉[0000−0001−9274−8195], Yanmin Zhao[3]✉, and Mingqiang Wang[1,2]✉

[1] School of Mathematics, Shandong University, Jinan, Shandong, 250100, P.R. China
[2] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, Shandong, 250100, P.R. China
[3] Department of Computer Science, The University of Hong Kong, P.R. China
wyang1114@sdu.edu.cn
ymzhao@cs.hku.hk
wangmingqiang@sdu.edu.cn

Abstract. Proxy re-encryption (PRE) schemes, which nicely solve the problem of delegating decryption rights, enable a semi-trusted proxy to transform a ciphertext encrypted under one key into a ciphertext of the same message under another arbitrary key. For a long time, the semantic security of PREs is quite similar to that of public key encryption (PKE) schemes. Cohen first pointed out the insufficiency of the security under chosen-plaintext attacks (CPA) of PREs in PKC 2019, and proposed a *strictly stronger* security notion, named security under honest re-encryption attacks (HRA), of PREs. Surprisingly, a few PREs satisfy the stronger HRA security and almost all of them are paring-based till now. To the best of our knowledge, we present the first detailed construction of HRA secure PREs based on standard LWE problems with *comparable small and polynomially-bounded* parameters in this paper. Combing known reductions, the HRA security of our PREs could also be guaranteed by the worst-case basic lattice problem (e.g. SIVP_γ). Our single-hop PRE schemes could be easily extended to the multi-hop case (as long as the maximum hop $L = O(1)$). Meanwhile, our single-hop PRE schemes are also key-private, which means that the implicit identities of a re-encryption key will not be revealed even in the case of a proxy colluding with some corrupted users. Some discussions about key-privacy of multi-hop PREs are also proposed, which indicates that several constructions of multi-hop PREs do not satisfy their key-privacy definitions.

Keywords: Lattice-based Cryptography · Proxy Re-encryption · Key Privacy · HRA Security · LWE

1 Introduction

A (public key, unidirectional) PRE scheme enables a semi-trusted proxy to transform ciphertexts under Alice's public key into ciphertexts which are decryptable by Bob's secret key with the help of a re-encryption key generated by Alice. Ever since it was initiated in [20] and formally constructed by Blaze, Bleumer and Strauss in [7], PRE has become a powerful cryptographic primitive with various applications, including email forwarding and publish/subscribe systems [5, 7, 8, 26], securing cloud storage and distributed file systems with fine-grained access control [6, 18, 23, 29], internet of things [1, 2], and so on.

Comparing with traditional PKEs, there are two additional probabilistic polynomial-time (PPT) algorithms of PRE schemes. One is a re-encryption key generation algorithm, which could be used to generate a re-encryption key from a user Alice to any other user Bob with the help of Alice's secret key [4]. The other is a re-encryption algorithm used to covert ciphertexts of Alice to ciphertexts of Bob. However, the semantic securities (namely, the CPA security [3, 5, 6, 26] and the CCA security (security under chosen-ciphertext attacks) [8, 12, 17]) of PREs are very similar to those of PKEs for a long time. Theoretically, a thought-out game-based security definition of PREs should take all possible situations into account after removing all the cases in which an adversary could win the corresponding security game trivially. The definition of CPA (and CCA) security of PREs is out of such an ideal range, since both the re-encryption oracle and the re-encryption key generation oracle from an honest user (including the challenge user) to any corrupted user are forbidden in the CPA (and CCA) security game. Constraints on querying the re-encryption oracle from the challenge user to corrupted users is not so reasonable. In applications, an adversary could, of course, get some re-encrypted ciphertexts from an honest user (say Alice) to some corrupted users without drawing Alice's attention. So, we may hope that the semantic security of other ciphertexts of Alice still holds under this situation. Moreover, the CPA security of PREs is inadequate even for an honest-but-curious delegatee (say Bob) [10]. As observed in [10], the CPA secure PRE scheme proposed in [26] does not prevent Bob from learning the secret key of a delegator (say Alice) with non-negligible probability after receiving a single honestly re-encrypted ciphertext of Alice. Bob could do this even without knowing the corresponding re-encryption key. Essentially, the definition of CPA security of PREs captures less security guarantees against the delegatees (Bob). Therefore, both in theory and in practical applications, the CPA security seems to be not suitable, and provides scant guarantees.

The HRA security of PREs, which is strictly stronger than the CPA security, is formally proposed by Cohen [10] in PKC 2019. The HRA security captures the goals of PREs better, and is sufficient for many applications, such as encrypted email forwarding [6], key escrow [15], single-writer many-reader encrypted storage [6, 26], key rotation for encrypted cloud storage [11]. But so far, it seems that a few constructions of PREs focus on the HRA security, and almost all known constructions are pairing-based. To the best of our knowledge, the only lattice-based PRE scheme with HRA security is constructed in [13] with exponential-sized parameters. With the rapid development of quantum computer, it's urgent to design quantum-immune PRE schemes with stronger security properties. We

[4] Such a PRE scheme is called non-interactive [6]. I.e., no trusted third party or interaction is required in the re-encryption key generation process.

mainly focus on PREs based on lattice in this paper, and aiming to design PREs with HRA security.

1.1 Our Contributions and Technique Overview

In this paper, we present the first HRA secure PRE scheme with *comparable small and polynomially-bounded* parameters in the standard model.

Our single-hop PRE schemes also satisfy the key-privacy property [5], a notion stating that it is impossible for the proxy and a set of colluding users to derive either the sender or the receiver's identities from a re-encryption key even when given the public keys and flexible interaction abilities within PREs. Both the key privacy and the HRA security of our schemes are based on standard LWE problems. Therefore, combing known reductions, we could show that our PRE schemes satisfy the HRA security and the key privacy for modulus $q = \tilde{O}(n^3)$, as long as the corresponding worst-case SIVP_γ problem with $\gamma = \tilde{O}(n^{3.5})$ is hard.

Our construction could be generalized to an L-hop PRE scheme with any $L = O(1)$ easily. Meanwhile, our single-hop PRE scheme could also be modified to a PRE+ scheme in which the re-encryption key of a ciphertext could be generated by its creator. Moreover, our single-hop PREs *with polynomially-bounded parameters* are also adaptive secure for some directed acyclic graph, since our single-hop constructions satisfy the requirements proposed in [13]. Some discussions about key-privacy of multi-hop PREs are also proposed, which indicates that several constructions of multi-hop PREs [3, 19, 25] do not satisfy their key-privacy definitions.

Technique Overview: Roughly speaking, there are two types of techniques for generating re-encryption keys in lattice-based PREs. We take the dual cryptosystem [14] as an example. The form of ciphertexts of a user i for a message $\boldsymbol{\mu} \in \{0,1\}^m$ is $(\boldsymbol{c}_{i,1} = B_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_1, \boldsymbol{c}_{i,2} = D_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_2 + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu})$, where $B_i, D_i \in \mathbb{Z}_q^{n\times m}$ are the public key of i, $\boldsymbol{s} \in \mathbb{Z}_q^n, \boldsymbol{e}_1, \boldsymbol{e}_2 \in \mathbb{Z}_q^m$ are some short vectors. The secret key of i is either a short matrix R_i satisfying $B_i \cdot R_i = D_i \bmod q$ (the case 1), or a trapdoor T_{B_i} of B_i for pre-image sampling (the case 2). For the case 1, the re-encryption key from i to a user j could be designed as $rk_{i\mapsto j} := \begin{pmatrix} S_{i,j}^T \cdot B_j + E_{i,j,1} & S_{i,j}^T \cdot D_j + E_{i,j,2} + R_i \\ 0_{m\times m} & I_{m\times m} \end{pmatrix} \in \mathbb{Z}_q^{2m\times 2m}$, where $S_{i,j}, E_{i,j,1}, E_{i,j,2}$ are matrices with small elements. Then, the re-encryption of $(\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2})$ is $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}) = (\boldsymbol{c}_{i,1}^T, \boldsymbol{c}_{i,2}^T) \cdot rk_{i\mapsto j}$. Notice that $rk_{i\mapsto j} \cdot (-R_j^T, I_{m\times m})^T \approx (-R_i^T, I_{m\times m})^T$, the user j could decrypt $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2})$ successfully [5]. For the case 2, the re-encryption key could be designed as $rk_{i\mapsto j} := (R_{i,j,1}; R_{i,j,2})$, where $R_{i,j,1}$ and $R_{i,j,2}$ are matrices with short elements which are sampled by using the trapdoor T_{B_i} of B_i, and satisfy $B_i \cdot R_{i,j,1} = B_j$, $B_i \cdot R_{i,j,2} = D_j - D_i$. Then, the re-encryption of $(\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2})$ is $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}) = (R_{i,j,1}^T \cdot \boldsymbol{c}_{i,1}, R_{i,j,2}^T \cdot \boldsymbol{c}_{i,1} + \boldsymbol{c}_{i,2})$.

[5] This is a simplified version. In order to control the growth of errors, bit decomposition technique is needed both in the re-encryption key generation algorithm and in the re-encryption algorithm [9, 25].

The *biggest difficulty* to achieve HRA security for the above two types of constructions is how to answer re-encryption queries from the challenge user i^* to other corrupted users. Since in the security proof, the public key of i^* is usually changed to be uniformly and independent (in order to embed corresponding LWE instances). Therefore, the challenger could not answer re-encryption queries in both cases. To handle this dilemma, we combine together the above two types of constructions. Let's take the single-hop construction as an example. Ciphertexts of a user i in our constructions are of the form $\boldsymbol{c}_{i,1} = B_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,1}$, $\boldsymbol{c}_{i,2} = D_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}$ and $\boldsymbol{c}_{i,3} = A_{i,1}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,3}$. Here, $A_{i,1} = A_i + [0|H \cdot \mathbf{G}|H' \cdot \mathbf{G}] = [\bar{A}|\bar{A} \cdot R_{i,1} + H \cdot \mathbf{G}|\bar{A} \cdot R_{i,2} + H' \cdot \mathbf{G}]$ with H' a random tag, the public key of i is (B_i, D_i, A_i) with $B_i \cdot R_i = D_i \bmod q$, and the secret key of i is $(R_i, R_{i,1}, R_{i,2})$ whose elements are all short. Notice that, $(\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2})$ and R_i are sufficient for decryption. To create a re-encryption key from i to another user j, we could use $[\bar{A}|\bar{A} \cdot R_{i,1} + H \cdot \mathbf{G}]$ to execute pre-image sampling for invertible H [21], and generate $rk_{i \mapsto j} = (R_{i,j,1}; R_{i,j,2})$ satisfying $[\bar{A}|\bar{A} \cdot R_{i,1} + H \cdot \mathbf{G}] \cdot R_{i,j,1} = B_j - B_i$ and $[\bar{A}|\bar{A} \cdot R_{i,1} + H \cdot \mathbf{G}] \cdot R_{i,j,2} = D_j - D_i$. Then, a re-encryption of $(\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3})$ is $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}) = (R_{i,j}^{(1)} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1}, R_{i,j}^{(2)} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,2})$, where $R_{i,j}^{(1)}$ and $R_{i,j}^{(2)}$ could be constructed from $R_{i,j,1}$ and $R_{i,j,2}$ easily. Notice that, though the forms of different level ciphertexts are different, they could be decrypted by using the same decryption algorithm. Meanwhile, for lattice-based PREs, it seems that different forms of ciphertexts are necessary. For more details, please refer to discussions above Definition 4 and Remark 6.

In the security proof, we will change A_{i^*} of the challenge use i^* to be the form $[\bar{A}|\bar{A} \cdot R_{i,1}|\bar{A} \cdot R_{i,2} - H^* \cdot \mathbf{G}]$. Here, H^* is a random tag of the challenge ciphertext. In order to answer re-encryption queries of ciphertexts with tags $H \neq H^*$ from i^* to corrupted users, the challenger could use trapdoor $R_{i^*,2}$ and $\boldsymbol{c}_{i^*,3}$ (more precisely, the parts of $\boldsymbol{c}_{i^*,3}$ corresponding to $[\bar{A}|\bar{A} \cdot R_{i^*,2} + (H - H^*) \cdot \mathbf{G}]$) to solve corresponding LWE-like problems and recover $\boldsymbol{s}$ (hence, $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2}, \boldsymbol{e}_{i,3}$ and $\boldsymbol{\mu}$) by using a LWE inversion algorithm [21]. Then, if we design the corresponding re-encryption algorithm carefully (please refer to the construction for more details), the challenger could generate a ciphertext that is distributed *statistically* close to a real re-encrypted ciphertext with the help of $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2}, \boldsymbol{e}_{i,3}, \boldsymbol{s}$ and $\boldsymbol{\mu}$ [6].

To achieve the key-privacy, *a key observation* is that we could replace B_j and D_j with random elements and re-randomize them by LWE problems. Since in our constructions, the re-encryption key generation algorithm does not use A_j. After replacing B_j and D_j, the user j still has the ability to generate re-encryption keys from himself to other users. That is to say, we could replace B_j, D_j to be $S_{i,j} \cdot B_j + E_{i,j,1}$ and $S_{i,j} \cdot D_j + E_{i,j,2}$. Then, under corresponding LWE assumptions and by the properties of Gaussian distributions, the distribution of $R_{i,j,1}$ and $R_{i,j,2}$ is (computationally) closed to some discrete Gaussian distribution that contains no information about identities i and j. We remark that our multi-hop PREs do not satisfy key-privacy, mainly because we could not re-randomize

[6] Notice that, we could only decrypt a fresh ciphertext by using this trapdoor in our design. A re-encrypted ciphertext does not contain information about matrix A anymore.

$A_{j,2} := [\bar{A}|\bar{A}\cdot R_{i,1}+H_2\cdot \mathbf{G}|\bar{A}\cdot R_{i,2}+H'\cdot \mathbf{G}]$ as $S\cdot A_{j,2}+E$, due to the $H'\cdot \mathbf{G}$ part. Our constructions are not tag-based CCA secure [12], since in our constructions, the challenger has no ability to answer the decryption query of re-encrypted ciphertexts of the challenge user i^*.

Organization: In Section 2, we will recall some notations and fundamental facts we need. Definitions of PRE schemes, detailed constructions and security analyses are put in Section 3.

2 Preliminaries

In this section, we introduce some notations and background results. Symbol $[n]$ denotes the set $\{1, 2, \cdots, n\}$. When we write $X \hookleftarrow \xi$, we mean that the random variable X follows the distribution ξ. If S is a finite set, then $|S|$ is its cardinality and $U(S)$ is the uniform distribution over S. Symbols $\mathbb{Z}^+$ and $\mathbb{R}^+$ stand for the sets of positive integers and positive reals. We use $\log x$ to represent $\log_2 x$ for $x \in \mathbb{R}^+$. We also denote $[\cdot|\cdot]$ as the horizontal concatenation of vectors or matrices.

2.1 Lattices and Gaussian Distributions

A (full-rank) n-dimensional integer lattice $\Lambda \subseteq \mathbb{Z}^n$ is a discrete additive group whose $\mathbb{R}$-span is $\mathbb{R}^n$. The basis (which is not unique) of a lattice Λ is a linearly independent set of lattice vectors $B := \{\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n\}$, whose $\mathbb{Z}$-span is Λ. In this case, we will denote it by $\Lambda = \mathcal{L}(B)$. For a matrix $A \in \mathbb{Z}^{n\times m}$ and any vector $\boldsymbol{u} \in \mathbb{Z}^n$ admitting an integral solution $\boldsymbol{x} \in \mathbb{Z}_q^m$ s.t. $A\cdot \boldsymbol{x} = \boldsymbol{u} \bmod q$, we define the "$q$-ary" integer lattices (or cosets) as $\Lambda^\perp(A) = \{\boldsymbol{z} \in \mathbb{Z}^m : A\cdot \boldsymbol{z} = \mathbf{0} \bmod q\}$, $\Lambda^\perp_{\boldsymbol{u}}(A) = \{\boldsymbol{z} \in \mathbb{Z}^m : A\cdot \boldsymbol{z} = \boldsymbol{u} \bmod q\}$ and $\Lambda(A^T) = \{\boldsymbol{z} \in \mathbb{Z}^m : \exists\, \boldsymbol{s} \in \mathbb{Z}_q^n \text{ s.t. } \boldsymbol{z} = A\cdot \boldsymbol{s} \bmod q\}$.

The Gaussian distributions are defined as follows. For any $s > 0$, $\boldsymbol{c} \in H$, which is taken to be $s = 1$ or $\boldsymbol{c} = 0$ when omitted, we define the (spherical) Gaussian function $\rho_{s,\boldsymbol{c}} : \mathbb{R}^n \to (0, 1]$ as $\rho_{s,\boldsymbol{c}}(\boldsymbol{x}) = e^{-\pi\frac{||\boldsymbol{x}-\boldsymbol{c}||^2}{s^2}}$. By normalizing this function, we obtain the continuous Gaussian probability distribution $D_{s,\boldsymbol{c}}$ of parameter s, whose density function is given by $s^{-n}\cdot \rho_{s,\boldsymbol{c}}(\boldsymbol{x})$. Applying a linear transformation given by a nonsingular matrix B to a spherical Gaussian with $s = 1$ yields the Gaussian function

$$\rho_B(\boldsymbol{x}) := \rho(B^{-1}\cdot \boldsymbol{x}) = e^{-\pi\cdot \boldsymbol{x}^T\cdot \Sigma^{-1}\cdot \boldsymbol{x}},$$

where $\Sigma = B\cdot B^T$ is a positive definite matrix (written $\Sigma > 0$) [7]. Since ρ_B is distributed only up to Σ, we'll denote it by $\rho_{\sqrt{\Sigma}}$ in the following. Normalizing $\rho_{\sqrt{\Sigma}}$ by its total measure $\int_{\mathbb{R}^n} \rho_{\sqrt{\Sigma}}(\boldsymbol{x})d\boldsymbol{x} = \sqrt{\det\Sigma}$ over $\mathbb{R}^n$, we could obtain the probability distribution function of the continuous Gaussian distribution $D_{\sqrt{\Sigma}}$.

[7] Recall that for every positive definite matrix Σ, there exists a unique positive definite matrix $\sqrt{\Sigma}$ such that $(\sqrt{\Sigma})^2 = \Sigma$.

Functions are extended to sets in the usual way, *e.g.* $\rho_{s,\boldsymbol{c}}(A) = \sum_{\boldsymbol{x}\in A} \rho_{s,\boldsymbol{c}}(\boldsymbol{x})$. Recall that, for an n-dimensional lattice Λ and a positive real ε, the smooth parameter $\eta_\varepsilon(\Lambda)$ of Λ is defined as the smallest s such that $\rho_{\frac{1}{s}}(\Lambda^*\backslash\{\mathbf{0}\}) \le \varepsilon$ [8].

For our applications, we need the following lemmata [14, 16, 21, 22, 24, 27].

Lemma 1. *Let n, q be positive integers with q prime, and let $m \ge 2n \log q$.*

- *For all but q^{-n} fraction of $A \in \mathbb{Z}_q^{n\times m}$ and any $\omega(\sqrt{\log m})$ function, there is a negligible function $\varepsilon(m)$, such that $\eta_\varepsilon(\Lambda^\perp(A)) \le \omega(\sqrt{\log m})$.*
- *For any $s \ge \omega(\sqrt{\log m})$, the distribution of $\boldsymbol{u} = A \cdot \boldsymbol{e} \bmod q$ with $\boldsymbol{e} \hookleftarrow D_{\mathbb{Z}^m,s}$ is within statistical distance 2ε of $U(\mathbb{Z}_q^n)$ for some negligible function $\varepsilon = \varepsilon(m)$. Furthermore, fix $\boldsymbol{u} \in \mathbb{Z}_q^n$ and let $\boldsymbol{t} \in \mathbb{Z}^m$ be an arbitrary solution of $A \cdot \boldsymbol{t} = \boldsymbol{u} \bmod q$, then the conditional distribution of $\boldsymbol{e} \hookleftarrow D_{\mathbb{Z}^m,s}$ given $A \cdot \boldsymbol{e} = \boldsymbol{u} \bmod q$ is exactly $D_{\Lambda_{\boldsymbol{u}}^\perp(A),s} = \boldsymbol{t} + D_{\Lambda^\perp(A),s,-\boldsymbol{t}}$.*
- *$\eta_\varepsilon(\mathbb{Z}^m) \le \omega(\sqrt{\log m})$ for some negligible $\varepsilon = \varepsilon(m)$. For any $s \ge \omega(\sqrt{\log m})$, there is an efficient sampling algorithm to output samples with distribution that is statistically close to $D_{\mathbb{Z}^m,s}$.*

Lemma 2. *The following facts hold:*

1. *Let Λ be an n-dimensional lattice and $\boldsymbol{u} \in \mathbb{R}^n$ be any vector. Assume $r, s > 0$ are two reals, and $t = \sqrt{r^2 + s^2}$. If $\frac{r\cdot s}{t} \ge \eta_\varepsilon(\Lambda)$ for some $\varepsilon \le \frac{1}{2}$, then the distribution Y, obtained by sampling from $D_{\Lambda+\boldsymbol{u},r}$ and then adding a noise vector taken from D_s, is within statistical distance 4ε from D_t.*
2. *Let $\Sigma_1, \Sigma_2 > 0$ be positive definite matrices with $\Sigma = \Sigma_1 + \Sigma_2 > 0$, Λ_1, Λ_2 be lattices such that $\sqrt{\Sigma_1} \ge \eta_\varepsilon(\Lambda_1)$ for some positive $\varepsilon \le \frac{1}{2}$. For arbitrary vectors $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathbb{R}^n$, consider the following probabilistic experiments:*

$$\textit{Samples } \boldsymbol{x}_2 \hookleftarrow D_{\Lambda_2+\boldsymbol{c}_2,\sqrt{\Sigma_2}}, \textit{ then set } \boldsymbol{x}_1 = \boldsymbol{x}_2 + D_{\Lambda_1+\boldsymbol{c}_1-\boldsymbol{x}_2,\sqrt{\Sigma_1}}.$$

   We have that the marginal distribution of $\boldsymbol{x}_1$ is within statistical distance 8ε of $D_{\Lambda_1+\boldsymbol{c}_1,\sqrt{\Sigma}}$. Moreover, if $\boldsymbol{x}_2$ is instead chosen from $D_{\sqrt{\Sigma_2}}$, the marginal distribution of $\boldsymbol{x}_1$ is also as above.

Lemma 3. *Let q, m, m' be positive integers and r be a positive real satisfying $r > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log m'})\}$. Let $\boldsymbol{b} \in \mathbb{Z}_q^m$ be arbitrary and $\boldsymbol{x}$ chosen from $D_{\mathbb{Z}^m,r}$. Then for any $V \in \mathbb{Z}^{m\times m'}$ and positive real $s > \mathfrak{s}_1(V)$ [9], there exists a* PPT *algorithm* $\mathrm{ReRand}(V, \boldsymbol{b}+\boldsymbol{x}, r, s)$ *that outputs $\boldsymbol{c}$ such that $\boldsymbol{c}^T = \boldsymbol{b}^T \cdot V + \boldsymbol{x}'$, where $\boldsymbol{x}'$ is distributed statistically close to $D_{\mathbb{Z}^{m'},2rs}$.*

Lemma 4. *Let $F \hookleftarrow D_{\mathbb{Z},\gamma}^{n\times m}$, assume for convenience that $m \ge n$. Then, with all but 2^{-m} probability it holds that $\mathfrak{s}_1(F) \le \gamma \cdot C \cdot \sqrt{m}$, where C ($\approx \frac{1}{\sqrt{2\pi}}$) is a universal constant.*

[8] Here, $\Lambda^* = \{\boldsymbol{x} \in \mathbb{R}^n :< \boldsymbol{x}, \Lambda >\subseteq \mathbb{Z}\}$.

[9] For any matrix A, symbol $\mathfrak{s}_1(A)$ represents its largest singular value.

Lemma 5. *Let $\Lambda \subseteq \mathbb{R}^n$ be any lattice, $\boldsymbol{c} \in Span_{\mathbb{R}}(\Lambda)$, and $\sigma \geq \eta_\varepsilon(\Lambda)$ for some $\varepsilon \in (0,1)$. We have $\Pr[||D_{\Lambda+\boldsymbol{c},\sigma}|| \geq \sigma \cdot \sqrt{n}] \leq 2^{-n} \cdot \frac{1+\varepsilon}{1-\varepsilon}$. Moreover, if $\boldsymbol{c} = 0$, then the bound holds for any $\sigma > 0$ with $\varepsilon = 0$.*

Lemma 6. *For any n-dimensional lattice Λ, vector $\boldsymbol{c} \in \mathbb{R}^n$, positive $\varepsilon > 0$, $\sigma \geq 2\eta_\varepsilon(\Lambda)$ and $\boldsymbol{c} \in \Lambda$, we have $D_{\Lambda,\sigma,\boldsymbol{c}}(\boldsymbol{x}) \leq \frac{1+\varepsilon}{1-\varepsilon} \cdot 2^{-n}$. In particular, for $\varepsilon \leq \frac{1}{3}$, the min-entropy of $D_{\Lambda,\sigma,\boldsymbol{c}}$ is at least $n-1$.*

For a modulus $q \geq 2$, we set $k = \lceil \log q \rceil$, and let $\mathbf{G} \in \mathbb{Z}^{n \times nk}$ be the "gadget" matrix defined in [21]. The lattice $\Lambda^{\perp(}\mathbf{G})$ has a basis B with $||B||_{\text{GS}} \leq \sqrt{5}$ [10].

Lemma 7. *Let n be a integer, $k = \lceil \log q \rceil$ with $q \geq 2$ a modulus, and $\bar{m} \geq 2nk$. For any invertible matrix $H \in \mathbb{Z}_q^{n \times n}$, $\bar{A} \hookleftarrow U(\mathbb{Z}_q^{n \times \bar{m}})$ and $R \hookleftarrow D_{\mathbb{Z}^{\bar{m} \times nk},s}$ for some $s = \omega(\log \bar{m}) \geq \eta_\varepsilon(\mathbb{Z}^{\bar{m}})$, we have:*

- *The matrix $A = [\bar{A} | \bar{A} \cdot R - H \cdot \mathbf{G}] \in \mathbb{Z}_q^{\bar{m}+nk}$ is within statistical distance $\text{negl}(n)$ of $U(\mathbb{Z}_q^{n \times (\bar{m}+nk)})$.*
- *For $\boldsymbol{b}^T = \boldsymbol{s}^T \cdot A + \boldsymbol{e}^T$, where $\boldsymbol{s} \in \mathbb{Z}_q^n$ is arbitrary and either $||\boldsymbol{e}|| < \frac{q}{C_T \sqrt{n \cdot \log q}}$ with C_T a universal constant or $\boldsymbol{e} \hookleftarrow D_{\mathbb{Z}^m,\alpha q}$ with $\frac{1}{\alpha} \geq \sqrt{n \cdot \log q} \cdot \omega(\sqrt{\log n})$, there is a deterministic algorithm* $\mathbf{Invert}(R, A, \boldsymbol{b})$ *outputs $\boldsymbol{s}$ and $\boldsymbol{e}$.*
- *For any $\boldsymbol{u} \in \mathbb{Z}_q^n$ and large enough $\gamma = O(\sqrt{n \cdot \log q})$, there is a randomized algorithm* $\mathbf{SampleD}(R; A; \boldsymbol{u}; \gamma)$ *samples from a distribution within $\text{negl}(n)$ statistical distance of $D_{\Lambda_{\boldsymbol{u}}^{\perp}(A),\gamma \cdot \omega(\sqrt{\log n})}$.*

We note that for any fixed matrix A, algorithm **SampleD** in Lemma 7 could output samples from a distribution within $\text{negl}(n)$ statistical distance of $D_{\Lambda_{\boldsymbol{u}}^{\perp}(A),\gamma_1 \cdot \omega(\sqrt{\log n})}$ for any $\gamma_1 \geq \gamma$. Also, Lemmata 1 and 2 (together with the fact that $\frac{1-\varepsilon}{1+\varepsilon} \cdot \rho_s(\Lambda) \leq \rho_s(\Lambda + \boldsymbol{u}) \leq \rho_s(\Lambda)$ for any $\varepsilon \in (0,1)$, $s \geq \varepsilon_\varepsilon(\Lambda)$ and any vector $\boldsymbol{u}$, which is implied by Lemmata 2.9 and 4.4 of [22]) imply the following result: for $m \geq 2n \log q$, $A \hookleftarrow U(\mathbb{Z}_q^{n \times m})$ and $\gamma \geq \omega(\sqrt{\log m})$, the distribution of $\boldsymbol{e}$ obtained by first choosing a vector $\boldsymbol{u} \hookleftarrow U(\mathbb{Z}_q^n)$, then choose $\boldsymbol{e} \hookleftarrow D_{\Lambda_{\boldsymbol{u}}^{\perp}(A),\gamma}$ is statistically closed to $D_{\mathbb{Z}^m,\gamma}$.

In our constructions, we need a large set $\mathfrak{S} \subseteq \mathbb{Z}_q^{n \times n}$ with the "unit differences" property: for any matrices $X \neq Y \in \mathfrak{S}$, the matrix $X - Y$ is invertible in $\mathbb{Z}_q^{n \times n}$. Such a set could be constructed efficiently, and for $q = p^e$ with p a prime, we have $|\mathfrak{S}| = p^n$ [21].

2.2 The LWE Problems

Even since introduced by Regev [27], the LWE problems have become a fundamental building block of many cryptographic primitives over lattices. The formal definition of decision LWE problems is proposed as follows.

[10] For a matrix $R \in \mathbb{Z}^{n \times n}$, denote $||R||_{\text{GS}}$ as the longest column of the Gram-Schmidt orthogonalization of R.

Definition 1. *For positive integers n, m, q and a noise distribution χ over $\mathbb{Z}$, the decision LWE problems with m samples, denoted by* $\mathrm{LWE}_{n,q,m,\chi}$ *is to distinguish the following two distributions:*

$$(A, A^T \cdot \boldsymbol{s} + \boldsymbol{e}) \text{ and } (A, \boldsymbol{u})$$

where $A \hookleftarrow U(\mathbb{Z}_q^{n\times m})$, $\boldsymbol{s} \hookleftarrow \chi^n$, $\boldsymbol{e} \hookleftarrow \chi^m$, and $\boldsymbol{u} \hookleftarrow U(\mathbb{Z}_q^m)$ are sampled independently.

There are quantum reductions from worst-case lattice problems to decision LWE problems for suitable χ [4, 27]. By using standard hybrid argument, it is possible to show that decision LWE problems with multiple secrets are also hard. Namely, it is hard to distinguish the following two distributions:

$$(A, A^T \cdot S + E) \text{ and } (A, U)$$

where $A \hookleftarrow U(\mathbb{Z}_q^{n\times m})$, $S \hookleftarrow \chi^{n\times k}$, $E \hookleftarrow \chi^{m\times k}$ and $U \hookleftarrow U(\mathbb{Z}_q^{m\times k})$ are sampled independently. We will denote the corresponding problem by Multi-$\mathrm{LWE}^k_{n,q,m,\chi}$.

3 The PRE Schemes Based on LWE

In this section, we shall give formal definitions and security models of PRE schemes, present some simple discussions, and propose our PRE schemes based on standard LWE problems with different properties.

3.1 Definitions and Security Models

Let's first recall the definitions of PRE schemes [5, 10].

Definition 2. *Let λ be the security parameter. An L-hop proxy re-encryption scheme contains the following six* PPT *algorithms.*

- **Setup**$(1^\lambda, L)$: *The key management center runs this algorithm with input λ and L, and generates the public parameters pp.*
- **KeyGen**(pp) : *To generate a public/secret key pair of a user i, the key management center runs this algorithm with input pp, and generates (pk_i, sk_i).*
- **Enc**(pp, pk_i, m) : *The encryption algorithm, takes as input the public parameters pp, a message m and the public key pk_i of a user i, generates a ciphertext $(C_i; 1)$* [11] *associated with m and i.*
- **ReKeyGen**(pp, sk_i, pk_j, κ) : *This algorithm, which is executed by a user i, takes as input the public parameters pp, the secret key sk_i of the user i, the public key pk_j of a user j and a level $\kappa \leq L$, outputs a re-encryption key $rk^\kappa_{i\mapsto j}$ which could be used to re-encrypt any level κ ciphertext of the user i to a corresponding level $\kappa + 1$ ciphertext of the user j.*

[11] Throughout this paper, we call such a ciphertext a fresh (level 1) ciphertext. In our constructions, every ciphertext C_i contains its corresponding level information.

- **ReEnc**$(pp, (C_i; \kappa), rk^{\kappa}_{i \mapsto j})$: *This algorithm is executed by the proxy, and takes as input the public parameters pp, a level κ ciphertext C_i of a user i, a re-encryption key $rk^{\kappa}_{i \mapsto j}$, and outputs a level $\kappa + 1$ ciphertext C_j. If the re-encryption key $rk^{\kappa}_{i \mapsto j}$ does not match the level κ of C_i, output $\perp$.*
- **Dec**$(pp, (C_i; \kappa), sk_i)$: *For any $\kappa \in [L+1]$, this algorithm could decrypt any level-κ ciphertext C_i of a user i to a message m with the secret sk_i and the public parameters pp.*

Unlike that proposed in [13], we choose to define the re-encryption key associated to a level κ as [12], so it can not be used to re-encrypt a ciphertext with any different level $\kappa' \neq \kappa$. Notice that one could decrypt his ciphertexts with his secret key by using the same decryption algorithm, regardless of the level of ciphertexts. So, our definitions and constructions satisfy the so-called proxy invisibility [23]. When $L = 1$ (namely, the single-hop PRE scheme), the re-encryption key generation algorithm needs not to take κ as an input. As usual, for an L-hop PRE scheme, the correctness is required that for any $pp \leftarrow \mathbf{Setup}(1^{\lambda})$, $(pk_{i_j}, sk_{i_j}) \leftarrow \mathbf{KeyGen}(pp)$ with $j \in [L+1]$, $(C_{i_1}; 1) = \mathbf{Enc}(pp, pk_{i_1}, m)$ with any plaintext m, re-encryption keys $rk^{j-1}_{i_{j-1} \mapsto i_j} = \mathbf{ReKeyGen}(pp, sk_{i_{j-1}}, pk_{i_j}, j-1)$, and all re-encrypted ciphertexts $(C_{i_j}; j) = \mathbf{ReEnc}(pp, C_{i_{j-1}}, rk^{j-1}_{i_{j-1} \mapsto i_j})$ with $j \in \{2, \cdots, L+1\}$, we have

$$\Pr[\mathbf{Dec}(pp, C_{i_j}, sk_{i_j}) \neq m] \leq \mathrm{negl}(\lambda)$$

for all $j \in [L+1]$.

The HRA security, proposed in [10], is strictly stronger than the CPA security [3, 5]. In the definition of HRA security, re-encryptions even from the challenge identity i^* to corrupted users are also allowed, as long as the queried ciphertext does not equal to the challenge ciphertext (or some derivations of it). In our constructions, every ciphertext is labeled by some randomly chosen tag as [12]. The formal definition of the HRA security of (our tag-based) L-hop PRE schemes (with $L = O(1)$) could be modified as follows.

Definition 3. *Consider the following interactions between a* PPT *adversary $\mathfrak{A}$ and a challenger $\mathfrak{C}$:*

- *Phase* 1*:*
  1. *The challenger $\mathfrak{C}$ generates the public parameters pp and sends them to an adversary $\mathfrak{A}$. It also maintains a counter* **numCt** *which is initialized to be* 0*, and two databases $\mathfrak{D}_K, \mathfrak{D}_C$ which are initialized to be empty.*
  2. *The adversary $\mathfrak{A}$ first decides the sizes of two sets U_H and U_C, and could adaptively query the following two oracles.*
     - *Corrupted key generation oracle, in which $\mathfrak{C}$ would return $(pk, sk) =$* **KeyGen**(pp) *to $\mathfrak{A}$ and record (pk, sk), together with corresponding user index, to the set U_C.*
     - *Uncorrupted key generation oracle, in which $\mathfrak{C}$ would first generate $(pk, sk) =$* **KeyGen**(pp)*, then return pk to $\mathfrak{A}$ and record (pk, sk), together with corresponding user index, to the set U_H.*

- Phase 2: In this phase, $\mathfrak{A}$ could adaptively query the following oracles. We require that the queried users' indexes must have been appeared in Phase 1, and user indexes $i \neq j$ if exist.
  1. Encryption oracle: On input an index i and a message m, the challenger $\mathfrak{C}$ computes $C_i = \mathbf{Enc}(pp, pk_i, m)$, increments $\mathbf{numCt}$, then adds the item $(m; \perp, i; \perp, \boldsymbol{ct}_i := (C_i; H_{C_i}; 1); \mathbf{numCt})$ [12] to $\mathfrak{D}_C$ and returns it to the adversary $\mathfrak{A}$.
  2. Re-encryption key generation oracle: Upon receiving indexes (i, j) and a level $\kappa \in [L]$, the challenger $\mathfrak{C}$ will generate a re-encryption key $rk^{\kappa}_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j, \kappa)$, record the tuple $(i, j, rk^{\kappa}_{i \mapsto j}; \kappa)$ to the database $\mathfrak{D}_K$ and return $rk^{\kappa}_{i \mapsto j}$ to $\mathfrak{A}$.
  3. Re-encryption oracle: $\mathfrak{A}$ submits query $(i, j, k; \kappa)$ for some $k \leq \mathbf{numCt}$ and $\kappa \in [L]$, the challenger $\mathfrak{C}$ will first retrieve $\mathfrak{D}_C$ to recover an item $(*, i; *, \boldsymbol{ct}_i; k)$, where $*$ could be equal to $\perp$, and could represent any value. If no such item exists, output $\perp$. Otherwise, generate a re-encryption key $rk^{\kappa}_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j, \kappa)$ and parse $\boldsymbol{ct}_i = (C_i; H_{C_i}; \kappa')$. If $\kappa' \neq \kappa$, return $\perp$. Otherwise, compute $C_j = \mathbf{ReEnc}(pp, C_i, rk^{\kappa}_{i \mapsto j})$, increment $\mathbf{numCt}$, record $(i, j; \boldsymbol{ct}_i, \boldsymbol{ct}_j := (C_j; H_{C_j}; \kappa + 1); \mathbf{numCt})$ to $\mathfrak{D}_C$, and return $(i, j; \boldsymbol{ct}_i, \boldsymbol{ct}_j; \mathbf{numCt})$ to $\mathfrak{A}$.
  4. Challenge oracle: This oracle could only be queried once. After $\mathfrak{A}$ submits a challenge (i^*, m^*_0, m^*_1) with $i^* \in U_H$ and $m^*_0 \neq m^*_1$, $\mathfrak{C}$ will first chooses a random bit $b \hookleftarrow U(\{0, 1\})$, and computes $C^*_b = \mathbf{Enc}(pp, pk_{i^*}, m^*_b)$. At the same time, $\mathfrak{C}$ increments $\mathbf{numCt}$, records the corresponding item $(\perp, i^*; \perp, \boldsymbol{ct}^* := (C^*_b; H^*; 1); \mathbf{numCt})$ to $\mathfrak{D}_C$ and returns it to $\mathfrak{A}$.
- Phase 3: Legitimacy detection. The challenger $\mathfrak{C}$ checks whether there is a re-encryption chain of $\boldsymbol{ct}^*$ (which consists of re-encryptions obtained from re-encryption oracle, and re-encryptions which could be generated by $\mathfrak{A}$ itself via re-encryption keys obtained by querying re-encryption key generation oracle) with length at most L from i^* to some corrupted user $j \in U_C$ by using $\mathfrak{D}_k$ and $\mathfrak{D}_C$ [13]. If so, output $\perp$.
- Phase 4: $\mathfrak{A}$ outputs a bit b', and wins if and only if $b' = b$.

We say an L*-hop PRE scheme is* HRA *secure, if for any* PPT *adversary* $\mathfrak{A}$*, its advantage* $\Pr[\mathfrak{A} \text{ wins}] := |\Pr[b' = b] - \frac{1}{2}|$ *in the above game is negligible.*

Remark 1. Our definition of the HRA security is different from that proposed in [10]. More precisely, we allow the adversary to query re-encryption keys from non-challenge honest users to corrupted users as [12]. Meanwhile, the legitimacy detection could only be finished by a PPT challenger for $L = O(1)$. Also notice that multi-hop security definition proposed in [12] requires an adversary to submit a graph structure at the beginning of the security game.

[12] Here, H_{C_i} is a random tag which could be regarded as part of C_i. We emphasize its role by giving it explicitly in this definition. In our constructions, the tag of different ciphertexs could be the same (which do not affect securities).

[13] Notice that $\mathfrak{A}$ is a PPT adversary. So, $|\mathfrak{D}_K| + |\mathfrak{D}_C| \leq \text{poly}(\lambda)$, and it is possible to recover all these chains, if exist, by using exhausted search since $L = O(1)$.

Remark 2. In the above definition (as well as that proposed in [10]) only indistinguishability of fresh ciphertexts is considered. It's possible to modify the encryption algorithm of Definition 2 to be level-based as [12]. The corresponding encryption algorithm is proposed as following:

- $\mathbf{Enc}(pp, pk_i, m, \kappa)$: The encryption algorithm, takes as input the public parameters pp, a message m, the public key pk_i of a user i and a level $\kappa \in [L+1]$, generates a ciphertext $(C_i; \kappa)$ associated with m and the user i.

We use Definition 2 mainly because we only consider the HRA security proposed in [10]. One could also define the indistinguishability of level κ ciphertexts for any $\kappa \in [L+1]$, either by generating corresponding ciphertexts via encryption algorithms, or by generating corresponding ciphertexts via re-encryption algorithms. For some constructions (e.g. those are source hiding [13], or re-encryption simulatable [10] [14]), these definitions are equivalent. Most importantly, for challenge users all belonged to U_H, the (fresh ciphertexts, $\kappa = 1$) HRA security defined in Definition 2 implies the level-based HRA security (defined via re-encryption algorithms) with any level $\kappa \geq 2$ [15].

As explained in [10], we need to use the encryption oracle to track derivations of the challenge ciphertext. *Another important reason* which has not been mentioned in [10] is that for a single-hop PRE scheme, the correctness only guarantees that we could decrypt successfully for both fresh and re-encrypted ciphertexts. However, for a ciphertext which is re-encrypted by 2 times, we do not know the probability that we could decrypt it successfully. If some of the re-encryption processes are malformed, this probability may be very high, especially in some lattice-based PRE schemes [3, 12, 17]. *Most importantly*, the encryptions of the challenge user i^* must be recorded in the definition the HRA security, since adding a small error to the challenge ciphertext may do not affect its decryption in latticed-based constructions. Similar concerns also exist in multi-hop lattice-based PRE schemes. Definitions given in [12] did not use the encryption oracle, and only required the queried ciphertext to be well-formed in the sense that it was an encryption of a message under the claimed public key. *This requirement is not enough in general.*

The key privacy of a single-hop PRE scheme is defined as follows [3, 5].

Definition 4. *Consider the following interactions between a* PPT *adversary* $\mathfrak{A}$ *and a challenger* $\mathfrak{C}$*, in which Phase* 1 *is almost the same as that in* Definition 3 *with the differences that the challenger does not need to maintain* $\mathfrak{D}_C$ *and* **numCt**.

[14] Source hiding requires corresponding ciphertexts to be *computationally* indistinguishable, while re-encryption simulatable requires the distributions of corresponding ciphertexts to be *statistically* close to each other.

[15] For those constructions with high level ($\kappa \geq 2$) ciphertexts having different forms or distributions (when generating from encryption algorithms or re-encryption algorithms), things seem to be more complicated.

- *Phase* 2*: in this phase,* $\mathfrak{A}$ *could adaptively query the following oracle. We require that the queried users' indexes must have been appeared in Phase* 1*.*
  1. *Re-encryption key generation oracle: Upon receiving indexes* (i, j) *with* $i \neq j$*, the challenger* $\mathfrak{C}$ *will generate* $rk_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j)$*, record* $(i, j, rk_{i \mapsto j})$ *to* $\mathfrak{D}_K$ *and return* $rk_{i \mapsto j}$ *to* $\mathfrak{A}$*. If this oracle is queried after the Challenge oracle and* $(i, j) = (i^*, j^*)$*, we require the generated re-encryption key does not equal to* rk^**.*
  2. *Re-encryption oracle:* $\mathfrak{A}$ *submits query* (i, j, C_i) *with* $i \neq j$*, the challenger* $\mathfrak{C}$ *will first create* $rk_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j)$*, and return* $C_j = \mathbf{ReEnc}(pp, C_i, rk_{i \mapsto j})$ *to* $\mathfrak{A}$*.*
  3. *Challenge oracle: This oracle could only be accessed once. On input* (i^*, j^*)*, the challenger* $\mathfrak{C}$ *creates* $rk_{i^* \mapsto j^*} = \mathbf{ReKeyGen}(pp, sk_{i^*}, pk_{j^*})$ *until* $(i^*, j^*, rk_{i^* \mapsto j^*}) \notin \mathfrak{D}_K$*, if no such key exists, return* $\perp$*. Then, he samples a random* $b \hookleftarrow U(\{0, 1\})$*, and returns* $rk_{i^* \mapsto j^*}$ *to* $\mathfrak{A}$ *if* $b = 1$*, or returns a random key* rk^* *(which may obey to some distributions with large enough entropy) in the key space to* $\mathfrak{A}$ *if* $b = 0$*. The constraints are* $i^* \neq j^*$ *and* $j^* \in \Gamma_H$*.*
- *Phase* 3*:* $\mathfrak{A}$ *outputs a bit* b'*, and wins if and only if* $b' = b$*.*

We say a single-hop PRE scheme is key-private, if for any PPT *adversary* $\mathfrak{A}$*, its advantage* $\Pr[\mathfrak{A}\ wins] := |\Pr[b' = b] - \frac{1}{2}|$ *in the above game is negligible.*

Remark 3. Note that there is no constraint on i^* as in [3], which means that honest delegatees are enough to provide key privacy. This definition seems stronger, since key-privacy definition proposed in [5] requires $i^* \in U_H$. However, in many applications, if user i^* is corrupted, there may be no point to discuss the privacy of re-encryption key $rk_{i^* \mapsto j^*}$, since the re-encryption key from i^* to j^* is usually produced by user i^*.

Remark 4. In the key privacy definition proposed in [5], the re-encryption key $rk_{i \mapsto j}$ generated for each pair of users i, j is unique. Under this constraint, the re-encryption algorithm can't be deterministic. However, in Definition 4, even many different re-encryption keys from challenger user i^* to j^* could be exposed to the adversary, as long as the re-encryption key sampled in the challenge oracle never appears before. Our sing-hop constructions satisfy this stronger key-privacy definition, however schemes constructed in [5] do not remain key-private if multiple keys per pair are released.

The key-privacy defined in Definition 4 is very strong in the sense that we do not constrain the ability of an adversary to query re-encryption key generation oracle and re-encryption oracle from challenge user i^* and j^* to corrupted users. Next, we give a simple impossible result, that the key privacy defined in Definition 4 could not be applied to multi-hop PRE schemes directly. *This result also means that the multi-hop PRE schemes proposed in [3, 19] do not satisfy their key-privacy definitions.*

Lemma 8. *The key privacy defined in Definition 4 could not be applied to multi-hop PRE scheme which satisfies correctness and CPA security.*

Proof. We give an attack for the multi-hop case. Consider the case both i^* and j^* in the challenge phase belonging to U_H. Adversary $\mathfrak{A}$ could choose an arbitrary message m, encrypt it under user i^*'s public key and get C_{i^*}. Notice that $\mathfrak{A}$ could do this both before and after the challenge phase. Assume $rk_{i^* \mapsto j^*}$ is the re-encryption key obtained from $\mathfrak{C}$, $\mathfrak{A}$ then compute $C_{j^*} = \mathbf{ReEnc}(pp, C_{i^*}, rk_{i^* \mapsto j^*})$, and query the re-encryption oracle with input j^*, k, C_{j^*} to get corresponding ciphertext C_k. Here, k is an arbitrary user in U_C. Finally, $\mathfrak{A}$ computes $m' = \mathbf{Dec}(pp, C_k, sk_k)$ and judges whether $m' = m$.

If $rk_{i^* \mapsto j^*}$ is a valid re-encryption key from user i^* to j^*, then $m' = m$ with overwhelming probability by the correctness of multi-hop PRE schemes. While if $rk_{i^* \mapsto j^*}$ is a random key, then the probability that $m' = m$ is negligible since the PRE scheme is CPA secure.

Remark 5. The key-privacy definition proposed in [3] adds an additional constraint in the challenge phase that there should be no chain of re-encryption keys from j^* to any $k \in U_C$. However, attack showed in Lemma 8 means that this constraint is not enough, *i.e.* the key privacy definition of multi-hop PRE re-encryption schemes proposed in [3] *is not well-defined.*

To define the key-privacy of multi-hop PRE schemes, a key point is to prohibit the adversary from winning the distinguishing game via attack proposed in Lemma 8 trivially. So, we also need to record the ciphertext generated in the security game as Definition 3. The detailed definitions are given in Appendix A.

3.2 Single-Hop Key-Private PRE Schemes with HRA Security

In this subsection, we propose our sing-hop PRE schemes, and use the following parameters.

- The modulus q is a prime, $\mathfrak{S} \subseteq \mathbb{Z}_q^{n \times n}$ is a set with the unit differences property [21].
- Integers $k = \lceil \log q \rceil$, $\bar{m} \geq 2nk$, $m = \bar{m} + 2nk$.
- $s = \omega(\sqrt{\log m}) \geq \eta_\varepsilon(\mathbb{Z}^m)$ is some fixed function, which is used for sampling Gaussian distributions over $\mathbb{Z}^m$ [16].
- $\beta = C \cdot s \cdot \sqrt{\bar{m}} + 1$ (which is used in the ReRand algorithm of Lemma 3) with C the global constant appeared in Lemma 4; $\gamma = O(\sqrt{n \cdot \log q}) \cdot \omega(\log m)$ (which is used for pre-image sampling) with the O function appeared in Lemma 7.
- $\chi = D_{\mathbb{Z},\alpha}$ is the error distribution of corresponding (single-secret/multi-secrets) LWE problems with $\alpha \geq 2\sqrt{n} \geq 2 \cdot \omega(\sqrt{\log m})$ (which is used for ensuring worst-case to average-case reductions of LWE problems).

In the following constructions, $\mathbf{G} \in \mathbb{Z}^{n \times nk}$ is the gadget matrix [21]. To ensure correctness of decryptions, we require

$$q \geq 10 \cdot \omega(\log m) \cdot \sqrt{m} \cdot (1 + \sqrt{n} \cdot \alpha + \sqrt{5} \cdot \sqrt{\bar{m} + nk} \cdot \beta \cdot \gamma) \cdot \alpha$$

[16] Note that $D^m_{\mathbb{Z},s} = D_{\mathbb{Z}^m,s}$.

in general. If $\alpha \leq \beta \cdot \gamma$ and $\omega(\sqrt{\log m}) \geq 1 + \sqrt{5}$, we get $q \geq 10 \cdot \omega(\log^{\frac{3}{2}} m) \cdot \sqrt{m \cdot (\bar{m} + nk)} \cdot \gamma \cdot \alpha \cdot \beta$ is sufficient. Notice that $k = O(\log q) = \tilde{O}(1)$, if we set $\bar{m} = 2nk$, then $q = \tilde{O}(n^2 \cdot \alpha^2)$ is sufficient. To ensure worst-case to average-case reductions of LWE problems, we set $\alpha = \alpha' \cdot q$ (according to our definition of LWE problems), and have a (quantum) reduction from worst-case SIVP$_\gamma$ problem to LWE problem with $\gamma = \tilde{O}(\frac{n}{\alpha'})$, as long as $\alpha' \cdot q \geq 2\sqrt{n}$ [27]. So, a possible parameter setting is $\alpha' = \tilde{O}(n^{-2.5})$ and $q = \tilde{O}(n^3)$. Meanwhile, we could conclude that our PRE schemes satisfy HRA security and key privacy assuming the worst-case SIVP$_{\tilde{O}(n^{3.5})}$ problem is hard (by Theorems 1 and 2).

Our single-hop key-private PRE scheme with HRA security is proposed as follows.

- **Setup**(1^λ) : This algorithm uses large enough integer n (e.g. $n \geq 512$) and parameters satisfying the above relations. It also samples $\bar{A} \hookleftarrow U(\mathbb{Z}_q^{n \times \bar{m}})$, $H_1 \hookleftarrow U(\mathfrak{G})$ and outputs the public parameters
$$pp = (n, q, \bar{m}, \chi, D_{\mathbb{Z},s}, h, \mathbf{G}; \bar{A}; H_1).$$
- **KeyGen**(pp) : To generate a public/secret key pair of a user i, this algorithm, implemented by the key management center, samples matrices $B_i \hookleftarrow U(\mathbb{Z}_q^{n \times m})$, $R_i \hookleftarrow D_{\mathbb{Z},s}^{m \times m}$ and $R_{i,1}, R_{i,2} \hookleftarrow D_{\mathbb{Z},s}^{\bar{m} \times nk}$. Then, it computes $A_i = [\bar{A} | \bar{A} \cdot R_{i,1} | \bar{A} \cdot R_{i,2}] \in \mathbb{Z}_q^{n \times m}$, $D_i = B_i \cdot R_i \in \mathbb{Z}_q^{n \times m}$ and returns
$$(pk_i, sk_i) = ((A_i, B_i, D_i), (R_i, R_{i,1}, R_{i,2})).$$
- **Enc**$(pp, pk_i, \boldsymbol{\mu})$: To encrypt a message $\boldsymbol{\mu} \in \{0,1\}^m$ under the public key pk_i of a user i, this algorithm first samples $\boldsymbol{s} \hookleftarrow \chi^n$ and errors $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2} \hookleftarrow \chi^m$. It also samples error $\boldsymbol{e}_{i,3} \hookleftarrow D_{\mathbb{Z}^m, 2 \cdot \beta \cdot \alpha}$ and $H_{\boldsymbol{c}_i} \hookleftarrow U(\mathfrak{G})$. Then, it sets the level matrix
$$\begin{aligned} A_{i,1} &= A_i + [0 | H_1 \cdot \mathbf{G} | H_{\boldsymbol{c}_i} \cdot \mathbf{G}] \\ &= [\bar{A} | \bar{A} \cdot R_{i,1} + H_1 \cdot \mathbf{G} | \bar{A} \cdot R_{i,2} + H_{\boldsymbol{c}_i} \cdot \mathbf{G}] \in \mathbb{Z}_q^{n \times m}, \end{aligned}$$
and computes $\boldsymbol{c}_{i,1} = B_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,1} \bmod q$, $\boldsymbol{c}_{i,2} = D_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q$, $\boldsymbol{c}_{i,3} = A_{i,1}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,3} \bmod q$. Finally, it outputs $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}; H_{\boldsymbol{c}_i}, 1) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^m \times \mathbb{Z}_q^m \times \mathbb{Z}_q^{n \times n} \times [2]$.
- **ReKeyGen**(pp, sk_i, pk_j) : The user i samples $X_{1,1} \in \mathbb{Z}^{\bar{m} \times \bar{m}}, X_{1,2}, X_{1,3} \in \mathbb{Z}^{\bar{m} \times nk}, X_{2,1} \in \mathbb{Z}^{nk \times \bar{m}}$, and $X_{2,2}, X_{2,3} \in \mathbb{Z}^{nk \times nk}$ by using algorithm **SampleD** (of Lemma 7) with $R_{i,1}$ and γ, such that
$$A_{i,1} \cdot \begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,3} \\ X_{2,1} & X_{2,2} & X_{2,3} \\ 0_{nk \times \bar{m}} & 0_{nk \times nk} & 0_{nk \times nk} \end{pmatrix} = S_{i,j} \cdot B_j + E_{i,j,1} - B_i \bmod q \text{ [17]},$$
where $S_{i,j} \hookleftarrow \chi^{n \times n}$ and $E_{i,j,1} \hookleftarrow \chi^{n \times m}$. Similarly, he also samples matrices $X'_{1,1} \in \mathbb{Z}^{\bar{m} \times \bar{m}}, X'_{1,2}, X'_{1,3} \in \mathbb{Z}^{\bar{m} \times nk}, X'_{2,1} \in \mathbb{Z}^{nk \times \bar{m}}$, and $X'_{2,2}, X'_{2,3} \in \mathbb{Z}^{nk \times nk}$

[17] In fact, $S_{i,j} \cdot B_j + E_{i,j,1}$ is sufficient.

by using algorithm **SampleD** with $R_{i,1}$ and γ, such that

$$A_{i,1} \cdot \begin{pmatrix} X'_{1,1} & X'_{1,2} & X'_{1,3} \\ X'_{2,1} & X'_{2,2} & X'_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix} = S_{i,j} \cdot D_j + E_{i,j,2} - D_i \bmod q,$$

where $E_{i,j,2} \hookleftarrow \chi^{n\times m}$. In the end, he returns the corresponding re-encryption key

$$rk_{i\mapsto j} = (\{X_{i',j'}; X'_{i',j'}\}_{i'\in\{1,2\},j'\in\{1,2,3\}}).$$

- **ReEnc**$(pp, \boldsymbol{ct}_i, rk_{i\mapsto j})$: The proxy parses $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}; h(\boldsymbol{\mu}), 1)$. If $\boldsymbol{ct}_i$ does not follow to this form, return $\perp$. Otherwise, it sets

$$R^{(1)}_{i,j} = \begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,3} \\ X_{2,1} & X_{2,2} & X_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix}^T, R^{(2)}_{i,j} = \begin{pmatrix} X'_{1,1} & X'_{1,2} & X'_{1,3} \\ X'_{2,1} & X'_{2,2} & X'_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix}^T.$$

He first computes

$$\boldsymbol{c}'_{j,1} := R^{(1)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1} + R^{(1)}_{i,j} \cdot \boldsymbol{e}'_1 \bmod q$$

and

$$\boldsymbol{c}'_{j,2} := R^{(2)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,2} + R^{(2)}_{i,j} \cdot \boldsymbol{e}'_2 \bmod q.$$

Here, $\boldsymbol{e}'_{k'} = ((\boldsymbol{e}'_{k',1})^T, (\boldsymbol{e}'_{k',2})^T, \boldsymbol{0}^T_{nk})^T \in \mathbb{R}^m$ with $\boldsymbol{e}'_{k',1} \hookleftarrow D^{\bar{m}}_\alpha$ and $\boldsymbol{e}'_{k',2} \hookleftarrow D^{nk}_\alpha$ for $k' \in \{1,2\}$. Then, he computes $\boldsymbol{c}_{j,1} = \boldsymbol{c}'_{j,1} + \boldsymbol{e}''_1 \bmod q$ and $\boldsymbol{c}_{j,2} = \boldsymbol{c}'_{j,2} + \boldsymbol{e}''_2 \bmod q$. Here, $\boldsymbol{e}''_1 \hookleftarrow D_{\mathbb{Z}^m - \boldsymbol{c}'_{j,1}, \boldsymbol{r}}$ with

$$\boldsymbol{r}_{k'} = \sqrt{((\bar{m}+nk)\cdot\gamma^2 - ||R^{(1)}_{i,j,k'}||^2)\cdot(4\beta^2+1)\cdot\alpha^2 + \omega(\sqrt{\log m})}$$

for $k' \in [m]$, where $R^{(1)}_{i,j,k'}$'s are the row vectors of $R^{(1)}_{i,j}$. Similarly, $\boldsymbol{e}''_2 \hookleftarrow D_{\mathbb{Z}^m - \boldsymbol{c}'_{j,2}, \boldsymbol{r}'}$ with

$$\boldsymbol{r}'_{k'} = \sqrt{((\bar{m}+nk)\cdot\gamma^2 - ||R^{(2)}_{i,j,k'}||^2)\cdot(4\beta^2+1)\cdot\alpha^2 + \omega(\sqrt{\log m})}$$

for $k' \in [m]$, where $R^{(2)}_{i,j,k'}$'s are the row vectors of $R^{(2)}_{i,j}$. Finally, he returns $\boldsymbol{ct}_j = (\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \perp; \perp, 2)$ [18].
- **Dec**$(pp, \boldsymbol{ct}_i, sk_i)$: To decrypt a ciphertext of user i, we first parse $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}$ or $\perp; H_{\boldsymbol{c}_i}$ or $\perp, l)$ with $l \in \{1,2\}$, compute $\boldsymbol{\mu}' = -R^T_i \cdot \boldsymbol{c}_{i,1} + \boldsymbol{c}_{i,2} \bmod q$, and return $\boldsymbol{\mu}$ with $\boldsymbol{\mu}_i = \lfloor \frac{2}{q} \cdot \boldsymbol{\mu}'_i \rceil$ for $i \in [m]$.

To show the correctness, we first notice that for a fresh (i.e. level 1) ciphertext $(\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}; H_{\boldsymbol{c}_i}, 1)$ of a user i, we have

$$-R^T_i \cdot \boldsymbol{c}_{i,1} + \boldsymbol{c}_{i,2} \bmod q = -R^T_i \cdot \boldsymbol{e}_{i,1} + \boldsymbol{e}_{i,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q.$$

[18] We could also put a random tag here. But this tag is irrelevant to ciphertexts.

Since $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2} \hookleftarrow \chi^m$, the absolute value of each of the coefficients of $-R_i^T \cdot \boldsymbol{e}_{i,1} + \boldsymbol{e}_{i,2}$ is bounded by $N_1 := (\sqrt{m}+1) \cdot \alpha \cdot \omega(\sqrt{\log m})$. Then, for any $j \in [m]$, we get $\boldsymbol{\mu}'_j = err_1 + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}_j$ for $err_1 \in (-N_1, N_1)$ with overwhelming probability. Therefore, our choice of q ensures that for any $j \in [m]$, $\boldsymbol{\mu}_j = \lfloor \frac{2}{q} \cdot (err_1 + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}_j) \rceil$, since $|err_1| \leq \frac{q}{5}$.

For a re-encrypted (i.e. level 2) ciphertext $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \perp; \perp, 2)$, we first analyze the distributions of $\boldsymbol{c}'_{j,1}$ and $\boldsymbol{c}'_{j,2}$ in the re-encryption algorithm. Notice that

$$\begin{aligned} & R_{i,j}^{(1)} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1 \bmod q \\ = & R_{i,j}^{(1)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1 + \boldsymbol{e}_{i,1} + B_j^T \cdot S_{i,j}^T \cdot \boldsymbol{s} + E_{i,j,1}^T \cdot \boldsymbol{s} \bmod q, \end{aligned}$$

by Lemma 2 and the property of Gaussian distribution, we get $R_{i,j}^{(1)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1 \hookleftarrow D_{\boldsymbol{s}}$ with $\boldsymbol{s}_{k'} = ||R_{i,j,k'}^{(1)}|| \cdot \sqrt{4\beta^2+1} \cdot \alpha$ for $k' \in [m]$. Therefore, by using Lemma 2 again, we get that the distribution of $R_{i,j}^{(1)} \cdot (\boldsymbol{e}_{i,3} + \boldsymbol{e}'_1) + \boldsymbol{e}''_1$ is statistically close to $D_{\mathbb{Z}^m, \sqrt{(\bar{m}+nk)\cdot\gamma^2\cdot(4\beta^2+1)\cdot\alpha^2+\omega(\sqrt{\log m})}}$ by noticing that $\mathbb{Z}^m + \boldsymbol{c}'_{j,1} = \mathbb{Z}^m + R_{i,j}^{(1)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1$. Therefore, we have

$$\begin{aligned} \boldsymbol{c}'_{j,1} + \boldsymbol{e}''_1 &= R_{i,j}^{(1)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1 + \boldsymbol{e}''_1 + \boldsymbol{e}_{i,1} + B_j^T \cdot S_{i,j}^T \cdot \boldsymbol{s} + E_{i,j,1}^T \cdot \boldsymbol{s} \bmod q \\ &\overset{s}{\approx} \boldsymbol{e}'''_1 + \boldsymbol{e}_{i,1} + B_j^T \cdot S_{i,j}^T \cdot \boldsymbol{s} + E_{i,j,1}^T \cdot \boldsymbol{s} \bmod q \end{aligned}$$

for some $\boldsymbol{e}'''_1 \hookleftarrow D_{\mathbb{Z}^m, \sqrt{(\bar{m}+nk)\cdot\gamma^2\cdot(4\beta^2+1)\cdot\alpha^2+\omega(\sqrt{\log m})}}$. Similarly,

$$\boldsymbol{c}'_{j,2} + \boldsymbol{e}''_2 \overset{s}{\approx} \boldsymbol{e}'''_2 + \boldsymbol{e}_{i,2} + D_j^T \cdot S_{i,j}^T \cdot \boldsymbol{s} + E_{i,j,2}^T \cdot \boldsymbol{s} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q$$

for some $\boldsymbol{e}'''_2 \hookleftarrow D_{\mathbb{Z}^m, \sqrt{(\bar{m}+nk)\cdot\gamma^2\cdot(4\beta^2+1)\cdot\alpha^2+\omega(\sqrt{\log m})}}$. So, we could deduce that $-R_j^T \cdot \boldsymbol{c}_{j,1} + \boldsymbol{c}_{j,2} \bmod q = \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} + err_2$, where

$$err_2 \overset{s}{\approx} -R_j^T \cdot (E_{i,j,1}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,1} + \boldsymbol{e}'''_1) + E_{i,j,2}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,2} + \boldsymbol{e}'''_2 \in \mathbb{Z}^m.$$

By Lemma 5, it is easy to bound

$$||err_2||_\infty \leq 2 \cdot \omega(\log m) \cdot \sqrt{m} \cdot (1 + \sqrt{n} \cdot \alpha + \sqrt{5} \cdot \sqrt{\bar{m}+nk} \cdot \beta \cdot \gamma) \cdot \alpha$$

with overwhelming probability. For our choice of q, we also $||err||_\infty \leq \frac{q}{5}$ and we could recover $\boldsymbol{\mu}$ successfully. Overall, we could deduce the following lemma.

Lemma 9. *For $q \geq 10 \cdot \omega(\log m) \cdot \sqrt{m} \cdot (1 + \sqrt{n} \cdot \alpha + \sqrt{5} \cdot \sqrt{\bar{m}+nk} \cdot \beta \cdot \gamma) \cdot \alpha$, our single-hop PRE scheme satisfies correctness.*

Next, let's show the key-privacy and HRA security of our sing-hop PRE scheme. The key point of our design to achieve key-privacy is that for the challenge identity j^*, the re-encryption key generation algorithm does not use $A_{j^*,1}$ which offers the ability to generate re-encryption keys from j^* to other users. Therefore, it's enough to change B_{j^*} and D_{j^*} to be uniform, and maintain A_{j^*} unchanged.

Theorem 1. *Under the* $\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}}$ *assumption for* $q \geq 10 \cdot \omega(\log m) \cdot \sqrt{m} \cdot (1 + \sqrt{n} \cdot \alpha + \sqrt{5} \cdot \sqrt{\bar{m} + nk} \cdot \beta \cdot \gamma) \cdot \alpha$*, our single-hop PRE scheme is key private in the standard model.*

Proof. Assume that there is an adversary $\mathfrak{A}$ who could break the key-privacy of our single-hop PRE scheme with advantage $\delta = |\Pr[\mathfrak{A} \text{ wins}] - \frac{1}{2}|$. We consider the following sequence of games. Assume the sizes of U_H and U_C are $N_1 = \text{poly}(n)$ and $N_2 = \text{poly}(n)$. Notice that by Lemma 6, for any users i and j, after we sample $Q = \text{poly(n)}$ re-encryption keys $rk^{(k')}_{i \mapsto j}$ with $k' \in [Q]$, the new sampled re-encryption key satisfies $rk_{i \mapsto j} \neq rk^{(k')}_{i \mapsto j}$ for any $k' \in [Q]$ with overwhelming probability.

- **Game** 0**:** This is the original key-privacy game. Notice that, in the challenge query, when $b = 0$, the re-encryption key
$$rk_{i^* \mapsto j^*} = (\{X_{i',j'}; X'_{i',j'}\}_{i' \in \{1,2\}, j' \in \{1,2,3\}})$$
of challenge users i^* and j^* are sampled as $(R^{(1)}_{i^*,\bar{j}^*})^T, (R^{(2)}_{i^*,\bar{j}^*})^T \hookleftarrow D^m_{\mathbb{Z}^{\bar{m}+nk},\gamma}$, where $R^{(1)}_{i,j}, R^{(2)}_{i,j}$ are defined as those in the **ReEnc** algorithm
- **Game** 1**:** This game is almost identical to **Game** 0, with the differences that the challenger $\mathfrak{C}$ will select a random user $\bar{j}^* \in U_H$ after the adversary decided the sizes of U_H and U_C in Phase 1, and if the users (i^*, j^*) queried in the challenge oracle query satisfiy $j^* \neq \bar{j}^*$, $\mathfrak{C}$ aborts.
- **Game** 2**:** This game is almost identical to **Game** 1, with the differences that after the challenger $\mathfrak{C}$ decides $\bar{j}^*$, he chooses $B_{\bar{j}^*}, D_{\bar{j}^*} \hookleftarrow U(\mathbb{Z}_q^{n \times m})$.
- **Game** 3**:** This game is almost identical to **Game** 2, with the differences that in the challenge oracle query, the syndromes in the generation of re-encryption key $rk_{i^* \mapsto j^*}$ when $b = 1$ are changed to uniform elements.
- **Game** 4**:** This game is almost identical to **Game** 4, with the differences that in the challenge oracle query, $\mathfrak{C}$ outputs
$$rk_{i^* \mapsto \bar{j}^*} = (\{X_{i',j'}; X'_{i',j'}\}_{i' \in \{1,2\}, j' \in \{1,2,3\}})$$
with $(R^{(1)}_{i^*,\bar{j}^*})^T, (R^{(2)}_{i^*,\bar{j}^*})^T \hookleftarrow D^m_{\mathbb{Z}^{\bar{m}+nk},\gamma}$, regardless of the value of b.

If no abort occurs, **Game** 0 and **Game** 1 are identical. So, the advantage of $\mathfrak{A}$ is $\frac{\delta}{N_1}$ in **Game** 1. In the following, assume that no abort occurs. We also notice that *in the following games, we do not change the generation of* $A_{\bar{j}^*}$ *of the challenge user* $\bar{j}^*$*, so both re-encryption key generation queries and re-encryption queries from* $\bar{j}^*$ *to any user* $k \in U_H \cup U_C$ *could be answered correctly.* We will use $\text{Adv}_i(\mathfrak{A})$ to denote the advantage of $\mathfrak{A}$ in **Game** i for $i \in \{1, 2, 3, 4\}$.

Recall that we choose $\bar{m} \geq 2nk$ for $k = \lceil \log q \rceil$, so $m = \bar{m} + 2nk \geq 4n\lceil \log q \rceil$ and with overwhelming probability of $B_{\bar{j}^*} \hookleftarrow U(\mathbb{Z}_q^{n \times m})$, the distribution of $(B_{\bar{j}^*}, B_{\bar{j}^*} \cdot R_{\bar{j}^*})$ with $R_{\bar{j}^*} \hookleftarrow D^{m \times m}_{\mathbb{Z},s}$ is within statistical distance $\text{negl}(n)$ of $U(\mathbb{Z}_q^{n \times m} \times \mathbb{Z}_q^{n \times m})$ due to Lemma 1. So, we have $|\text{Adv}_1(\mathfrak{A}) - \text{Adv}_2(\mathfrak{A})| \leq \text{negl}(n)$.

In **Game** 2, the re-encryption key $rk_{i^* \mapsto \bar{j}^*} = (\{X_{i',j'}; X'_{i',j'}\}_{i' \in \{1,2\}, j' \in \{1,2,3\}})$ is generated satisfying

$$A'_{i^*,1} \cdot \begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,3} \\ X_{2,1} & X_{2,2} & X_{2,3} \end{pmatrix} = S_{i^*,\bar{j}^*} \cdot B_{\bar{j}^*} + E_{i^*,\bar{j}^*,1} - B_{i^*} \bmod q$$

and

$$A'_{i^*,1} \cdot \begin{pmatrix} X'_{1,1} & X'_{1,2} & X'_{1,3} \\ X'_{2,1} & X'_{2,2} & X'_{2,3} \end{pmatrix} = S_{i^*,\bar{j}^*} \cdot D_{\bar{j}^*} + E_{i^*,\bar{j}^*,2} - D_{i^*} \bmod q$$

for some $S_{i^*,\bar{j}^*} \hookleftarrow D_{\mathbb{Z},\alpha}^{n\times n}$ and $E_{i^*,\bar{j}^*,1}, E_{i^*,\bar{j}^*,2} \hookleftarrow D_{\mathbb{Z},\alpha}^{n\times m}$ when $b = 1$. Here, $A'_{i^*,1} = [\bar{A} | \bar{A} \cdot R_{\bar{i}^*,1} + H_1 \cdot \mathbf{G}]$. Therefore, under the Multi-LWE$^n_{n,q,2m,D_{\mathbb{Z},\alpha}}$ assumption, if we change $S_{i^*,\bar{j}^*} \cdot B_{\bar{j}^*} + E_{i^*,\bar{j}^*,1} - B_{i^*}$ and $S_{i^*,\bar{j}^*} \cdot D_{\bar{j}^*} + E_{i^*,\bar{j}^*,2} - D_{i^*}$ to be $U_1, U_2 \hookleftarrow U(\mathbb{Z}_q^m)$, the adversary could not distinguish the re-encryption keys generated from the above two cases. Hence, we get $|\mathrm{Adv}_2(\mathfrak{A}) - \mathrm{Adv}_3(\mathfrak{A})| \leq \mathrm{Adv}_{\mathfrak{A}}(\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}})$.

Discussions after Lemma 7 shows that the distribution of the re-encryption key $rk_{i^* \mapsto j^*}$ in **Game** 4 is statistically closed to that in **Game** 3 when $b = 1$. So, $|\mathrm{Adv}_3(\mathfrak{A}) - \mathrm{Adv}_4(\mathfrak{A})| \leq$ negl(n). Moreover, in **Game** 4, the distribution of re-encryption $rk_{i^* \mapsto j^*}$ is identical, regardless of the value of b. So, the advantage of $\mathfrak{A}$ in **Game** 4 is $\frac{1}{2}$. Overall, we deduce that

$$|\frac{\delta}{N_1} - \frac{1}{2}| \leq \mathrm{Adv}_{\mathfrak{A}}(\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}}) + \mathrm{negl}(n),$$

and conclude the desired result.

It is a little more complicated to show the HRA security. The *biggest obstacle* is how to answer re-encryption queries from i^* to other users (including those corrupted) after we change the generation of A_{i^*}. To do so, we add some "back door" (i.e. the $[\bar{A} \cdot R_{i^*,2} + h(\boldsymbol{\mu}) \cdot \mathbf{G}]$-part of $A_{i^*,1}$) to decrypt-then-encrypt corresponding ciphertexts.

Theorem 2. *Our single-hop PRE scheme satisfies* HRA *security under the* Multi-LWE$^n_{n,q,2m,D_{\mathbb{Z},\alpha}}$ *assumption and the* LWE$_{n,q,2m+\bar{m},D_{\mathbb{Z},\alpha}}$ *assumption in the standard model.*

Proof. In the following, we shall show that any PPT adversary could not distinguish the ciphertext of an arbitrary message (the case $b = 1$) and a random element chosen from the ciphertext space (the case $b = 0$). This implies our desired result. Assume that there is an adversary $\mathfrak{A}$ who could break the HRA security of our single-hop PRE scheme with advantage $\delta = |\Pr[\mathfrak{A} \text{ wins}] - \frac{1}{2}|$. We consider the following sequence of games. Assume the sizes of U_H and U_C are $N_1 = \mathrm{poly}(n)$ and $N_2 = \mathrm{poly}(n)$.

- **Game** 0**:** This is the original HRA security game.
- **Game** 1**:** This game is almost identical to **Game** 0, with the differences that the challenger $\mathfrak{C}$ will select a random user $\bar{i}^* \in U_H$ after the adversary decided the sizes of U_H and U_C in Phase 1, and if the challenge users (i^*, j^*) queried in challenge oracle query satisfies $i^* \neq \bar{i}^*$, $\mathfrak{C}$ aborts.

We will also use $\mathrm{Adv}_i(\mathfrak{A})$ to denote the advantage of adversary $\mathfrak{A}$ in **Game** i for $i \in [8]$, then we have $\mathrm{Adv}_1(\mathfrak{A}) = \frac{\delta}{N_1}$. Assume that in the following games no abort occurs.

- **Game 2:** This game is almost identical to **Game** 1, with the differences that after the challenger $\mathfrak{C}$ decides $\bar{i}^*$, he changes all the $\{B_i, D_i\}$'s of honest users $i \in U_H$ to be uniformly random elements of $\mathbb{Z}_q^{n\times m}$.

We note that *the* $\{A_i\}$*'s of every user* $i \in U_H$ *is unchanged now, so re-encryption key generation queries, hence re-encryption queries, from user* $i \in U_H$ *to any user* $j \in U_H \cup U_C$ *could be answered correctly.* By Lemma 1, we have $|\mathrm{Adv}_1(\mathfrak{A}) - \mathrm{Adv}_2(\mathfrak{A})| \leq N_1 \cdot \mathrm{negl}(n)$.

- **Game 3:** This game is almost identical to **Game** 2, with the differences that when generating re-encryption keys $rk_{\bar{i}^* \mapsto j}$ with $j \in U_H$, we changes the syndromes to be uniformly random elements as in **Game** 3 of the proof of Theorem 1.

Since there are at most $N_1 - 1$ such queries, under Multi-LWE$^n_{n,q,2m,D_{\mathbb{Z},\alpha}}$ assumption, we get $|\mathrm{Adv}_2(\mathfrak{A}) - \mathrm{Adv}_3(\mathfrak{A})| \leq (N_1 - 1) \cdot \mathrm{Adv}_{\mathfrak{A}}(\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}})$.

- **Game 4:** This game is almost identical to **Game** 3, with the differences that when generating re-encryption keys $rk_{\bar{i}^* \mapsto j}$ with $j \in U_H$, we just generate the re-encryption key

  $$rk_{\bar{i}^* \mapsto j} = (\{X_{i',j'}; X'_{i',j'}\}_{i' \in \{1,2\}, j' \in \{1,2,3\}})$$

  by sampling $(R^{(1)}_{i,j})^T, (R^{(2)}_{i,j})^T \hookleftarrow D^m_{\mathbb{Z}^{\bar{m}+nk},\gamma}$, where $R^{(1)}_{i,j}, R^{(2)}_{i,j}$ are defined as those in the **ReEnc** algorithm.

Similar to discussion in Theorem 1, we have $|\mathrm{Adv}_3(\mathfrak{A}) - \mathrm{Adv}_4(\mathfrak{A})| \leq (N_1 - 1) \cdot \mathrm{negl}(n)$. *Notice that, from now on, we do not need to use the trapdoor embedded in* $A_{\bar{i}^*}$ *to generate re-encryption keys from* $\bar{i}^*$ *to honest users.*

- **Game 5:** This game is almost identical to **Game** 4, with the differences that after the challenger $\mathfrak{C}$ decides $\bar{i}^*$, he chooses a $H^* \hookleftarrow U(\mathfrak{S})$ which will be used as the tag of the encryption of the challenge message $\boldsymbol{\mu}^*$, and set $A_{\bar{i}^*} = [\bar{A}|\bar{A} \cdot R_{\bar{i}^*,1} - H_1 \cdot \mathbf{G}|\bar{A} \cdot R_{\bar{i}^*,2} - H^* \cdot \mathbf{G}]$. Notice that, the public key of $\bar{i}^*$ is $(A_{\bar{i}^*}, B_{\bar{i}^*}, D_{\bar{i}^*})$ with $B_{\bar{i}^*}, D_{\bar{i}^*} \hookleftarrow U(\mathbb{Z}_q^{n\times m})$ now. Meanwhile, the re-encryption queries from $\bar{i}^*$ to $j \in U_C$ will be replaced as follows.
  - When receiving query $(\bar{i}^*, j \in U_C, (\boldsymbol{c}_1, \boldsymbol{c}_2, \boldsymbol{c}_3; H_{\boldsymbol{c}}, 1))$, $\mathfrak{C}$ will abort if $H_{\boldsymbol{c}} = H^*$. Otherwise, we have $A_{\bar{i}^*,1} = [\bar{A}|\bar{A} \cdot R_{\bar{i}^*,1}|\bar{A} \cdot R_{\bar{i}^*,2} + (h(\boldsymbol{\mu}) - H^*) \cdot \mathbf{G}]$. By Lemma 7, the challenge $\mathfrak{C}$ could recover $\boldsymbol{s}$ used in the encryption algorithm from $\boldsymbol{c}_{i,3}$ and $[\bar{A}|\bar{A} \cdot R_{\bar{i}^*,2} + (H_{\boldsymbol{c}} - H^*) \cdot \mathbf{G}]$ with trapdoor $R_{\bar{i}^*,2}$, since $\frac{q}{\alpha} > \sqrt{n \cdot \log q} \cdot \omega(\sqrt{\log n})$. With the recovered $\boldsymbol{s}$, $\mathfrak{C}$ could recover errors $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2}, \boldsymbol{e}_{i,3}$, as well as $\boldsymbol{\mu}$, by our choices of parameters. Therefore, the challenger $\mathfrak{C}$ could sample $S_{\bar{i}^*,j} \hookleftarrow \chi^{n\times n}$, $E_{\bar{i}^*,j,1}, E_{\bar{i}^*,j,2} \hookleftarrow \chi^{n\times m}$

and $\boldsymbol{e}_1''', \boldsymbol{e}_2''' \hookleftarrow D_{\mathbb{Z}^m, \sqrt{(\bar{m}+nk)\cdot(4\beta^2+1)\cdot\gamma^2\cdot\alpha^2+\omega(\sqrt{\log m})}}$. Then, he returns $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \perp; \boldsymbol{\mu}, 2)$ to $\mathfrak{A}$, where

$$\boldsymbol{c}_{j,1} = \boldsymbol{e}_1''' + \boldsymbol{e}_{i,1} + B_j^T \cdot S_{\bar{i}^*,j}^T \cdot \boldsymbol{s} + E_{\bar{i}^*,j,1}^T \cdot \boldsymbol{s} \bmod q$$

and

$$\boldsymbol{c}_{j,2} = \boldsymbol{e}_2''' + \boldsymbol{e}_{i,2} + D_j^T \cdot S_{\bar{i}^*,j}^T \cdot \boldsymbol{s} + E_{\bar{i}^*,j,2}^T \cdot \boldsymbol{s} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q.$$

Assume that there is at most $Q = \mathrm{poly}(n)$ queries of the encryption oracle, then the probability that $\mathfrak{C}$ aborts in the above modified re-encryption query oracle is less than $\frac{Q}{|\mathfrak{S}|} = \mathrm{negl}(n)$. Since in our security game, encryptions are guaranteed to be honest-generated. As long as no abort occurs, the outputted re-encryptions are distributed statistically closed to a real re-encrypted ciphertext according to our analysis in the correctness (above Lemma 9). So, we have $|\mathrm{Adv}_4(\mathfrak{A}) - \mathrm{Adv}_5(\mathfrak{A})| \leq \mathrm{negl}(n)$.

- **Game 6:** This game is almost identical to **Game** 5, with the differences that we change the generation of the challenge ciphertext when $b = 1$ as follows.
  - To encrypt the a message $\boldsymbol{\mu}^*$, $\mathfrak{C}$ first samples $\boldsymbol{s} \hookleftarrow \chi^n$, $\boldsymbol{e}_{\bar{i}^*,1}, \boldsymbol{e}_{\bar{i}^*,2} \hookleftarrow \chi^m$ and $\boldsymbol{e}_{\bar{i}^*,3,1} \hookleftarrow \chi^{\bar{m}}$. Then, he computes $\boldsymbol{c}_{\bar{i}^*,1} = B_{\bar{i}^*}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,1} \bmod q$, $\boldsymbol{c}_{\bar{i}^*,2} = D_{\bar{i}^*}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}^* \bmod q$ and $\boldsymbol{c}^* = \bar{A}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,3,1} \bmod q$. Finally, he computes
$$\boldsymbol{c}_{\bar{i}^*,3} = \mathrm{ReRand}([I_{\bar{m}\times\bar{m}}|R_{\bar{i}^*,1}|R_{\bar{i}^*,2}], \boldsymbol{c}^*, \alpha, \beta)$$
    by using Lemma 3, and returns $(\boldsymbol{c}_{\bar{i}^*,1}, \boldsymbol{c}_{\bar{i}^*,2}, \boldsymbol{c}_{\bar{i}^*,3}; H^*, 1)$ to $\mathfrak{A}$.

Recall that the challenge ciphertext is of the form $\boldsymbol{c}_{\bar{i}^*,1} = B_{\bar{i}^*}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,1} \bmod q$, $\boldsymbol{c}_{\bar{i}^*,2} = D_{\bar{i}^*}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}^* \bmod q$, and

$$\boldsymbol{c}_{\bar{i}^*,3} = [\bar{A}|\bar{A}\cdot R_{\bar{i}^*,1}|\bar{A}\cdot R_{\bar{i}^*,2}]^T \cdot \boldsymbol{s} + \boldsymbol{e}_{\bar{i}^*,3} \bmod q$$

for the case $b = 1$ in **Game** 5. Here, $\boldsymbol{s} \hookleftarrow \chi^n$, $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2} \hookleftarrow \chi^m$ and $\boldsymbol{e}_{i,3} \hookleftarrow D_{\mathbb{Z}^m, 2\cdot\beta\cdot\alpha}$. According to our choice of parameters and Lemma 4, we have $\beta \geq 1 + \mathfrak{s}_1([R_{\bar{i}^*,1}|R_{\bar{i}^*,2}]) \geq \mathfrak{s}_1([I_{\bar{m}\times\bar{m}}|R_{\bar{i}^*,1}|R_{\bar{i}^*,2}])$ with overwhelming probability. So, Lemma 3 implies that the distribution of $\boldsymbol{c}_{\bar{i}^*,3}$ in **Game** 5 is statistically closed to the distribution of $\boldsymbol{c}_{\bar{i}^*,3}$ generated in the above modified encryption algorithm in **Game** 6. Therefore, we have $|\mathrm{Adv}_5(\mathfrak{A}) - \mathrm{Adv}_6(\mathfrak{A})| \leq \mathrm{negl}(n)$.

- **Game 7:** This game is almost identical to **Game** 6, with the differences that we change the generation of challenge ciphertext when $b = 1$ as follows. To encrypt the challenged message $\boldsymbol{\mu}^*$, $\mathfrak{C}$ sets $\boldsymbol{c}_{\bar{i}^*,1} = \boldsymbol{u}_1 \hookleftarrow U(\mathbb{Z}_q^m)$, $\boldsymbol{c}_{\bar{i}^*,2} = \boldsymbol{u}_2 + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu}^*$ with $\boldsymbol{u}_2 \hookleftarrow U(\mathbb{Z}_q^m)$ and $\boldsymbol{c}^* = \boldsymbol{u}_3 + \boldsymbol{e}_{\bar{i}^*,3,1}$ with $\boldsymbol{u}_3 \hookleftarrow U(\mathbb{Z}_q^{\bar{m}})$ and $\boldsymbol{e}_{\bar{i}^*,3,1} \hookleftarrow \chi^{\bar{m}}$. He computes $\boldsymbol{c}_{\bar{i}^*,3} = \mathrm{ReRand}([I_{\bar{m}\times\bar{m}}|R_{\bar{i}^*,1}|R_{\bar{i}^*,2}], \boldsymbol{c}^*, \alpha, \beta)$ and returns $(\boldsymbol{c}_{\bar{i}^*,1}, \boldsymbol{c}_{\bar{i}^*,2}, \boldsymbol{c}_{\bar{i}^*,3}; H^*, 1)$ to $\mathfrak{A}$ in the end.

The main difference between **Game** 7 and **Game** 6 is the distributions of $\boldsymbol{c}_{\bar{i}^*,1}, \boldsymbol{c}_{\bar{i}^*,2}$ and $\boldsymbol{c}^*$, that differ by some LWE samples. Hence, it is easy to verify that $|\mathrm{Adv}_6(\mathfrak{A}) - \mathrm{Adv}_7(\mathfrak{A})| \leq \mathrm{Adv}_{\mathfrak{A}}(\mathrm{LWE}_{n,q,2m+\bar{m},D_{\mathbb{Z},\alpha}})$.

- **Game** 8**:** This game is almost identical to **Game** 7, with the differences that we change the generation of challenge ciphertext when $b = 1$ as follows. To encrypt the challenged message $\boldsymbol{\mu}^*$, $\mathfrak{C}$ sets $\boldsymbol{c}_{\bar{i}^*,1}, \boldsymbol{c}_{\bar{i}^*,2}, \boldsymbol{c}_{\bar{i}^*,3} \hookleftarrow U(\mathbb{Z}_q^m)$.

Notice that we have that $\boldsymbol{c}_{\bar{i}^*,3} = [I_{\bar{m}\times\bar{m}}|R_{\bar{i}^*,1}|R_{\bar{i}^*,2}]^T \cdot \boldsymbol{u}_3 + \boldsymbol{e}'$ for some $\boldsymbol{e}' \overset{s}{\hookleftarrow} D_{\mathbb{Z}^m,2\beta\cdot\alpha}$ in **Game** 7 by Lemma 3. Meanwhile, the entire view of $(\bar{A}, \boldsymbol{u}; \bar{A} \cdot R_{\bar{i}^*,k'}, \boldsymbol{u}^T \cdot R_{\bar{i}^*,k'}) \overset{s}{\approx} (\bar{A}, \boldsymbol{u}; U', (\boldsymbol{u}')^T) \overset{s}{\approx} (\bar{A}, \boldsymbol{u}; \bar{A} \cdot R_{\bar{i}^*,k'}, (\boldsymbol{u}')^T)$ for $k' \in [2]$, $U' \hookleftarrow U(\mathbb{Z}_q^{n\times nk})$, $\boldsymbol{u} \hookleftarrow U(\mathbb{Z}_q^{\bar{m}})$ and $\boldsymbol{u}' \hookleftarrow U(\mathbb{Z}_q^{nk})$ by Lemma 1. Therefore, we have $|\mathrm{Adv}_7(\mathfrak{A}) - \mathrm{Adv}_8(\mathfrak{A})| \leq \mathrm{negl}(n)$.

Finally, notice that we have $|\mathrm{Adv}_8(\mathfrak{A})| = \frac{1}{2}$. We could deduce that

$$\begin{aligned}|\frac{\delta}{N_1} - \frac{1}{2}| \leq & \mathrm{negl}(n) + (N_1 - 1) \cdot \mathrm{Adv}_{\mathfrak{A}}(\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}}) \\ & + \mathrm{Adv}_{\mathfrak{A}}(\mathrm{LWE}_{n,q,2m+\bar{m},D_{\mathbb{Z},\alpha}}).\end{aligned}$$

The proof is finished.

Remark 6. We note that for our constructions of single-hop PRE schemes, the encryption oracle only needs to record encryptions of the challenge user i^* in the security game. I.e. we only need to make sure that ciphertexts of the user i^* are not malformed. This is because by our constructions, the adversary could not re-encrypt a level 2 ciphertext of an honest user to any other corrupted users, since the form of level 2 ciphertext is different from those of fresh ciphertexts.

Remark 7. Since our constructions satisfy the requirements ((weakly) key privacy, source hiding, indistinguishability) proposed in [13], our single-hop PRE schemes are also *adaptive* secure for some directed acyclic graphs *with polynomially bounded parameters* by Theorem 6 of [13].

Remark 8. As [28], our single-hop PRE scheme could be easily modified to a PRE+ scheme, in which the re-encryption keys of a ciphertext of any message could be generated by its encrypter. For LWE-based constructions, the intuition is quite easy. Any encrypter of a message m knows the ephemeral secrets $\boldsymbol{s}$ and $\boldsymbol{e}$'s. So, the corresponding re-encryption key generation algorithm and the re-encryption algorithm could be designed as follows.

- **ReKeyGen**($pp, pk_i, pk_j, \boldsymbol{ct}_i; eph$): Sample $\boldsymbol{e}_{i,j,1}, \boldsymbol{e}_{i,j,2} \hookleftarrow \chi^m$, then return $rk_{i\mapsto j}(\boldsymbol{ct}_i) = ((B_j - B_i) \cdot \boldsymbol{s} + \boldsymbol{e}_{i,j,1}; (D_j - D_i) \cdot \boldsymbol{s} + \boldsymbol{e}_{i,j,2})$.
- **ReEnc**($pp, \boldsymbol{ct}_i, rk_{i\mapsto j}(\boldsymbol{ct}_i)$): Parse $rk_{i\mapsto j}(\boldsymbol{ct}_i) = (rk_1; rk_2)$, then return $\boldsymbol{c}_{j,1} = \boldsymbol{c}_{i,1} + rk_1$ and $\boldsymbol{c}_{j,2} = \boldsymbol{c}_{i,2} + rk_1$.

Here, *eph* represents the ephemeral secret $\boldsymbol{s}$.

Recall that as discussed in Remark 2, our sing-hop PRE schemes also satisfies the second-level security of ciphertexts [12]. In the above security proof, the challenger has no ability to answer re-encryption queries of non-challenge ciphertexts with tag H^* from i^* to corrupted users. By our definition of security, ciphertexts of i^* are generated in the encryption oracle. So, this case will happen with negligible-probability. While in [12], this case is avoided by adding an additional not-derivative requirement [12, Definition 3.2]. Meanwhile, we note that in the secure proof of [12, Theorem 4.2], the distributions of re-encryption keys between (i^*, j) with $j \in U_H$ and (i, j) with $i \in U_H \backslash \{i^*\}$ and $j \in U_H \cup U_C$ are different from each other (due to the quality of trapdoors used for sampling preimages), which may result in that the norms of re-encryption keys of the later case would be larger than the desired (Gaussian) bound as in the real schemes. This means that the statistical distance of Hybrid H_0 and Hybrid H_1 in the security proof of [12] is not negligible, so the adversary may output $\perp$ indicating that he detects the scheme is different from the real scheme.

3.3 Multi-Hop PRE Schemes with HRA Security

Our constructions of single-hop PRE schemes could be easily extended to multi-hop PRE schemes with HRA security (only). Note that in the re-encryption key generation algorithm of our sing-hop PRE constructions, $S_{i,j}$ and $\{E_{i,j,k}\}_{k\in[2]}$ are used to randomize the syndromes (and are mainly used to show the key privacy). In this subsection, we use the following parameters.

- The modulus q is a prime, $\mathfrak{S} \in \mathbb{Z}_q^{n\times n}$ is the set with unit differences property, $L = O(1)$ is the maximum times of re-encryptions.
- Integers $k = \lceil \log q \rceil$, $\bar{m} \geq 2nk$, $m = \bar{m} + 2nk$.
- $s = \omega(\sqrt{\log m}) \geq \max\{\eta_\varepsilon(\mathbb{Z}^m), 6\}$ is some fixed function.
- $\beta = C \cdot s \cdot \sqrt{\bar{m}} + 1$ with C the global constant appeared in Lemma 4; $\gamma = O(\sqrt{n \cdot \log q}) \cdot \omega(\log m)$ with the O function appeared in Lemma 7.
- $\chi = D_{\mathbb{Z},\alpha}$ is the error distribution of corresponding (single-secret/multi-secrets) LWE problems with $\alpha \geq 2\sqrt{n} \geq 2 \cdot \omega(\sqrt{\log m})$.

As before, $\mathbf{G} \in \mathbb{Z}^{n\times nk}$ is the gadget matrix. To ensure the correctness of our L-hop PRE schemes, we require

$$q \geq 5 \cdot \omega(\log^{\frac{L}{2}} m) \cdot \sqrt{m} \cdot (\bar{m} + nk)^{\frac{L}{2}} \cdot \alpha \cdot \beta \cdot \gamma^L$$

in general. The detailed construction of our L-hop PRE scheme is as following.

- **Setup**(1^λ) : This algorithm uses large enough integer n (e.g. $n \geq 512$) and parameters satisfying the above relations. It samples $\bar{A} \hookleftarrow U(\mathbb{Z}_q^{n\times \bar{m}})$, $\{H_i\}_{i\in[L]} \hookleftarrow U(\mathfrak{S})$ such that $H_i \neq H_j$ for any $i \neq j \in [L]$, and outputs the public parameters
$$pp = (n, q, \bar{m}, \chi, D_{\mathbb{Z},s}, h; \mathbf{G}; \bar{A}; \{H_i\}_{i\in[L]}).$$
- **KeyGen**(pp) : To generate a public/secret key pair of a user i, this algorithm, implemented by the key management center, samples matrices $B_i \hookleftarrow U(\mathbb{Z}_q^{n\times m})$, $R_i \hookleftarrow D_{\mathbb{Z},s}^{m\times m}$ and $R_{i,1}, R_{i,2} \hookleftarrow D_{\mathbb{Z},s}^{\bar{m}\times nk}$. Then, it computes $A_i = [\bar{A}|\bar{A} \cdot R_{i,1}|\bar{A} \cdot R_{i,2}] \in \mathbb{Z}_q^{n\times m}$, $D_i = B_i \cdot R_i \in \mathbb{Z}_q^{n\times m}$ and returns
$$(pk_i, sk_i) = ((A_i, B_i, D_i), (R_i, R_{i,1}, R_{i,2})).$$

- $\mathbf{Enc}(pp, pk_i, \boldsymbol{\mu})$: To encrypt a message $\boldsymbol{\mu} \in \{0,1\}^m$ under the public key pk_i of a user i, this algorithm first samples $\boldsymbol{s} \hookleftarrow \chi^n$ [19] and errors $\boldsymbol{e}_{i,1}, \boldsymbol{e}_{i,2} \hookleftarrow \chi^m$. It also samples error $\boldsymbol{e}_{i,3} \hookleftarrow D_{\mathbb{Z}^m, 2\cdot\beta\cdot\alpha}$ and $H_{\boldsymbol{c}_i} \hookleftarrow U(\mathfrak{S})$. Then, it sets the level matrix

$$\begin{aligned} A_{i,1} &= A_i + [0|H_1 \cdot \mathbf{G}|H_{\boldsymbol{c}_i} \cdot \mathbf{G}] \\ &= [\bar{A}|\bar{A} \cdot R_{i,1} + H_1 \cdot \mathbf{G}|\bar{A} \cdot R_{i,2} + H_{\boldsymbol{c}_i} \cdot \mathbf{G}] \in \mathbb{Z}_q^{n \times m}, \end{aligned}$$

  and computes $\boldsymbol{c}_{i,1} = B_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,1} \bmod q$, $\boldsymbol{c}_{i,2} = D_i^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,2} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q$, $\boldsymbol{c}_{i,3} = A_{i,1}^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,3} \bmod q$. Finally, it outputs $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}; H_{\boldsymbol{c}_i}, 1) \in \mathbb{Z}_q^m \times \mathbb{Z}_q^m \times \mathbb{Z}_q^m \times \mathbb{Z}_q^{n\times n} \times [L+1]$.
- $\mathbf{ReKeyGen}(pp, sk_i, pk_j, \kappa)$: For $\kappa \leq L$, the user i samples $X_{1,1} \in \mathbb{Z}^{\bar{m}\times\bar{m}}$, $X_{1,2}, X_{1,3} \in \mathbb{Z}^{\bar{m}\times nk}$, $X_{2,1} \in \mathbb{Z}^{nk\times\bar{m}}$, and $X_{2,2}, X_{2,3} \in \mathbb{Z}^{nk\times nk}$ by using algorithm **SampleD** with $R_{i,1}$ and γ, such that

$$A_{i,\kappa} \cdot \begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,3} \\ X_{2,1} & X_{2,2} & X_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix} = B_j - B_i \bmod q \text{ [20]}$$

  with $A_{i,\kappa} = [\bar{A}|\bar{A} \cdot R_{i,1} + H_\kappa \cdot \mathbf{G}|\bar{A} \cdot R_{i,2} + h(\boldsymbol{\mu}) \cdot \mathbf{G}]$. Similarly, he also samples matrices $X'_{1,1} \in \mathbb{Z}^{\bar{m}\times\bar{m}}, X'_{1,2}, X'_{1,3} \in \mathbb{Z}^{\bar{m}\times nk}, X'_{2,1} \in \mathbb{Z}^{nk\times\bar{m}}$, and $X'_{2,2}, X'_{2,3} \in \mathbb{Z}^{nk\times nk}$ by using algorithm **SampleD** with $R_{i,1}$ and γ, such that

$$A_{i,\kappa} \cdot \begin{pmatrix} X'_{1,1} & X'_{1,2} & X'_{1,3} \\ X'_{2,1} & X'_{2,2} & X'_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix} = D_j - D_i \bmod q.$$

  If $\kappa \leq L-1$, he then samples $X''_{1,2}, X''_{1,3} \in \mathbb{Z}^{\bar{m}\times nk}$ and $X''_{2,2}, X''_{2,3} \in \mathbb{Z}^{nk\times nk}$ by using algorithm **SampleD** with $R_{i,1}$ and γ, such that

$$\begin{aligned} &[\bar{A}|\bar{A} \cdot R_{i,1} + H_\kappa \cdot \mathbf{G}|\bar{A} \cdot R_{i,2} + B] \cdot \begin{pmatrix} I_{\bar{m}\times\bar{m}} & X''_{1,2} & X''_{1,3} \\ 0_{nk\times\bar{m}} & X''_{2,2} & X''_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & I_{nk\times nk} \end{pmatrix} \\ &= [\bar{A}|\bar{A} \cdot R_{j,1} + H_{\kappa+1} \cdot \mathbf{G}|\bar{A} \cdot R_{j,2} + B] \end{aligned} \tag{1}$$

  for any matrix $B \in \mathbb{Z}_q^{n\times nk}$. In the end, he returns the corresponding re-encryption key

$$rk_{i\mapsto j}^{\kappa} = (\{X_{i',j'}; X'_{i',j'}\}_{i'\in\{1,2\}, j'\in\{1,2,3\}}; \{X''_{i',j'}\}_{i',j'\in\{2,3\}}; \kappa)$$

  for $\kappa \leq L-1$. If $\kappa = L$, he only returns

$$rk_{i\mapsto j}^{\kappa} = (\{X_{i',j'}; X'_{i',j'}\}_{i'\in\{1,2\}, j'\in\{1,2,3\}}; \perp; L).$$

[19] Since we only consider HRA security, $\boldsymbol{s} \hookleftarrow U(\mathbb{Z}_q^n)$ is also sufficient.

[20] In fact, B_j is enough.

- $\mathbf{ReEnc}(pp, \boldsymbol{ct}_i, rk^{\kappa}_{i\mapsto j})$: The proxy parses $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3}; H_{\boldsymbol{c}_i}, \kappa')$ and $rk^{\kappa}_{i\mapsto j} = (\{X_{i',j'}; X'_{i',j'}\}_{i'\in\{1,2\}, j'\in\{1,2,3\}}; *; \kappa)$. If $\boldsymbol{ct}_i$ does not equal to this form, or $\kappa \neq \kappa'$, return $\perp$. The proxy sets

$$R^{(1)}_{i,j} = \begin{pmatrix} X_{1,1} & X_{1,2} & X_{1,3} \\ X_{2,1} & X_{2,2} & X_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix}^T, R^{(2)}_{i,j} = \begin{pmatrix} X'_{1,1} & X'_{1,2} & X'_{1,3} \\ X'_{2,1} & X'_{2,2} & X'_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & 0_{nk\times nk} \end{pmatrix}^T.$$

with the help of $rk^{\kappa}_{i\mapsto j}$. If $\kappa \leq L-1$, he also sets

$$R^{(3)}_{i,j} = \begin{pmatrix} I_{\bar{m}\times\bar{m}} & X''_{1,2} & X''_{1,3} \\ 0_{nk\times\bar{m}} & X''_{2,2} & X''_{2,3} \\ 0_{nk\times\bar{m}} & 0_{nk\times nk} & I_{nk\times nk} \end{pmatrix}^T,$$

and sample $H_{\boldsymbol{c}_j} \hookleftarrow U(\mathfrak{S})$.

  - If $\kappa = 1$, he first computes

$$\boldsymbol{c}'_{j,1} := R^{(1)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1} + R^{(1)}_{i,j} \cdot \boldsymbol{e}'_1 \bmod q,$$

$$\boldsymbol{c}'_{j,2} := R^{(2)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,2} + R^{(2)}_{i,j} \cdot \boldsymbol{e}'_2 \bmod q,$$

and

$$\boldsymbol{c}'_{j,3} := R^{(3)}_{i,j} \cdot \boldsymbol{c}_{i,3} + R^{(3)}_{i,j} \cdot \boldsymbol{e}'_3 \bmod q.$$

Here, $\boldsymbol{e}'_{k'} = ((\boldsymbol{e}'_{k',1})^T, (\boldsymbol{e}'_{k',2})^T, \boldsymbol{0}^T_{nk})^T \in \mathbb{R}^m$ with $\boldsymbol{e}'_{k',1} \hookleftarrow D^{\bar{m}}_{\alpha}$ and $\boldsymbol{e}'_{k',2} \hookleftarrow D^{nk}_{\alpha}$ for $k' \in \{1,2\}$; $\boldsymbol{e}'_3 = ((\boldsymbol{e}'_{3,1})^T, (\boldsymbol{e}'_{3,2})^T, (\boldsymbol{e}'_{3,3})^T) \in \mathbb{R}^m$ with $\boldsymbol{e}'_{3,1} \hookleftarrow D^{\bar{m}}_{2\beta\cdot\alpha}$ and $\boldsymbol{e}'_{3,2}, \boldsymbol{e}'_{3,3} \hookleftarrow D^{nk}_{\alpha}$. Then, he computes $\boldsymbol{c}_{j,1} = \boldsymbol{c}'_{j,1} + \boldsymbol{e}''_1 \bmod q$, $\boldsymbol{c}_{j,2} = \boldsymbol{c}'_{j,2} + \boldsymbol{e}''_2 \bmod q$ and $\boldsymbol{c}_{j,3} = \boldsymbol{c}'_{j,3} + \boldsymbol{e}''_3 \bmod q$. Here, $\boldsymbol{e}''_1 \hookleftarrow D_{\mathbb{Z}^m - \boldsymbol{c}'_{j,1}, \boldsymbol{r}}$ with

$$\boldsymbol{r}_{k'} = \sqrt{((\bar{m}+nk)\cdot\gamma^2 - ||R^{(1)}_{i,j,k'}||^2)\cdot(4\beta^2+1)\cdot\alpha^2 + \omega(\sqrt{\log m})}$$

for $k' \in [m]$, $\boldsymbol{e}''_2 \hookleftarrow D_{\mathbb{Z}^m - \boldsymbol{c}'_{j,2}, \boldsymbol{r}'}$ with

$$\boldsymbol{r}'_{k'} = \sqrt{((\bar{m}+nk)\cdot\gamma^2 - ||R^{(2)}_{i,j,k'}||^2)\cdot(4\beta^2+1)\cdot\alpha^2 + \omega(\sqrt{\log m})}$$

for $k' \in [m]$, and $\boldsymbol{e}''_3 \hookleftarrow D_{\mathbb{Z}^m - \boldsymbol{c}'_{j,3}, \boldsymbol{r}''}$ with

$$\boldsymbol{r}''_{k'} = \sqrt{((\bar{m}+nk)\cdot\gamma^2 + 1 - ||R^{(3)}_{i,j,k'}||^2)\cdot(4\beta^2+1)\cdot\alpha^2 + \omega(\sqrt{\log m})}$$

for $k' \in [m]$. Finally, he returns $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \boldsymbol{c}_{j,3}; H_{\boldsymbol{c}_j}, 2)$.

  - If $2 \leq \kappa \leq L-1$, the proxy simply computes $\boldsymbol{c}_{j,1} := R^{(1)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1} \bmod q$, $\boldsymbol{c}_{j,2} := R^{(2)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,2} \bmod q$, $\boldsymbol{c}_{j,3} := R^{(3)}_{i,j} \cdot \boldsymbol{c}_{i,3} \bmod q$, and returns $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \boldsymbol{c}_{j,3}; H_{\boldsymbol{c}_j}, \kappa+1)$.
  - If $\kappa = L$, the proxy simply computes $\boldsymbol{c}_{j,1} := R^{(1)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,1} \bmod q$, $\boldsymbol{c}_{j,2} := R^{(2)}_{i,j} \cdot \boldsymbol{c}_{i,3} + \boldsymbol{c}_{i,2} \bmod q$, and returns $(\boldsymbol{c}_{j,1}, \boldsymbol{c}_{j,2}, \perp; \perp, L+1)$.

  - Otherwise, return $\perp$.

- **Dec**$(pp, \boldsymbol{ct}_i, sk_i)$: To decrypt a ciphertext of a user i, we first parse $\boldsymbol{ct}_i = (\boldsymbol{c}_{i,1}, \boldsymbol{c}_{i,2}, \boldsymbol{c}_{i,3} \text{ or } \perp; h(\boldsymbol{\mu}), \kappa)$ with $\kappa \in [L+1]$, compute $\boldsymbol{\mu}' = -R_i^T \cdot \boldsymbol{c}_{i,1} + \boldsymbol{c}_{i,2} \bmod q$, and return $\boldsymbol{\mu}$ with $\boldsymbol{\mu}_i = \lfloor \frac{2}{q} \cdot \boldsymbol{\mu}'_i \rceil$ for $i \in [m]$.

To show the correctness, we notice that the decryption of a fresh (i.e. level 1) ciphertext is the same as the sing-hop case. Meanwhile, for a level 2 ciphertext, similar analyses as those above Lemma 9 show that

$$\begin{aligned} \boldsymbol{c}_{j,1} &= R_{i,j}^{(1)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'_1 + \boldsymbol{e}''_1 + \boldsymbol{e}_{i,1} + B_j^T \cdot \boldsymbol{s} \bmod q \\ &\overset{s}{\approx} \boldsymbol{e}'''_1 + \boldsymbol{e}_{i,1} + B_j^T \cdot \boldsymbol{s} \bmod q \end{aligned}$$

and

$$\begin{aligned} \boldsymbol{c}_{j,2} &= R_{i,j}^{(2)} \cdot \boldsymbol{e}_{i,3} + R_{i,j}^{(2)} \cdot \boldsymbol{e}'_2 + \boldsymbol{e}''_2 + \boldsymbol{e}_{i,2} + D_j^T \cdot \boldsymbol{s} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q \\ &\overset{s}{\approx} \boldsymbol{e}'''_2 + \boldsymbol{e}_{i,2} + D_j^T \cdot \boldsymbol{s} + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q \end{aligned}$$

with

$$\boldsymbol{e}'''_1, \boldsymbol{e}'''_2 \hookleftarrow D_{\mathbb{Z}^m, \sqrt{(\bar{m}+nk)\cdot\gamma^2\cdot(4\beta^2+1)\cdot\alpha^2+\omega(\sqrt{\log m})}}. \tag{2}$$

Therefore, we have $-R_j^T \cdot \boldsymbol{c}_{j,1} + \boldsymbol{c}_{j,2} \bmod q = \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} + err$, where

$$err \overset{s}{\approx} -R_j^T \cdot (\boldsymbol{e}_{i,1} + \boldsymbol{e}'''_1) + \boldsymbol{e}_{i,2} + \boldsymbol{e}'''_2 \in \mathbb{Z}^m.$$

With overwhelming probability, we get

$$||err||_\infty \leq \omega(\log m) \cdot \sqrt{m \cdot (\bar{m} + nk)} \cdot \gamma \cdot \alpha \cdot \beta.$$

Besides, we also have $\boldsymbol{c}_{j,3} \overset{s}{\approx} A_{j,2}^T \cdot \boldsymbol{s} + \boldsymbol{e}'''_3$ with

$$\boldsymbol{e}'''_3 \hookleftarrow D_{\mathbb{Z}^m, \sqrt{((\bar{m}+nk)\cdot\gamma^2+1)\cdot(4\beta^2+1)\cdot\alpha^2+\omega(\sqrt{\log m})}} \tag{3}$$

via a similar analysis. For a re-encryption of level κ ciphertext with $2 \leq \kappa \leq L$, the estimate of error terms are all similar. Taking $\kappa = 2$ as an example, we have (the level 3 of a user j)

$$\begin{cases} \boldsymbol{c}_{j,1} \overset{s}{\approx} B_j^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,1} + R_{i,j}^{(1)} \cdot \boldsymbol{e}'''_3 + \boldsymbol{e}'''_1 \bmod q \\ \boldsymbol{c}_{j,2} \overset{s}{\approx} D_j^T \cdot \boldsymbol{s} + \boldsymbol{e}_{i,2} + R_{i,j}^{(2)} \cdot \boldsymbol{e}'''_3 + \boldsymbol{e}'''_2 + \lfloor \frac{q}{2} \rceil \cdot \boldsymbol{\mu} \bmod q \\ \boldsymbol{c}_{j,3} \overset{s}{\approx} A_{j,3}^T \cdot \boldsymbol{s} + R_{i,j}^{(3)} \cdot \boldsymbol{e}'''_3 \bmod q \text{ (if } \kappa \leq L-1) \end{cases}$$

with $\boldsymbol{e}'''_1, \boldsymbol{e}'''_2$ distributed as (2) and $\boldsymbol{e}'''_3$ distributed as (3). In particular, we could bound the error terms of $\boldsymbol{c}_{j,k'}$ as

$$||err^{(k')}||_\infty := ||\boldsymbol{e}_{i,k'} + R_{i,j}^{(k')} \cdot \boldsymbol{e}'''_3 + \boldsymbol{e}'''_{k'}||_\infty \leq \omega(\sqrt{\log m}) \cdot (\bar{m} + nk) \cdot \alpha \cdot \beta \cdot \gamma^2$$

for $k' \in [2]$ and $||err^{(3)}||_\infty := ||R_{i,j}^{(3)} \cdot \boldsymbol{e}_3'''||_\infty \leq \omega(\sqrt{\log m}) \cdot (\bar{m}+nk) \cdot \alpha \cdot \beta \cdot \gamma^2$ with overwhelming probability. Therefore, the error term of $-R_j^T \cdot \boldsymbol{c}_{j,1} + \boldsymbol{c}_{j,2}$ could be bounded as

$$\begin{aligned}&||-R_j^T \cdot (\boldsymbol{e}_{i,1} + R_{i,j}^{(1)} \cdot \boldsymbol{e}_3''' + \boldsymbol{e}_1''') + \boldsymbol{e}_{i,2} + R_{i,j}^{(2)} \cdot \boldsymbol{e}_3''' + \boldsymbol{e}_2'''||_\infty \\ &\leq \omega(\log m) \cdot \sqrt{m} \cdot (\bar{m} + nk) \cdot \alpha \cdot \beta \cdot \gamma^2.\end{aligned}$$

Moreover, a simple induction shows that the error terms of a level κ ciphertext of a user j could be bounded as $||err^{(k')}||_\infty \leq \omega(\log^{\frac{\kappa-2}{2}} m) \cdot (\bar{m}+nk)^{\frac{\kappa-1}{2}} \cdot \alpha \cdot \beta \cdot \gamma^{\kappa-1}$ with $k' \in [3]$. Hence, we could decrypt a level κ ciphertext successfully for $\kappa \geq 3$, if

$$\omega(\log^{\frac{\kappa-1}{2}} m) \cdot \sqrt{m} \cdot (\bar{m} + nk)^{\frac{\kappa-1}{2}} \cdot \alpha \cdot \beta \cdot \gamma^{\kappa-1} \leq \frac{q}{5}.$$

Lemma 10. *For $q \geq 5 \cdot \omega(\log^{\frac{L}{2}} m) \cdot \sqrt{m} \cdot (\bar{m} + nk)^{\frac{L}{2}} \cdot \alpha \cdot \beta \cdot \gamma^L$, our L-hop PRE scheme with $L = O(1)$ satisfies correctness.*

The proof of the HRA security of our multi-hop PRE schemes is similar to those showed in Theorem 2 by noticing that after changing $A_{\bar{i}^*}$ to the form $[\bar{A}|\bar{A} \cdot R_{\bar{i}^*,1} - H_1 \cdot \mathbf{G}|\bar{A} \cdot R_{\bar{i}^*,2} - H^* \cdot \mathbf{G}]$ in corresponding games, the challenger still have the abilities to generate level $\kappa \geq 2$ re-encryption keys from i^* to any other user. We omit its details, and just propose the following theorem.

Theorem 3. *For any $L = O(1)$, our L-hop PRE scheme satisfies* HRA *security under the* $\text{Multi-LWE}^n_{n,q,2m,D_{\mathbb{Z},\alpha}}$ *assumption and the* $\text{LWE}_{n,q,2m+\bar{m},D_{\mathbb{Z},\alpha}}$ *assumption in the standard model.*

The reason why our multi-hop PRE constructions could not be modified to satisfy the key-privacy is that we could not re-randomize $A_{i,l}$ for $l \in \{2, L\}$. Since in equation 1, there is a correspondence of corresponding part (matrix B).

A Supporting Definitions and Discussions

The definitions of the HRA security and the key privacy of an L-hop PRE scheme with $L = \text{poly}(n)$ are proposed as following. The main difference between Definition 5 and Definition 3 is the permission of re-encryption key generation queries from honest users to corrupted users. Notice that, in our definitions, multiple re-encryptions of any pair of users (i, j) could be exposed to the adversary (which is different from that proposed in [10], in which the re-encryption key for each pair of users i, j is unique).

Definition 5. *Consider the following interactions between a* PPT *adversary $\mathfrak{A}$ and a challenger $\mathfrak{C}$:*

- *Phase* 1*:*
  1. *The challenger $\mathfrak{C}$ generates the public parameters pp and sends them to an adversary $\mathfrak{A}$. It also maintains a counter* **numCt** *which is initialized to be* 0*, and two databases* **Deriv** *and $\mathfrak{D}_C$ which are initialized to be empty.*

2. *The adversary $\mathfrak{A}$ first decides the sizes of sets U_H and U_C, and could adaptively query the following two oracles.*
   - *Corrupted key generation oracle, in which $\mathfrak{C}$ would return $(pk, sk) =$ $\mathbf{KeyGen}(pp)$ to $\mathfrak{A}$ and record (pk, sk), together with corresponding user index, to the set U_C.*
   - *Uncorrupted key generation oracle, in which $\mathfrak{C}$ would first generate $(pk, sk) = \mathbf{KeyGen}(pp)$, then return pk to $\mathfrak{A}$ and record (pk, sk), together with corresponding user index, in the set U_H.*

- *Phase* 2*: In this phase, $\mathfrak{A}$ could adaptively query the following oracles. We require that the queried users' indexes must have been appeared in Phase* 1*, and user indexes $i \neq j$ if exist.*
  1. *Encryption oracle: On input an index i and a message m, the challenger $\mathfrak{C}$ computes $C_i = \mathbf{Enc}(pp, pk_i, m)$, increments* **numCt***, adds the item $(\perp, i; \perp, \boldsymbol{ct}_i; \mathbf{numCt})$ to $\mathfrak{D}_C$, and returns $(\perp, i; \perp, \boldsymbol{ct}_i; \mathbf{numCt})$ to $\mathfrak{A}$.*
  2. *Re-encryption key generation oracle: Upon receiving indexes (i, j) and a level $\kappa \in [L]$, $\mathfrak{C}$ will return $\perp$ if $i \in U_H$ and $j \in U_C$. Otherwise, it will return a re-encryption key $rk^{\kappa}_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j, \kappa)$ to $\mathfrak{A}$.*
  3. *Re-encryption oracle: $\mathfrak{A}$ submits query $(i, j, k; \kappa)$ for some $k \leq \mathbf{numCt}$ and $\kappa \in [L]$, the challenger $\mathfrak{C}$ will first retrieve $\mathfrak{D}_C$ to recover an item $(*, i; *, \boldsymbol{ct}_i; k)$, where $*$ could be equal to $\perp$, and could represent any value. If no such item exists, output $\perp$. If $k \in \mathbf{Deriv}$ and $j \in U_C$, output $\perp$. Or else, $\mathfrak{C}$ generates a re-encryption key $rk^{\kappa}_{i \mapsto j} = \mathbf{ReKeyGen}(pp, sk_i, pk_j, \kappa)$ and parses $\boldsymbol{ct}_i = (C_i; \kappa')$. If $\kappa' \neq \kappa$, return $\perp$. Otherwise, compute $C_j = \mathbf{ReEnc}(pp, C_i, rk^{\kappa}_{i \mapsto j})$, increment* **numCt***, record $(i, j; \boldsymbol{ct}_i, \boldsymbol{ct}_j :=$ $(C_j; \kappa + 1); \mathbf{numCt})$ to $\mathfrak{D}_C$, and return $(i, j; \boldsymbol{ct}_i, \boldsymbol{ct}_j; \mathbf{numCt})$ to $\mathfrak{A}$. If $k \in \mathbf{Deriv}$, add* **numCt** *to* **Deriv***.*
  4. *Challenge oracle: This oracle could only be queried once. The adversary $\mathfrak{A}$ submits challenge (i^*, m_0^*, m_1^*) with $i^* \in U_H$, then the challenger $\mathfrak{C}$ chooses a random bit $b \hookleftarrow U(\{0, 1\})$, computes $(C_b^*; 1)$ with $C_b^* =$ $\mathbf{Enc}(pp, pk_{i^*}, m_b^*)$. At the same time, $\mathfrak{C}$ increments* **numCt***, records $(\perp, i^*; \perp, \boldsymbol{ct}^* := (C^*; 1); \mathbf{numCt})$ to $\mathfrak{D}_C$, adds* **numCt** *to* **Deriv***, and returns $(\perp, i^*; \perp, \boldsymbol{ct}^*; \mathbf{numCt})$ to $\mathfrak{A}$.*
- *Phase* 3*: $\mathfrak{A}$ outputs a bit b', and wins if and only if $b' = b$.*

We say an L-hop PRE scheme is HRA *secure, if for any* PPT *adversary $\mathfrak{A}$, its advantage* $\Pr[\mathfrak{A}$ *wins*$] := |\Pr[b' = b] - \frac{1}{2}|$ *in the above game is negligible.*

When $L = O(1)$, the key privacy of an L-hop PRE scheme could be defined similarly as Definition 3 by adding a legitimacy detection phase. While for $L =$ poly(λ), it is even more complicated (than the HRA case) to record all the possible "trivial" wins of an adversary $\mathfrak{A}$ if we add no constraint to the re-encryption key generation oracle (or the re-encryption oracle). A possible way to define a weaker key privacy is following the route of Definition 5 by forbidding the re-encryption key generation queries from U_H to U_C. Another possible way is to ask the adversary commits a tree structure before the security game begins [12]. We take $L = O(1)$ as an example, and propose the following definition. Notice that, our re-encryption keys are also level-based, and the following definition only considers the indistinguishability of re-encryption keys of the level 1. For other level re-encryption keys, similar definitions could be proposed easily.

Definition 6. *Consider the following interactions between a* PPT *adversary* $\mathfrak{A}$ *and a challenger* $\mathfrak{C}$*, in which Phase* 1 *is the same as that in* Definition 3.

- *Phase* 2*: In this phase,* $\mathfrak{A}$ *could adaptively query the following oracles. We require that the queried users' indexes must have been appeared in Phase* 1*, and user indexes* $i \neq j$ *if exist.*
  1. *Encryption oracle, re-encryption oracle and re-encryption key generation oracle are the same as those in Definition 3.*
  2. *Challenge oracle: This oracle could only be challenged once. On input* (i^*, j^*) *with* $j^* \in U_H$*,* $\mathfrak{C}$ *creates* $rk_{i^* \mapsto j^*} = \mathbf{ReKeyGen}(pp, sk_{i^*}, pk_{j^*})$ *until the triplet* $(i^*, j^*, rk_{i^* \mapsto j^*}) \notin \mathfrak{D}_K$*. Then, he records* $(i^*, j^*, rk_{i^* \mapsto j^*})$ *to* $\mathfrak{D}_K$*, samples a random* $b \hookleftarrow U(\{0,1\})$*, and returns* $rk_{i^* \mapsto j^*}$ *to* $\mathfrak{A}$ *if* $b = 1$*, or returns a random key* rk^* *(which may obey to some distributions with large enough entropy) in the key space to* $\mathfrak{A}$ *if* $b = 0$*.*
- *Phase* 3*: Legitimacy detection.* $\mathfrak{C}$ *checks whether there is a re-encryption chain of any ciphertext* C_{i^*} *of user* i^* *(which consists of re-encryptions obtained from re-encryption oracle, and re-encryptions which could be generated by* $\mathfrak{A}$ *itself via re-encryption keys obtained by querying re-encryption key generation oracle) with length at most* $L-1$ *from* i^* *to some corrupted user* $j \in U_C$ *by using* $\mathfrak{D}_k$ *and* $\mathfrak{D}_C$*. If so, output* $\perp$*.*
- *Phase* 4*: If no* $\perp$ *outputted before,* $\mathfrak{A}$ *outputs a bit* b'*, and wins if and only if* $b' = b$*.*

We say an L-hop PRE scheme is key-private, if for any PPT *adversary* $\mathfrak{A}$*, its advantage* $\Pr[\mathfrak{A}\ wins] := |\Pr[b' = b] - \frac{1}{2}|$ *in the above game is negligible.*

References

1. Ahene, E., Qin, Z., Adusei, A.K., Li, F.: Efficient signcryption with proxy re-encryption and its application in smart grid. IEEE Internet of Things Journal **6**(6), 9722–9737 (2019)
2. Al-Asli, M., Elrabaa, M.E.S., Abu-Amara, M.: Fpga-based symmetric re-encryption scheme to secure data processing for cloud-integrated internet of things. IEEE Internet of Things Journal **6**(1), 446–457 (2019)
3. Aono, Y., Boyen, X., Phong, L.T., Wang, L.: Key-private proxy re-encryption under lwe. In: Paul, G., Vaudenay, S. (eds.) Progress in Cryptology – INDOCRYPT 2013. pp. 1–18. Springer International Publishing, Cham (2013)
4. Applebaum, B., Cash, D., Peikert, C., Sahai, A.: Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In: Halevi, S. (ed.) Advances in Cryptology - CRYPTO 2009. pp. 595–618. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)
5. Ateniese, G., Benson, K., Hohenberger, S.: Key-private proxy re-encryption. In: Fischlin, M. (ed.) Topics in Cryptology – CT-RSA 2009. pp. 279–294. Springer Berlin Heidelberg, Berlin, Heidelberg (2009)

6. Ateniese, G., Fu, K., Green, M., Hohenberger, S.: Improved proxy re-encryption schemes with applications to secure distributed storage. ACM Trans. Inf. Syst. Secur. **9**(1), 1–30 (Feb 2006)
7. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) Advances in Cryptology — EUROCRYPT'98. pp. 127–144. Springer Berlin Heidelberg, Berlin, Heidelberg (1998)
8. Canetti, R., Hohenberger, S.: Chosen-ciphertext secure proxy re-encryption. In: Proceedings of the 14th ACM Conference on Computer and Communications Security. p. 185–194. CCS '07, Association for Computing Machinery, New York, NY, USA (2007)
9. 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.) Public-Key Cryptography – PKC 2014. pp. 95–112. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
10. Cohen, A.: What about bob? the inadequacy of cpa security for proxy reencryption. In: Lin, D., Sako, K. (eds.) Public-Key Cryptography – PKC 2019. pp. 287–316. Springer International Publishing, Cham (2019)
11. Everspaugh, A., Paterson, K., Ristenpart, T., Scott, S.: Key rotation for authenticated encryption. In: Katz, J., Shacham, H. (eds.) Advances in Cryptology – CRYPTO 2017. pp. 98–129. Springer International Publishing, Cham (2017)
12. Fan, X., Liu, F.H.: Proxy re-encryption and re-signatures from lattices. In: Deng, R.H., Gauthier-Umaña, V., Ochoa, M., Yung, M. (eds.) Applied Cryptography and Network Security. pp. 363–382. Springer International Publishing, Cham (2019)
13. Fuchsbauer, G., Kamath, C., Klein, K., Pietrzak, K.: Adaptively secure proxy re-encryption. In: Lin, D., Sako, K. (eds.) Public-Key Cryptography – PKC 2019. pp. 317–346. Springer International Publishing, Cham (2019)
14. Gentry, C., Peikert, C., Vaikuntanathan, V.: Trapdoors for hard lattices and new cryptographic constructions. In: Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing. pp. 197–206. STOC '08, ACM, New York, NY, USA (2008)
15. Ivan, A., Dodis, Y.: Proxy cryptography revisited. In: in Proceedings of the Network and Distributed System Security Symposium (NDSS) (2003)
16. Katsumata, S., Yamada, S.: Partitioning via non-linear polynomial functions: More compact ibes from ideal lattices and bilinear maps. In: Cheon, J.H., Takagi, T. (eds.) Advances in Cryptology – ASIACRYPT 2016. pp. 682–712. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
17. Kirshanova, E.: Proxy re-encryption from lattices. In: Krawczyk, H. (ed.) Public-Key Cryptography – PKC 2014. pp. 77–94. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)
18. Liang, K., Au, M.H., Liu, J.K., Susilo, W., Wong, D.S., Yang, G., Phuong, T.V.X., Xie, Q.: A dfa-based functional proxy re-encryption scheme for secure public cloud data sharing. IEEE Transactions on Information Forensics and Security **9**(10), 1667–1680 (2014)
19. Liang, X., Weng, J., Yang, A., Yao, L., Jiang, Z., Wu, Z.: Attribute-based conditional proxy re-encryption in the standard model under lwe. Cryptology ePrint Archive, Report 2021/613 (2021), https://eprint.iacr.org/2021/613
20. Mambo, M., Okamoto, E.: Proxy cryptosystems: Delegation of power to decrypt ciphertexts. IEICE Trans. Fundament Electron Commun. Sci. **80**(1), 54–63 (1997)

21. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) Advances in Cryptology – EUROCRYPT 2012. pp. 700–718. Springer Berlin Heidelberg, Berlin, Heidelberg (2012)
22. Micciancio, D., Regev, O.: Worst-case to average-case reductions based on gaussian measures. In: Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science. p. 372–381. FOCS '04, IEEE Computer Society, USA (2004)
23. Pareek, G., Purushothama, B.R.: Proxy re-encryption for fine-grained access control: Its applicability, security under stronger notions and performance. Journal of Information Security and Applications **54**, 102543 (2020)
24. Peikert, C.: An efficient and parallel gaussian sampler for lattices. In: Rabin, T. (ed.) Advances in Cryptology – CRYPTO 2010. pp. 80–97. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
25. Phong, L.T., Wang, L., Aono, Y., Nguyen, M.H., Boyen, X.: Proxy re-encryption schemes with key privacy from lwe. Cryptology ePrint Archive, Report 2016/327 (2016), https://eprint.iacr.org/2016/327
26. Polyakov, Y., Rohloff, K., Sahu, G., Vaikuntanathan, V.: Fast proxy re-encryption for publish/subscribe systems. ACM Trans. Priv. Secur. **20**(4) (Sep 2017)
27. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing. p. 84–93. STOC '05, Association for Computing Machinery, New York, NY, USA (2005)
28. Singh, K., Rangan, C.P., Agrawal, R., Sheshank, S.: Provably secure lattice based identity based unidirectional pre and pre+ schemes. Journal of Information Security and Applications **54**, 102569 (2020)
29. Xu, P., Jiao, T., Wu, Q., Wang, W., Jin, H.: Conditional identity-based broadcast proxy re-encryption and its application to cloud email. IEEE Transactions on Computers **65**(1), 66–79 (2016)