Unbounded Multi-Hop Proxy Re-Encryption with HRA Security: An LWE-Based Optimization

Xiaohan Wan[1], Yang Wang[1], Haiyang Xue[2], and Mingqiang Wang[1]

[1] School of Mathematics, Shandong University
xhwan@mail.sdu.edu.cn, {wyang1114, wangmingqiang}@sdu.edu.cn
[2] Singapore Management University
haiyangxc@gmail.com

Abstract. Proxy re-encryption (PRE) schemes enable a semi-honest proxy to transform a ciphertext of one user i to another user j while preserving the privacy of the underlying message. Multi-hop PRE schemes allow a legal ciphertext to undergo multiple transformations, but for lattice-based multi-hop PREs, the number of transformations is typically bounded due to the increase of error terms. Recently, Zhao *et al.* (Esorics 2024) introduced a lattice-based unbounded multi-hop (homomorphic) PRE scheme that supports an unbounded number of hops. Nevertheless, their scheme only achieves the selective CPA security. In contrast, Fuchsbauer *et al.* (PKC 2019) proposed a generic framework for constructing HRA-secure unbounded multi-hop PRE schemes from FHE. Despite this, when instantiated with state-of-the-art FHEW-like schemes, the overall key size and efficiency remain unsatisfactory.
In this paper, we present a lattice-based unbounded multi-hop PRE scheme with the stronger adaptive HRA security (i.e. security against honest re-encryption attacks), which is more suitable for practical applications. Our scheme features an optimized re-encryption process based on the FHEW-like blind rotation, which resolves the incompatibility between the noise flooding technique and Fuchsbauer *et al.* 's framework when instantiated with FHEW-like schemes. This results in reduced storage requirements for public keys and offers higher efficiency. Moreover, our optimized unbounded multi-hop PRE scheme can be modified to an unbounded homomorphic PRE, a scheme allowing for arbitrary homomorphic computations over fresh, re-encrypted, and evaluated ciphertexts.

Keywords: Proxy re-encryption · Unbounded multi-hop · HRA security · LWE

1 Introduction

Proxy re-encryption (PRE), introduced by Blaze *et al.* [5], is a cryptographic primitive that allows a semi-honest proxy to transform a ciphertext encrypted under user i's key into a ciphertext of the same message encrypted under user j's key without revealing the original message. Suppose $(pk^{(i)}, sk^{(i)})$ and $(pk^{(j)}, sk^{(j)})$

are the key pairs of users i and j, respectively. With $pk^{(j)}$ and the secret key $sk^{(i)}$, user i can generate and provide a re-encryption key $rk_{i\to j}$ to a semi-honest proxy. This enables the proxy to convert the ciphertext $ct^{(i)}$ (encrypted for user i) into a ciphertext decryptable by user j, denoted as $ct^{(j)}$ (the re-encrypted ciphertext).

Thanks to this functionality, PRE has found a variety of applications, such as digital rights management (DRM) [17], encrypted email forwarding [5], distributed system [3], etc. Numerous PRE schemes have been proposed in the literature, with the majority relying on classical cryptographic assumptions [16, 22]. Among these, there are also quantum-resistant schemes, particularly those based on lattices [6, 20]. In this paper, we mainly focus on *lattice-based PRE* schemes.

CPA Security and HRA Security. As standard public-key encryption (PKE) schemes, security against chosen-plaintext attacks (CPA) is a well-established security notion for PREs. However, CPA security is not adequate in real-world application scenarios. In the CPA model, re-encryption queries (which allow the adversary to obtain re-encryptions of non-challenge ciphertexts) and re-encryption key queries (which return re-encryption keys while preventing trivial attacks) between an honest user and a corrupted user are prohibited. In practice, adversaries may gain access to re-encryptions from honest users to corrupted users. Cohen [9] demonstrated that this capability could lead to practical attacks on the lattice-based scheme proposed in [25]. To address this, Cohen introduced a stronger security model known as security against honest re-encryption attacks (HRA) [9], which accounts for this adversarial capability and is more applicable to real-world use-cases of related PREs.

Single-hop and (Un)bounded Multi-hop. According to the re-encryption times of ciphertexts, PRE can be subdivided into single-hop PRE, where ciphertexts can be re-encrypted only once, and multi-hop PRE (mPRE), where the ciphertexts can undergo multiple re-encryptions (e.g., from user i to user j and then to user k, and so on). Compared to single-hop PRE, multi-hop PRE schemes are more practical and desirable, as they offer greater flexibility and convenience for providing re-encryption services.

For multi-hop PRE schemes based on classical cryptographic assumptions, such as DDH [22], the number of re-encryptions is typically unbounded. However, in the lattice setting, each re-encryption introduces additional noise into the ciphertext, and this cumulative noise increases linearly with the number of re-encryptions. Thus, to ensure that the final re-encrypted ciphertext remains decryptable, the number of re-encryptions must be bounded by a predefined threshold, restricting the flexibility of PRE.

Recently, Zhao *et al.* [28] introduced a lattice-based unbounded multi-hop homomorphic PRE (HPRE) scheme by emerging the FHEW bootstrapping procedure [10] into the re-encryption process, thereby supporting an unbounded number of re-encryptions. However, this approach suffers from redundancy in both storage and computation. Additionally, their scheme only achieves selective CPA security, where the target user is designated at the start of the game,

and the adversary is prohibited from learning re-encryptions between honest users and corrupted users, which is impractical in real-world scenarios.

We notice that Fuchsbauer *et al.* [13] introduced a generic framework for constructing unbounded mPRE schemes from fully homomorphic encryption (FHE). In their CPA-secure construction, the encryption algorithm is simply the corresponding FHE encryption, while the re-encryption can be regard as homomorphically evaluating the decryption circuit of the underlying FHE scheme. To achieve adaptive HRA security, a sanitization algorithm [11] is applied to both the encryption and re-encryption processes, ensuring the statistical indistinguishability between fresh ciphertexts and re-encrypted ciphertexts.

One of the main approaches for ciphertext sanitization is Ducas-Stehlé washing machine [11], which involves a sequence of iterations, each including a re-randomization procedure followed by the invocation of a bootstrapping algorithm. As mentioned in [11], at least 8 iterations are required to sanitize an FHEW ciphertext with the original FHEW parameters [10], which may not be considered secure, and achieving provable security necessitates additional iterations. When applied to Fuchsbauer's framework, this means that the corresponding PRE scheme must invoke multiple bootstrapping operations for both encryption and re-encryption, leading to significant costs in terms of both storage and computation. An alternative approach, noise flooding [2], can also be used for ciphertexts sanitization. However, directly applying it to Fuchsbauer's framework with the state-of-the-art FHEW-like schemes [10, 18] leads to incompatibility (see Section 1.2 for a detailed discussion).

Therefore, this paper seeks to advance both security and efficiency in this direction by proposing efficient lattice-based unbounded multi-hop PRE schemes that achieve adaptive HRA security.

1.1 Our Contributions

We primarily focuses on investigating how to design a lattice-based HRA-secure unbounded multi-hop PRE scheme, and aims to reduce the storage and computational overhead of both encryption and re-encryption algorithms. The contributions of this paper are summarized as follows:

- We propose a lattice-based unbounded multi-hop PRE scheme with HRA security in the standard model, supporting an unbounded number of re-encryptions. More precisely, our scheme achieves HRA security in an adaptive manner, which allows the adversary to adaptively designate the target user and make all oracle queries in the case of HRA.
- We introduce a novel re-encryption method that primarily utilizes an FHEW-like blind rotation algorithm, addressing the incompatibility of the noise flooding technique with Fuchsbauer *et al.* 's framework. Although the use of noise flooding technique results in a super-polynomial modulus, rather than the polynomial modulus in [13, 28], our method still outperforms them in terms of public key size and computational efficiency. For a detailed analysis and comparison, please refer to Section 1.2 and Table 1.

– Additionally, our multi-hop PRE scheme can be extended to a homomorphic PRE scheme, which enables arbitrary homomorphic computations over fresh, re-encrypted, and evaluated ciphertexts. Refer to Section 4 for details.

We refer to Table 2 for a comparison of our scheme with other existing multi-hop PRE schemes.

Schemes	**Security**	**PK**	**KSK**	**RK**	**Mult**
FKKP19 [13]	**HRA**	$O(n^3 \log^4 n)$	$O(n^{2.5} \log^3 n)$	$O(n^{2.5} \log^3 n)$	$(3n + \frac{N-n}{w})(\kappa + 1)$
ZWW24 [28]	**CPA**	$O(n^{3.5} \log^5 n)$	$O(n^{3.5} \log^4 n)$	$O(n^2 \log^2 n)$	$2d_r(1 - \frac{1}{B_r})n$
Ours	**HRA**	$O(n^{2.75} \log^3 n)$	–	$O(n^3 \log n)$	$3n + \frac{N-n}{w}$

Table 1. Key size and computation complexity of [13,28] and ours. The columns "**PK**", "**KSK**" and "**RK**" represent the bit size of public keys, key switching keys and re-encryption keys, respectively. The column "**Mult**" represents the number of the scalar multiplications between a polynomial and a vector consisting of RLWE ciphertexts in one re-encryption. The parameter n is the lattice dimension and also serves as security parameter, N is the ring dimension, w is the window size satisfying $2N/n - 2 < w < n$, κ is the number of iterations in $\mathsf{Sanitize}$ algorithm with a minimum value of 8, and B_r is the FHEW bootstrapping base, which breaks integers modulus q into d_r digits.

Scheme	**Assumption**	**Security**	**Ubounded?**	**Adaptive?**	**Homomorphic?**
LHAM20 [16]	iO	CCA	✓	×	×
MPW23 [22]	DDH	HRA	✓	×	×
CCLN+14 [6]	LWE	HRA	×	✓	×
LMW17 [20]	LWE	CPA	×	✓	✓
ZLH24 [29]	LWE	HRA	×	✓	×
FKKP19 [13]	–	HRA	✓	✓	✓
ZWW24 [28]	$(\mathsf{R})\mathsf{LWE}^*$	CPA	✓	×	✓
ours	$(\mathsf{R})\mathsf{LWE}^*$	HRA	✓	✓	✓

Table 2. Comparison with multi-hop PRE schemes. The "✓" indicates that the scheme satisfies the corresponding property, while "×" indicates that the scheme does not satisfy it. The column "**Adaptive?**" asks whether the security models allow the adversary to designate the target user and make all oracle queries adaptively. "–" denotes that the assumption is determined by the underlying FHE scheme. "$(\mathsf{R})\mathsf{LWE}^*$" means that the corresponding PRE schemes require an additional circular security assumption.

1.2 Technical Overview

We provide a high-level overview of the construction of our unbounded multi-hop PRE scheme with HRA security.

Fuchsbauer et al.'s Framework. We provide a brief description of Fuchsbauer *et al.*'s generic framework, instantiated with the FHEW-like scheme in [18], which we denote as GPRE. The public key version of the FHEW-like encryption for $m \in \{0,1\}$ under user i's public key $(\mathbf{A}_i, \mathbf{b}_i)$ is of the form:

$$ct_i = (\mathbf{A}_i\mathbf{r}_i + \mathbf{e}_{i,1}, \mathbf{b}_i^\top \mathbf{r}_i + e_{i,2} + \lfloor \frac{q}{4} \rceil \cdot m),$$

where $\mathbf{r}_i$, $\mathbf{e}_{i,1}$ and $e_{i,2}$ are small randomness. Assume $\mathbf{b}_i = -\mathbf{A}_i^\top \mathbf{s}_i + \mathbf{e}_i$ with $\mathbf{s}_i$ the corresponding secret key and q is the ciphertext modulus, we denote the encryption as P-LWE$_{q,\mathbf{s}_i}(m)$ and its symmetric counterpart as LWE$_{q,\mathbf{s}_i}(m)$. The encryption in GPRE is represented as $ct^{(i)} \coloneqq \mathsf{Sanitize}(ct_i)$, where $\mathsf{Sanitize}$ denotes the sanitization algorithm [11], which involves a sequence of iterations, each consisting of a re-randomization procedure and a bootstrapping algorithm. The re-encryption algorithm of GPRE first homomorphically evaluates the FHEW decryption circuit to transform i's ciphertext $ct^{(i)}$ into j's ciphertext with a lower noise, followed by the $\mathsf{Sanitize}$ algorithm. The whole re-encryption algorithm of GPRE is shown in Fig. 1.

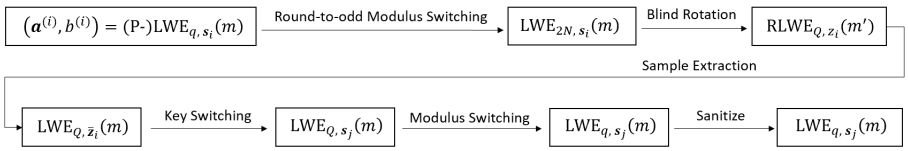

Fig. 1. The re-encryption procedure of GPRE [13]

Note that the input to the $\mathsf{Sanitize}$ algorithm in the encryption process is a public key encryption, whereas in the re-encryption process, as can be seen from FHEW bootstrapping, the input to $\mathsf{Sanitize}$ algorithm is a symmetric LWE ciphertext of the form: $(\mathbf{a}, -\langle \mathbf{a}, \mathbf{s}_j \rangle + \lfloor \frac{q}{4} \rceil \cdot m + e)$, where $\mathbf{a}$ and e are randomness. Therefore, it's not possible to directly use the noise flooding technique as the $\mathsf{Sanitize}$ algorithm to achieve statistical indistinguishability between fresh ciphertexts and re-encrypted ciphertexts, as required for HRA security.

Modified Re-Encryption. The key to addressing the above issue is that the input ciphertext before noise flooding in the re-encryption process should be encrypted with the public key, as in encryption. We begin with an original re-encryption algorithm based on the key switching technique, commonly used in PRE, which transforms the i's ciphertext $ct^{(i)} = (\mathbf{a}^{(i)}, b^{(i)})$ under the secret key

$\mathbf{s}_i$ into j's ciphertext $ct^{(j)}$ under $\mathbf{s}_j$:

$$\begin{aligned} ct^{(j)} =& rk_{i\to j} \cdot \begin{bmatrix} \mathsf{BitDecomp}(\mathbf{a}^{(i)}) \\ b^{(i)} \end{bmatrix} \\ =& \begin{bmatrix} \mathbf{A}_j\mathbf{R}_{i,j} + \mathbf{E}'_{i,j} & \mathbf{0}_{n\times 1} \\ \mathbf{b}_j^\top \mathbf{R}_{i,j} + \mathbf{e}'^\top_{i,j} + \mathsf{Powersof2}(\mathbf{s}_i)^\top & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathsf{BitDecomp}(\mathbf{a}^{(i)}) \\ b^{(i)} \end{bmatrix}, \end{aligned}$$

where $\mathbf{R}_{i,j}, \mathbf{e}'_{i,j}$ and $\mathbf{E}'_{i,j}$ are error vectors/matrices. Regardless of whether the input ciphertext is in the public key or symmetric form, it is clear that the output of the original re-encryption is always in the public key form. Therefore, our **key observation** is that we can reverse the order of key switching and modulus switching in Fig. 1, and then replace key switching with the original re-encryption process. In this way, we can apply the noise flooding technique to sanitize ciphertexts, both in encryption and re-encryption. The re-encryption algorithm of ours is illustrated in Fig. 2.

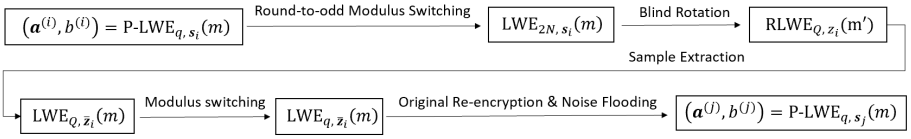

Fig. 2. The re-encryption procedure of our scheme

Our modified re-encryption algorithm **outperforms** [13] and [28] in terms of both public key size and computation complexity. Specifically, the re-encryption process in [28] (cf. Fig. 3) consists of two parts: original re-encryption and FHEW bootstrapping [10]. The re-encryption in [13], which instantiates $\mathsf{Sanitize}$ with Ducas-Stehlé washing method, requires multiple bootstrapping (see the dashed box in Fig. 3) operations for both encryption and re-encryption. This implies that, in addition to blind rotation keys required for blind rotation, both public keys must include additional key switching keys. Furthermore, the public key in [13] also includes large re-randomization keys to sanitize ciphertexts. In contrast, our method eliminates the need for additional keys, significantly reducing the size of public key. Due to the use of noise flooding, a super-polynomial modulus is employed, resulting in a slightly larger re-encryption key in our scheme, but its size remains smaller than the public key in [13,28], please refer to Section 3.4 for a detailed analysis. Notably, the use of only one LMKC+ blind rotation [18] in our re-encryption achieves significantly better efficiency compared to the methods in [13,28].

Adaptive HRA Security. The core idea of our proof follows the FKKP framework [13], which demonstrates that a multi-hop PRE scheme satisfying indistinguishability (IND), weak key-privacy (wKP) and source-hiding (SH) is also adaptive HRA-secure. The IND security, which requires the indistinguishability of ciphertexts for adversary who has no access to re-encryption keys or secret keys, follows from the semantic security of the underlying FHEW scheme. We

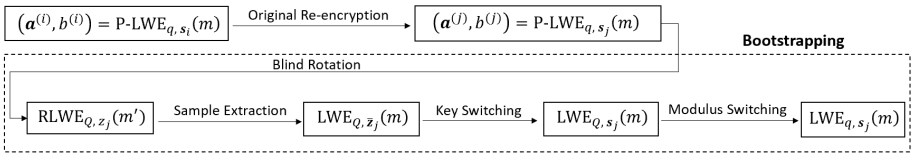

Fig. 3. The re-encryption procedure of [28]

then show that the honestly generated re-encryption key $rk_{i\to j}$ in our scheme can be indistinguishably changed into a simulated one from the view of an adversary who has no knowledge of $sk^{(i)}$, thereby proving wKP security. Finally, the use of the noise flooding technique ensures statistical indistinguishability between fresh ciphertexts and re-encrypted ciphertexts, thereby establishing SH security.

1.3 Related Works

Blind Rotation. Blind rotation is a key technique for constructing FHEW [10] and TFHE [7, 8] bootstrapping algorithms. The AP blind rotation method, introduced by Alperin-Sheriff and Peikert [1], supports arbitrary secret key distribution but requires a large evaluation key size. The GINX blind rotation method [14] features a significantly smaller evaluation key size but is practical only for binary and ternary secrets. Recently, Lee *et al.* [18] proposed an automorphism-based blind rotation method that supports arbitrary secret key distributions, while using small evaluation keys.

Multi-hop PRE. Chandran *et al.* [6] introduced the first multi-hop unidirectional PRE scheme from program obfuscation, achieving selective obfuscation-based security. Subsequently, [27] presented a multi-hop PRE scheme with selective CPA security. Later, Cohen [9] demonstrated that CPA security is inadequate for PRE and proposed a stronger security definition known as HRA security. Fuchsbauer *et al.* [13] presented a generic framework to construct HRA-secure multi-hop PRE schemes from FHE. More recently, Miao *et al.* [22] proposed a multi-hop HRA-secure PRE scheme under the DDH assumption that supports an unbounded number of re-encryptions, but at the cost that the ciphertext length grows linearly with the number of re-encryptions. Zhou *et al.* [29] introduced the first lattice-based multi-hop fine-grained PRE scheme, offering fine-grained re-encryption capabilities with HRA security.

HPRE. Homomorphic PRE is an extension of PRE that enables homomorphic evaluation over ciphertexts under the same public key. Ma *et al.* [21] introduced a unidirectional single-hop HPRE scheme with CPA security based on the LWE assumption. Then, Li *et al.* [19] proposed the first lattice-based multi-hop HPRE scheme with strong anti-collusion property. Note that all these HPRE schemes support a bounded number of re-encryptions. More recently, Zhao *et al.* [28] proposed a lattice-based unbounded multi-hop FPRE scheme that supports an

unbounded number of re-encryptions. However, their scheme only achieves selective CPA security.

2 Preliminaries

Notations. Throughout this paper, we set N to be a positive integer of a power of 2, and set $\mathcal{R} := \mathbb{Z}[X]/(X^N + 1)$. For a positive integer q, denote $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$ and $\mathcal{R}_q := \mathcal{R}/q\mathcal{R}$. We denote column vectors by bold lower-case letters (e.g. $\mathbf{a}$) and matrices by bold capital letters (e.g. $\mathbf{A}$). For $i, j \in \mathbb{N}$ with $i < j$, define $[i, j] \coloneqq \{i, i+1, \cdots, j\}$ and $[i] \coloneqq \{1, 2, \cdots, i\}$. Symbol $x \xleftarrow{\$} \mathcal{D}$ denotes that a random variable x is sampled from a distribution $\mathcal{D}$, and whenever $\mathcal{D}$ is a finite set, it indicates x is sampled from a uniform distribution over $\mathcal{D}$. Let $\lfloor \cdot \rfloor$ ($\lceil \cdot \rceil$) be the floor (ceiling) function, and define $\lfloor \cdot \rceil := \lfloor \cdot + \frac{1}{2} \rfloor$. We use $\mathsf{negl}(n)$ to represent a negligible function (w.r.t n). For two sets $\mathcal{X}$ and $\mathcal{Y}$, let $\mathcal{X} \triangle \mathcal{Y}$ be their symmetric difference. We only consider directed acyclic graphs $G = (\mathcal{V}, \mathcal{E})$ with vertices $\mathcal{V}$ and edges $\mathcal{E} \subseteq \mathcal{V}^2$ in this paper, and say a vertex j is reachable from another vertex i if there is a directed path from i to j in G. Symbol $\mathcal{G}(\mathfrak{n}, \delta, d)$ represents the set consisting of all directed acyclic graphs with $\mathfrak{n}$ vertices, outdegree δ and depth d. We use $\mathsf{children}(i, G)$ to denote the set of vertices j such that $(i, j) \in \mathcal{E}$.

2.1 Lattices and Gaussian Distributions

For $s > 0$ and $\boldsymbol{c} \in \mathbb{R}^n$, the Gaussian function $\rho_{s,\mathbf{c}} : \mathbb{R}^n \to \mathbb{R}$ is defined by $\rho_{s,\mathbf{c}}(\mathbf{x}) := e^{-\pi \|\mathbf{x}-\mathbf{c}\|^2/s^2}$. A discrete Gaussian distribution over $\mathbb{Z}^n$ is $\mathcal{D}_{\mathbb{Z}^n,s,\mathbf{c}}(\boldsymbol{x}) := \rho_{s,\mathbf{c}}(\boldsymbol{x}) / \sum_{\mathbf{y}\in\mathbb{Z}^n} \rho_{s,\mathbf{c}}(\mathbf{y})$, with parameters omitted when $s = 1$ and $\mathbf{c} = 0$. Notice that $\mathcal{D}_{\mathbb{Z}^n,s} = (\mathcal{D}_{\mathbb{Z},s})^n$. We say a distribution $\mathcal{D}$ (over $\mathbb{Z}^n$) is B-bounded if $\Pr[\boldsymbol{x} \xleftarrow{\$} \mathcal{D} : \|\boldsymbol{x}\| \geq B] \leq \mathsf{negl}(n)$. It is known that $\mathcal{D}_{\mathbb{Z}^n,s}$ is $\sqrt{n} \cdot s$-bounded, and $\mathcal{D}_{\mathbb{Z},s}$ is $\log n \cdot s$-bounded [23]. We also need the following lemma.

Lemma 1 (Smudging Lemma [2]). *Let $B_1, B_2 \in \mathbb{N}$, for any $e_1 \in [-B_1, B_1]$, let E_1 and E_2 be independent random variables uniformly distributed on $[-B_2, B_2]$ and define the two stochastic variables $X_1 = E_1 + e_1$ and $X_2 = E_2$. Then the statistical distance $\Delta(X_1, X_2) < \frac{B_1}{B_2}$.*

Ever since introduced by Regev [26], the LWE problem (and its variants Ring/Module LWE problem) has become a fundamental problem in lattice-based cryptography. For $n, m, q \in \mathbb{N}$, and a noise distribution χ over $\mathbb{Z}$, the (normal form, decision) LWE problem, denoted by $\mathsf{LWE}^m_{n,q,\chi}$, is to distinguish $(\mathbf{A}, \mathbf{As}+\mathbf{e})$ from $(\mathbf{A}, \mathbf{u})$, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{s} \xleftarrow{\$} \chi^n$, $\mathbf{e} \xleftarrow{\$} \chi^m$ and $\mathbf{u} \xleftarrow{\$} \mathbb{Z}_q^m$. For any PPT adversary $\mathcal{A}$, its advantage of attacking $\mathsf{LWE}^m_{n,q,\chi}$ is defined as

$$\mathsf{Adv}_{\mathcal{A}}(\mathsf{LWE}^m_{n,q,\chi}) := |\Pr[\mathcal{A}(\mathbf{A}, \mathbf{As} + \mathbf{e}) = 1] - \Pr[\mathcal{A}(\mathbf{A}, \mathbf{u}) = 1]|.$$

For suitable parameters, there are (quantum) reductions from worst-case lattice problems (e.g. SIVP_γ) to corresponding LWE problems [26]. Also, by using a

standard hybrid argument, it is possible to show that decision LWE problems with multiple secrets, denoted by k-$\mathsf{LWE}^m_{n,q,\chi}$, are also hard. Namely, it is hard to distinguish $(\mathbf{A}, \mathbf{A}\cdot\mathbf{S}+\mathbf{E})$ from $(\mathbf{A}, \mathbf{U})$, where $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{m\times n}$, $\mathbf{S} \xleftarrow{\$} \chi^{n\times k}$, $\mathbf{E} \xleftarrow{\$} \chi^{m\times k}$ and $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{m\times k}$. Similar results also hold for RLWE problems over $\mathcal{R}$ for our purpose (and we use the so-called coefficient embedding).

2.2 Useful Algorithms in FHEs

Let's briefly review several algorithms we require. For more details, please refer to App. A.

Let $l = \lceil \log q \rceil$. For any $x \in \mathbb{Z}_q$, we define $\mathsf{BitDecomp}(x) := (x_0, \cdots, x_{l-1})^\top \in \{0,1\}^l$ with $\sum_{i=0}^{l-1} x_i \cdot 2^i$. Also, we define $\mathsf{Powersof2}(y) := (y, 2\cdot y, \cdots, 2^{l-1}\cdot y)^\top \in \mathbb{Z}_q^l$. It is easy to verify that $\mathsf{BitDecomp}(x)^\top \cdot \mathsf{Powersof2}(y) = x\cdot y \pmod q$. Similarly, for any vectors $\mathbf{x}, \mathbf{y} \in \mathbb{Z}_q^n$, $\mathsf{BitDecomp}(\mathbf{x})^\top \cdot \mathsf{Powersof2}(\mathbf{y}) = \langle \mathbf{x}, \mathbf{y}\rangle \pmod q$. We will use the following useful algorithms:

- $\mathsf{SampleExtract}(c)$: On input an RLWE ciphertext $c = (a, b = -a\cdot s + m + e) \in \mathcal{R}_q^2$, output an LWE ciphertext of the form $(\boldsymbol{a}, b = -\langle \boldsymbol{a}, \boldsymbol{s}\rangle + m_0 + e_0) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$, where $\boldsymbol{a}$ is related to a, $\boldsymbol{s}$ is the coefficient vector of s, m_0 and e_0 are the constant terms of polynomials m and e, respectively.
- $\mathsf{ModSwitch}_{Q,q}(ct)$: On input an LWE ciphertext $ct = (\mathbf{a}, b) \in \mathbb{Z}_Q^N \times \mathbb{Z}_Q$ under a modulus Q, it outputs an LWE ciphertext $ct' = (\lfloor \frac{q}{Q}\cdot \mathbf{a}\rceil, \lfloor \frac{q}{Q}\cdot b\rceil) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$ under a modulus q. Similarly, $\mathsf{ModSwitch}^{\mathsf{odd}}_{Q,q}(ct)$ outputs $ct' = (\lfloor \frac{q}{Q}\cdot \mathbf{a}\rceil_{\mathsf{odd}}, \lfloor \frac{q}{Q}\cdot b\rceil_{\mathsf{odd}}) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$, where $\lfloor x\rceil_{\mathsf{odd}}$ is the nearest odd integer to x.
- $\mathsf{KeySwitch}(\mathbf{ksk}_{\mathbf{s}\to\mathbf{t}}, ct)$: On input an LWE ciphertext $ct = (\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ of a message m under the secret key $\mathbf{s} \in \mathbb{Z}_q^n$ and the key switching key $\mathbf{ksk}_{\mathbf{s}\to\mathbf{t}} = \{\mathrm{LWE}_{q,\mathbf{t}}(kB_{ks}^i s_j)\}_{k\in[0,B_{ks}-1], i\in[0,d_{ks}-1], j\in[n]}$ where $d_{ks} = \lceil \log_{B_{ks}} q\rceil$, it computes $a_{j,i}$ such that $a_j = \sum_i a_{j,i} B_{ks}^i$ for all i, j and outputs $ct' = (\mathbf{0}, b) + \sum_{i,j} \mathrm{LWE}_{q,\mathbf{t}}(a_{j,i} B_{ks}^i s_j) \pmod q$.

The blind rotation algorithm we use to control error expansion is the LMKC+ blind rotation algorithm [18]. We summarize related results in Lemma 2, and the detailed LMKC+ algorithm $\mathsf{BlindRotate}$ is described in App. A.2.

Lemma 2 (LMKC+ Blind Rotation [18]). *Let n, q, Q, B_g, w, d_g be integers such that $8|Q$ and $d_g = \lceil \log_{B_g} Q\rceil$, and let r be a positive real. Then there exists an algorithm $\mathsf{BrKeyGen}$ that takes secret keys $\mathbf{s} \in \mathbb{Z}_q^n$ and $z \in \mathcal{R}_Q$ as inputs and outputs a blind rotation key $\mathbf{brk}$ of size $2d_g(2n+w+1)N\log Q$ bits. Here, $\mathbf{brk}$ consists of a group of fresh RLWE ciphertexts with error distribution $\mathcal{D}_{\mathcal{R},r}$. Meanwhile, for a test polynomial $\mathsf{testP} \in \mathcal{R}_Q$, given an all-odd LWE ciphertext $ct = (\mathbf{a}, b) \in \mathbb{Z}_{2N}^n \times \mathbb{Z}_{2N}$ under the secret key $\mathbf{s} \in \mathbb{Z}_q^n$, the blind rotation algorithm $\mathsf{BlindRotate}(\mathsf{testP}, ct, \mathbf{brk})$ outputs an RLWE ciphertext $(\tilde{a}, -\tilde{a}\cdot z + e_{br} + \mathsf{testP}\cdot X^{b+\langle \mathbf{a},\mathbf{s}\rangle})$ for some $\tilde{a} \in \mathcal{R}_Q$ with $\|e_{br}\|_\infty \le (3n + \frac{N-n}{w})\cdot d_g \cdot B_g \cdot N \cdot r \cdot \log n$.*

3 Multi-hop Proxy Re-Encryption

In this section, we first review the definition and security models of mPREs. Then, we present our unbounded mPRE construction and analyze its security. Finally, we give a brief comparison between schemes in [13, 28] and ours.

3.1 Definitions and Security Models

Recall that an mPRE scheme consists of a tuple of PPT algorithms

$$\mathsf{mPRE} = \{\mathsf{Setup}, \mathsf{KeyGen}, \mathsf{Enc}, \mathsf{ReKeyGen}, \mathsf{ReEnc}, \mathsf{Dec}\}.$$

Here, $\mathsf{Setup}(1^\lambda)$ takes a security parameter λ as input and outputs a set of public parameters pp [3]. The key generation algorithm $\mathsf{KeyGen}(\cdot)$ generates a public/secret key pair $(pk^{(i)}, sk^{(i)})$ for user i. Anyone can use $\mathsf{Enc}(pk^{(i)}, m)$ to generate a ciphertext $ct^{(i)}$ for user i. A user i can use $\mathsf{ReKeyGen}(sk^{(i)}, pk^{(j)})$ to generate a re-encryption key $rk_{i\to j}$ from himself to another user j with the help of his secret key $sk^{(i)}$. With a re-encryption key $rk_{i\to j}$, a semi-honest proxy could invoke $\mathsf{ReEnc}(pk^{(i)}, rk_{i\to j}, ct^{(i)})$ to convert i's ciphertext into one that can be decrypted by j. The algorithm $\mathsf{Dec}(sk^{(i)}, ct^{(i)})$ is used for decryption.

The correctness requires that for any $pp \leftarrow \mathsf{Setup}(1^\lambda)$, any $(pk^{(i)}, sk^{(i)}) \leftarrow \mathsf{KeyGen}(\cdot)$, any message $m \in \mathcal{M}$ and ciphertext $ct^{(i)} \leftarrow \mathsf{Enc}(pk^{(i)}, m)$, it holds that $\mathsf{Pr}[\mathsf{Dec}(sk^{(i)}, ct^{(i)}) \neq m] \leq \mathsf{negl}(\lambda)$. Meanwhile, it also requires that for any $L \in \mathbb{N}$ (note that this reflects the unbounded property), any $pp \leftarrow \mathsf{Setup}(1^\lambda)$, any $(pk^{(i_j)}, sk^{(i_j)}) \leftarrow \mathsf{KeyGen}(\cdot)$ for $j \in [0, L]$, $ct^{(i_0)} \leftarrow \mathsf{Enc}(pk^{(i_0)}, m)$ with any $m \in \mathcal{M}$, and re-encryption hops $ct^{(i_0)} \xrightarrow{rk_{i_0\to i_1}} ct^{(i_1)} \xrightarrow{rk_{i_1\to i_2}} \cdots \xrightarrow{rk_{i_{L-1}\to i_L}} ct^{(i_L)}$, in which

$$rk_{i_{j-1}\to i_j} \leftarrow \mathsf{ReKeyGen}(sk^{(i_{j-1})}, pk^{(i_j)}) \text{ and } ct^{(i_j)} \leftarrow \mathsf{ReEnc}(pk^{(i_{j-1})}, rk_{i_{j-1}\to i_j}, ct^{(i_{j-1})}),$$

it holds that for all $j \in [0, L]$, $\mathsf{Pr}[\mathsf{Dec}(sk^{(i_j)}, ct^{(i_j)}) \neq m] \leq \mathsf{negl}(\lambda)$.

HRA Security. We consider the adaptive HRA security for mPREs as in [13], and define corresponding security game (denoted by $\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$) in Fig. 4.

Definition 1 (HRA security for multi-hop PRE). *An mPRE scheme* mPRE *is* HRA *secure, if for any* PPT *adversary* $\mathcal{A}$ *and any polynomial* $\mathfrak{n}$*, it holds that* $\mathsf{Adv}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(\lambda) \coloneqq |\mathsf{Pr}[\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}} = 1] - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$.

In $\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$, we use $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk}, \mathcal{Q}_c) = 1$ to indicate that there exists a chain of index pairs $(i^*, j_1, \cdot), (j_1, j_2, \cdot), \cdots, (j_{t-1}, j_t, \cdot) \in \mathcal{Q}_{rk}$ such that $j_t \in \mathcal{Q}_c$ for some $t \geq 1$, and 0 otherwise. This condition ensures the exclusion of **all** possible trivial attacks for $\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$. However, in general cases, the information recorded in $\mathcal{Q}_{rk}$ may be too complicated for one to perform $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk}, \mathcal{Q}_c)$ in PPT. **This is the main reason** why we leverage the FKKP framework to prove adaptive HRA security of our mPREs. As a result, the reduction loss in

[3] For simplicity, we regard pp as a default input of other algorithms, and will omit it.

$\underline{\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}:}$

Initial. Initialize an empty set $\mathcal{Q}_{rk}$ to record re-encryption key queries and certain re-encryption queries that meet specific conditions. Initialize empty sets $\mathcal{Q}_c, \mathcal{L}, \mathcal{L}^*$ to record corruption queries, honestly generated ciphertexts, and derivatives of the challenge ciphertext respectively. It also maintains an index i^* of the challenge user which is initialized to be $\perp$ and a counter ctr which is initialized to be 0.
Setup. The challenger $\mathcal{C}$ runs $pp \leftarrow \mathsf{Setup}(1^\lambda)$, $(pk^{(i)}, sk^{(i)}) \leftarrow \mathsf{KeyGen}(\cdot)$ for $i \in [\mathfrak{n}]$ and returns public keys $\{pk^{(i)}\}_{i\in[\mathfrak{n}]}$ to the adversary $\mathcal{A}$.
Query phase 1. The adversary $\mathcal{A}$ can adaptively query the following oracles:

- Re-encryption key oracle $\mathcal{O}_{\mathsf{ReKey}}(i,j)$: Upon receiving an index pair (i,j), if there exists $(i,j,\cdot)$ in $\mathcal{Q}_{rk}$, $\mathcal{C}$ retrieves $(i,j,rk_{i\to j})$ from $\mathcal{Q}_{rk}$ and returns $rk_{i\to j}$ to $\mathcal{A}$. Else, if $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk} \cup \{(i,j,\cdot)\}, \mathcal{Q}_c) = 1$, $\mathcal{C}$ returns $\perp$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ generates re-encryption key $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(sk^{(i)}, pk^{(j)})$, records $(i,j,rk_{i\to j})$ in $\mathcal{Q}_{rk}$, and returns $rk_{i\to j}$ to $\mathcal{A}$.
- Corruption oracle $\mathcal{O}_{\mathsf{Cor}}(i)$: On input an index i, if $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk}, \mathcal{Q}_c \cup \{i\}) = 0$ and $i \neq i^*$, $\mathcal{C}$ records i in $\mathcal{Q}_c$ and returns $sk^{(i)}$ to $\mathcal{A}$. Otherwise, $\mathcal{C}$ returns $\perp$.
- Encryption oracle $\mathcal{O}_{\mathsf{Enc}}(i,m)$: Upon receiving an index i and a message m, the challenger $\mathcal{C}$ computes $ct^{(i)} \leftarrow \mathsf{Enc}(pk^{(i)}, m)$, increments ctr, then records $(\mathsf{ctr}, i, m, ct^{(i)})$ in $\mathcal{L}$ and returns $(\mathsf{ctr}, ct^{(i)})$ to $\mathcal{A}$.
- Re-encryption oracle $\mathcal{O}_{\mathsf{ReEnc}}(i,j,k)$: On input indexes i, j and a counter number k, if $(k,i) \in \mathcal{L}^*$ and $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk} \cup \{(i,j,\cdot)\}, \mathcal{Q}_c) = 1$, $\mathcal{C}$ returns $\perp$. Else, if $\mathcal{C}$ fails to retrieve (k,i,m,ct') from $\mathcal{L}$, $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ computes $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(sk^{(i)}, pk^{(j)})$ and $ct^{(j)} \leftarrow \mathsf{ReEnc}(pk^{(i)}, rk_{i\to j}, ct^{(i)})$, then increments ctr and records $(\mathsf{ctr}, j, m, ct^{(j)})$ in $\mathcal{L}$. Additionally, if $(k,i) \in \mathcal{L}^*$, $\mathcal{C}$ adds (ctr, j) to $\mathcal{L}^*$ and $(i,j,rk_{i\to j})$ to $\mathcal{Q}_{rk}$. Finally, returns $(\mathsf{ctr}, ct^{(j)})$ to $\mathcal{A}$.

Challenge phase. Upon receiving a challenge query (i^*, m_0, m_1, st) from $\mathcal{A}$, if $i^* \in \mathcal{Q}_c$ or $\mathsf{CheckTA}(i^*, \mathcal{Q}_{rk}, \mathcal{Q}_c) = 1$, $\mathcal{C}$ returns a bit $b \xleftarrow{\$} \{0,1\}$. Otherwise, $\mathcal{C}$ chooses a random bit $\beta \in \{0,1\}$ and computes $ct^* \leftarrow \mathsf{Enc}(pk^{(i^*)}, m_\beta)$. Then, $\mathcal{C}$ increments ctr, records $(\mathsf{ctr}, i^*, m_\beta, ct^*)$ in $\mathcal{L}$ and (ctr, i^*) in $\mathcal{L}^*$. Finally, $\mathcal{C}$ returns ct^* to $\mathcal{A}$.
Query phase 2. $\mathcal{A}$ can continue to make adaptive queries as in **Query phase 1**.
Guess. $\mathcal{A}$ outputs a bit β', the experiment outputs 1 if $\beta' = \beta$ and 0 otherwise.

Fig. 4. The HRA security experiment $\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$

corresponding HRA security proof will be inevitably large for complex DAGs. Nerveless, we remark that other mPREs may suffer similar problems [12, 22], as they all rely on computational assumptions to establish security and require the challenger to exclude "trivial wins" in corresponding security games within PPT.

Below, we recall three security notions from the FKKP framework [13] that are used to prove adaptive HRA security of multi-hop PRE.

Indistinguishability. The IND security of multi-hop PRE considers the indistinguishability of ciphertexts in a multi-challenge setting, where the adversary is given no re-encryption keys.

Definition 2 (IND Security [13,29]). *An mPRE scheme* mPRE *is* IND *secure, if for any* PPT *adversary* $\mathcal{A}$*, it holds that* $\mathsf{Adv}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}} = 1] - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Exp}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}$ *is defined in Fig. 5.*

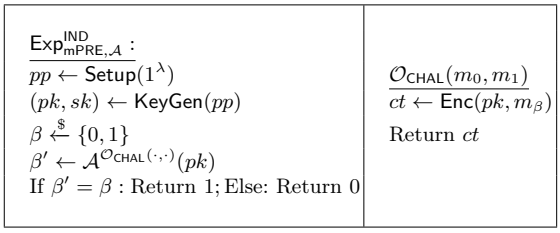

Fig. 5. The indistinguishability experiment $\mathsf{Exp}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}$ for mPRE.

Weak Key-Privacy. Roughly speaking, the wKP security requires that there exists a PPT algorithm $\mathsf{ReKeyGen}^*$ which can simulate the generation of re-encryption keys $rk_{0\to j}$ without the knowledge of $sk^{(0)}$.

Definition 3 (wKP Security [13, 29]). *An mPRE scheme* mPRE *has weak key-privacy, if there exists a* PPT *simulation algorithm* $\mathsf{ReKeyGen}^*$ *such that for any* PPT *adversary* $\mathcal{A}$ *and any polynomial* $\mathfrak{n}$*, it holds that* $\mathsf{Adv}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(\lambda) := |\Pr[\mathsf{Exp}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}} = 1] - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Exp}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$ *is defined in Fig. 6.*

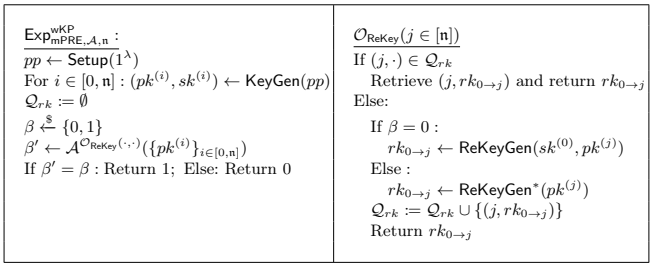

Fig. 6. The weak key-privacy experiment $\mathsf{Exp}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$ for mPRE.

Source-Hiding. Roughly speaking, the SH security requires fresh ciphertexts to be statistically indistinguishable from re-encrypted ciphertexts, even if the adversary is given the secret keys for the source and target public keys, as well as the corresponding re-encryption keys.

Definition 4 (SH Security). *An mPRE scheme* mPRE *has the property of source-hiding, if for any (unbounded) adversary* $\mathcal{A}$*, it holds that* $\mathsf{Adv}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}}(\lambda) \coloneqq |\mathsf{Pr}[\mathsf{Exp}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}} = 1] - \frac{1}{2}| \leq \mathsf{negl}(\lambda)$*, where* $\mathsf{Exp}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}}$ *is defined in Fig. 7.*

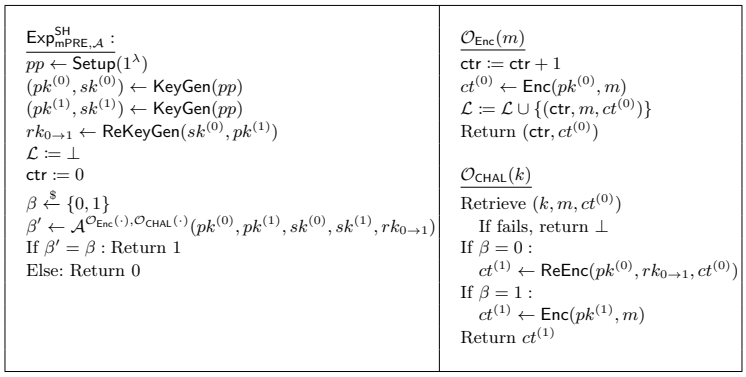

Fig. 7. The source-hiding experiment $\mathsf{Exp}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}}$ for mPRE.

Below, we revisit the theorem from [13] that provides a generic framework for achieving adaptive HRA security of mPREs. Notably, in the proof reduction, the number of challenge queries Q_{CHAL} in IND experiment is 1.

Theorem 1 (IND+wKP+SH $\implies$ HRA for multi-hop PRE [13]). *If an mPRE scheme* mPRE *has* IND, wKP *and* SH *security, then it is also* HRA *secure. More precisely, for any polynomial* $\mathfrak{n}$*, for any* PPT *adversary* $\mathcal{A}$ *that makes at most polynomial queries to* $\mathcal{O}_{\mathsf{ReKey}}$ *and* $\mathcal{O}_{\mathsf{ReEnc}}$*, and forms a challenge graph* G [4] *in* $\mathcal{G}(\mathfrak{n}, \delta, d)$*, there exist* PPT *algorithms* $\mathcal{B}, \mathcal{B}'$*, and* $\mathcal{B}''$ *such that* $\mathsf{Adv}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(\lambda) \leq (\mathsf{Adv}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{B}}(\lambda) + 2\tau \cdot \mathsf{Adv}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{B}'}(\lambda)) \cdot \mathfrak{n}^{\sigma+\delta+1} + 2\mathfrak{n}(\mathfrak{n}-1)(Q_{\mathsf{E}} + Q_{\mathsf{RE}})Q_{\mathsf{RE}} \cdot \mathsf{Adv}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{B}''}(\lambda)$*, where* Q_{RE} *and* Q_{E} *are the upper bounds of the* $\mathcal{O}_{\mathsf{ReEnc}}$ *and* $\mathcal{O}_{\mathsf{Enc}}$ *queries,* δ *denotes the outdegree,* d *the depth,* τ *and* σ *denote the time complexity and space complexity of the pebbling game (see App. A.3 for more details) for* $\mathcal{G}(\mathfrak{n}, \delta, d)$*, respectively.*

3.2 Construction of Our Unbounded mPREs

In this subsection, we use the following parameters:

– A positive integer n, representing the dimension of lattices/LWE problems, is also used as the security parameter for simplicity; a power of two integer N represents the RLWE dimension, two moduli q and Q with $8|Q$ (one for plain LWE and one for RLWE), an integer $l = \lceil \log q \rceil$.

[4] Note that the users $[\mathfrak{n}]$ and $\mathcal{Q}_{rk}$ form a directed graph in $\mathsf{Exp}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}$, we define the subgraph that is reachable from the challenger user as the challenge graph.

- A base integer B_g with $d_g := \lceil \log_{B_g} Q \rceil$, a window size $w < n$ such that $N < (\frac{w}{2} + 1) \cdot n$, a test polynomial $\mathsf{testP} := \frac{Q}{8} \cdot X^{\frac{3N}{4}}(1 + X + \cdots + X^{N-1})$ and a distribution $\tilde{\chi} = \mathcal{D}_{\mathbb{Z},\tilde{r}}$ for $\mathsf{BrKeyGen}$ (See Lemma 2 for more details).
- $\hat{\chi} = \mathcal{D}_{\mathbb{Z},\hat{r}}$ is a distribution for corresponding LWE problems.
- $\chi = \mathcal{D}_{\mathbb{Z},r}$ is a distribution with r satisfying

$$\left[\frac{q}{Q}(3n + \frac{N-n}{w})d_g B_g N\tilde{r}\log n + N\hat{r}\log n(l + \frac{1}{2}) + 1\right]/r\log n = \mathsf{negl}(n),$$

  which are mainly used for showing that fresh ciphertexts are indistinguishable from re-encrypted ciphertexts by using the error smudging lemma.

To ensure the correctness of encryption and re-encryption, we require

$$q \geq \frac{8N}{N - 4(1 + n\hat{r}\log n)} \cdot \left[\frac{q}{Q} \cdot (3n + \frac{N-n}{w})d_g B_g N\tilde{r}\log n + \frac{1 + N\hat{r}\log n}{2} + r\log n + 2n\hat{r}r + Nl \cdot (2n \cdot \hat{r}^2 + \hat{r}\log n)\right].$$

A possible choice of parameters is: $N = O(n^{1.5}\log n), \tilde{r} = O(\sqrt{n}), Q = O(2^{\sqrt[4]{n}\cdot\log n} \cdot n^3), \hat{r} = O(\sqrt{n}), r = O(2^{\sqrt[4]{n}} \cdot n^{2.25})$, $d_g = O(\log n)$. Given the parameters defined above, it's evident that $q = O(2^{\sqrt[4]{n}} \cdot n^{3.75})$ is sufficient. We set $\hat{r} = r' \cdot q \geq 2\sqrt{n}$, and a possible choice for r' is $O(2^{-\sqrt[4]{n}} \cdot n^{-3.25})$. Consequently, there exists a quantum reduction from worst-case SIVP_γ to LWE problem for $\gamma = O(\frac{n}{r'}) = O(2^{\sqrt[4]{n}} \cdot n^{4.25})$. Therefore, our scheme is HRA secure, assuming the hardness of $\mathsf{SIVP}_{O(2^{\sqrt[4]{n}} \cdot n^{4.25})}$ [5] problem and circular security assumption.

Our unbounded mPRE scheme with HRA security is proposed as follows:

- $\mathsf{Setup}(\cdot)$: It selects large enough n (and N), and outputs the public parameter $pp = (n, N, q, Q, \chi, \hat{\chi}, \tilde{\chi}, l, w, B_g, d_g, \mathsf{testP})$ as defined above.
- $\mathsf{KeyGen}(\cdot)$: It generates public/secret keys of user i as follows: (1) Sample a polynomial $z_i \in \mathcal{R}_Q$ with coefficients obeying to $\hat{\chi}$, and sample $\mathbf{s}_i \xleftarrow{\$} \hat{\chi}^n$; (2) Sample $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$, $\mathbf{e}_i \xleftarrow{\$} \hat{\chi}^n$, and compute $\mathbf{b}_i = -\mathbf{A}_i^\top \mathbf{s}_i + \mathbf{e}_i \in \mathbb{Z}_q^n$; (3) Run $\mathsf{BrKeyGen}(\mathbf{s}_i, z_i)$ to generate blind rotation keys $\mathbf{brk}^{(i)}$ as in Lemma 2; (4) Set $\widetilde{\mathbf{pk}}^{(i)} = (\mathbf{A}_i, \mathbf{b}_i) \in \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^n$, and output $pk^{(i)} = \{\widetilde{\mathbf{pk}}^{(i)}, \mathbf{brk}^{(i)}\}$ and $sk^{(i)} = (\mathbf{s}_i, \bar{\mathbf{z}}_i)$, where $\bar{\mathbf{z}}_i$ is the coefficient vector of z_i.
- $\mathsf{Enc}(pk^{(i)}, m)$: On input the public key $pk^{(i)}$ and a message $m \in \{0, 1\}$, it randomly chooses $\mathbf{r}_i, \mathbf{e}_{i,1} \xleftarrow{\$} \chi^n$ and $e_{i,2} \xleftarrow{\$} \chi$. Then, it returns a ciphertext $ct^{(i)} = (\mathbf{a}^{(i)}, b^{(i)}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where $\mathbf{a}^{(i)} = \mathbf{A}_i\mathbf{r}_i + \mathbf{e}_{i,1} \in \mathbb{Z}_q^n$ and $b^{(i)} = \mathbf{b}_i^\top \mathbf{r}_i + e_{i,2} + \lfloor \frac{q}{4} \rceil \cdot m \in \mathbb{Z}_q$.

[5] Notice that this bound is normal as security of HEs are usually based on non-standard LWE assumptions with the ratio of Gaussian parameters and modulus being super-polynomial [1, 8, 28].

– $\mathsf{ReKeyGen}(sk^{(i)}, pk^{(j)})$: On input the secret key $sk^{(i)}$ of a user i, and the public key $pk^{(j)}$ of another user j, it samples $\mathbf{R}_{i,j}, \mathbf{E}'_{i,j} \xleftarrow{\$} \hat{\chi}^{n\times Nl}$ and $\mathbf{e}'_{i,j} \xleftarrow{\$} \hat{\chi}^{Nl}$. Then, it outputs the re-encryption key:

$$rk_{i\to j} = \begin{bmatrix} \mathbf{A}_j\mathbf{R}_{i,j} + \mathbf{E}'_{i,j} & \mathbf{0}_{n\times 1} \\ \mathbf{b}_j^\top \mathbf{R}_{i,j} + \mathbf{e}_{i,j}'^\top + \mathsf{Powersof2}(\bar{\mathbf{z}}_i)^\top & 1 \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times(Nl+1)}.$$

– $\mathsf{ReEnc}(pk^{(i)}, rk_{i\to j}, ct^{(i)})$: On input the public key $pk^{(i)}$, the re-encryption key $rk_{i\to j}$ and a ciphertext $ct^{(i)}$, it generates a ciphertext $ct^{(j)}$ of user j as follows:
  - Run $\mathsf{ModSwitch}^{\mathsf{odd}}_{q,2N}(ct^{(i)})$ to obtain an LWE ciphertext $\check{ct}^{(i)} = (\check{\mathbf{a}}^{(i)}, \check{b}^{(i)}) \in \mathbb{Z}_{2N}^n \times \mathbb{Z}_{2N}$ under the secret $\mathbf{s}_i$;
  - Run $\mathsf{BlindRotate}(\check{ct}^{(i)}, \mathbf{brk}^{(i)}, \mathsf{testP})$ to obtain an RLWE ciphertext $\tilde{c}^{(i)} \in \mathcal{R}_Q^2$ under the secret key z_i by using Lemma 2;
  - Run $\mathsf{SampleExtract}(\tilde{c}^{(i)} + (0, \frac{Q}{8}))$ to obtain an LWE ciphertext $\hat{ct}^{(i)} = (\hat{\mathbf{a}}^{(i)}, \hat{b}^{(i)}) \in \mathbb{Z}_Q^N \times \mathbb{Z}_Q$ under the secret key $\bar{\mathbf{z}}_i$;
  - Run $\mathsf{ModSwitch}_{Q,q}(\hat{ct}^{(i)})$ to obtain an LWE ciphertext $\bar{ct}^{(i)} = (\bar{\mathbf{a}}^{(i)}, \bar{b}^{(i)}) \in \mathbb{Z}_q^N \times \mathbb{Z}_q$ under the same secret key $\bar{\mathbf{z}}_i$;
  - Sample $\bar{\mathbf{r}}_j, \bar{\mathbf{e}}_{j,1} \xleftarrow{\$} \chi^n$, $\bar{e}_{j,2} \xleftarrow{\$} \chi$, output a ciphertext $ct^{(j)} = (\mathbf{a}^{(j)}, b^{(j)}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ for j, where $\mathbf{a}^{(j)}$ and $b^{(j)}$ are generated as follows:

$$\begin{bmatrix} \mathbf{a}^{(j)} \\ b^{(j)} \end{bmatrix} = rk_{i\to j} \cdot \begin{bmatrix} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) \\ \bar{b}^{(i)} \end{bmatrix} + \begin{bmatrix} \mathbf{A}_j\bar{\mathbf{r}}_j + \bar{\mathbf{e}}_{j,1} \\ \mathbf{b}_j^\top \bar{\mathbf{r}}_j + \bar{e}_{j,2} \end{bmatrix} \in \mathbb{Z}_q^{n+1}.$$

– $\mathsf{Dec}(sk^{(i)}, ct^{(i)})$: On input the secret key $sk^{(i)}$ and a ciphertext $ct^{(i)} = (\mathbf{a}^{(i)}, b^{(i)})$, it outputs $m = \lfloor \frac{4}{q} \cdot (b^{(i)} + \mathbf{s}_i^\top \mathbf{a}^{(i)}) \rceil$.

3.3 Correctness and Security

Let's begin by analyzing the correctness of our scheme.

Correctness of Fresh Ciphertexts. For a fresh ciphertext $ct^{(i)} = (\mathbf{a}^{(i)}, b^{(i)})$ generated by $\mathsf{Enc}(pk^{(i)}, m)$, we have $\mathbf{a}^{(i)} = \mathbf{A}_i\mathbf{r}_i + \mathbf{e}_{i,1} \in \mathbb{Z}_q^n$ and $b^{(i)} = \mathbf{b}_i^\top \mathbf{r}_i + e_{i,2} + \lfloor \frac{q}{4} \rceil \cdot m \in \mathbb{Z}_q$. Therefore, we can obtain that

$$\begin{aligned} \lfloor \frac{4}{q} \cdot (b^{(i)} + \mathbf{s}_i^\top \mathbf{a}^{(i)}) \rceil &= \lfloor \frac{4}{q} \cdot (\mathbf{b}_i^\top \mathbf{r}_i + e_{i,2} + \lfloor \frac{q}{4} \rceil \cdot m + \mathbf{s}_i^\top \mathbf{A}_i\mathbf{r}_i + \mathbf{s}_i^\top \mathbf{e}_{i,1}) \rceil \\ &= \lfloor m + \frac{4}{q} \cdot (\mathbf{e}_i^\top \mathbf{r}_i + e_{i,2} + \mathbf{s}_i^\top \mathbf{e}_{i,1} + \varepsilon) \rceil. \end{aligned}$$

Here $\varepsilon \in (-\frac{1}{2}, \frac{1}{2}]$, and we can recover the original message m as long as

$$|e^{(i)}| := |\mathbf{e}_i^\top \mathbf{r}_i + e_{i,2} + \mathbf{s}_i^\top \mathbf{e}_{i,1} + \varepsilon| < (2n \cdot \hat{r} + \log n) \cdot r + \frac{1}{2} < \frac{q}{8}.$$

Correctness of Re-Encrypted Ciphertexts. For a re-encrypted ciphertext $ct^{(j)}$ generated by $\mathsf{ReEnc}(pk^{(i)}, rk_{i\to j}, ct^{(i)})$ where $rk_{i\to j} \leftarrow \mathsf{ReKeyGen}(sk^{(i)}, pk^{(j)})$

and $ct^{(i)} \leftarrow \mathsf{Enc}(pk^{(i)}, m)$, let's analyze the re-encryption process. It first computes an all-odd LWE ciphertext $\check{ct}^{(i)} = (\check{\mathbf{a}}^{(i)}, \check{b}^{(i)})$ with modulus $2N$ satisfying that $\check{b}^{(i)} + \langle \check{\mathbf{a}}^{(i)}, \mathbf{s}_i \rangle = \lfloor \frac{2N}{q} b^{(i)} \rceil_{\mathsf{odd}} + \langle \lfloor \frac{2N}{q} \mathbf{a}^{(i)} \rceil_{\mathsf{odd}}, \mathbf{s}_i \rangle = \frac{N}{2} m + \frac{2N}{q} e^{(i)} + e^{(i)}_{ms1}$, where $e^{(i)}_{ms1} = (\lfloor \frac{2N}{q} b^{(i)} \rceil_{\mathsf{odd}} - \frac{2N}{q} b^{(i)}) + \langle \lfloor \frac{2N}{q} \mathbf{a}^{(i)} \rceil_{\mathsf{odd}} - \frac{2N}{q} \mathbf{a}^{(i)}, \mathbf{s}_i \rangle$. After the blind rotation algorithm, we obtain an RLWE ciphertext

$$\tilde{c}^{(i)} = (\tilde{a}^{(i)}, -\tilde{a}^{(i)} \cdot z_i + e^{(i)}_{br} + \mathsf{testP} \cdot X^{\frac{N}{2} m + \frac{2N}{q} e^{(i)} + e^{(i)}_{ms1}}) \in \mathcal{R}^2_Q$$

for some $\tilde{a}^{(i)} \in \mathcal{R}_Q$ with $\|e^{(i)}_{br}\|_\infty \leq (3n + \frac{N-n}{w}) \cdot d_g N B_g \tilde{r} \cdot \log n$ by Lemma 2. Notice that $\mathsf{testP} = \frac{Q}{8} \cdot X^{\frac{3N}{4}} (1 + X + \cdots + X^{N-1})$, as long as $|\frac{2N}{q} e^{(i)} + e^{(i)}_{ms1}| < \frac{N}{4}$, the constant term of $\mathsf{testP} \cdot X^{\frac{N}{2} m + \frac{2N}{q} e^{(i)} + e^{(i)}_{ms1}}$ is exactly $-\frac{Q}{8}$ if $m = 0$ and $\frac{Q}{8}$ if $m = 1$. Then, according to the sample extraction algorithm, the LWE ciphertext $\hat{ct}^{(i)} = (\hat{\mathbf{a}}^{(i)}, \hat{b}^{(i)}) \leftarrow \mathsf{SampleExtract}(\tilde{c}^{(i)} + (0, \frac{Q}{8}))$ satisfies that $\hat{b}^{(i)} + \langle \hat{\mathbf{a}}^{(i)}, \bar{\mathbf{z}}_i \rangle = \frac{Q}{4} \cdot m + e^{(i)}_{br,0} \pmod{Q}$, where $e^{(i)}_{br,0}$ is the constant term of $e^{(i)}_{br}$. After modulus switching, the ciphertext $\bar{ct}^{(i)} = (\bar{\mathbf{a}}^{(i)}, \bar{b}^{(i)}) \leftarrow \mathsf{ModSwitch}_{Q,q}(\hat{ct}^{(i)})$ satisfies that

$$\bar{b}^{(i)} + \langle \bar{\mathbf{a}}^{(i)}, \bar{\mathbf{z}}_i \rangle = \frac{q}{4} \cdot m + \frac{q}{Q} \cdot e^{(i)}_{br,0} + e^{(i)}_{ms2} \pmod{q},$$

where $e^{(i)}_{ms2} = (\lfloor \frac{q}{Q} \hat{b}^{(i)} \rceil - \frac{q}{Q} \hat{b}^{(i)}) + \langle \lfloor \frac{q}{Q} \hat{\mathbf{a}}^{(i)} \rceil - \frac{q}{Q} \hat{\mathbf{a}}^{(i)}, \bar{\mathbf{z}}_i \rangle$. Let $e_{sum} = \frac{q}{Q} \cdot e^{(i)}_{br,0} + e^{(i)}_{ms2}$, then we have

$$\begin{aligned}
\begin{bmatrix} \mathbf{a}^{(j)} \\ b^{(j)} \end{bmatrix} &= \begin{bmatrix} \mathbf{A}_j \mathbf{R}_{i,j} + \mathbf{E}'_{i,j} & \mathbf{0}_{n \times 1} \\ \mathbf{b}_j^\top \mathbf{R}_{i,j} + \mathbf{e}'^\top_{i,j} + \mathsf{Powersof2}(\bar{\mathbf{z}}_i)^\top & 1 \end{bmatrix} \cdot \begin{bmatrix} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) \\ \bar{b}^{(i)} \end{bmatrix} + \begin{bmatrix} \mathbf{A}_j \bar{\mathbf{r}}_j + \bar{\mathbf{e}}_{j,1} \\ \mathbf{b}_j^\top \bar{\mathbf{r}}_j + \bar{e}_{j,2} \end{bmatrix} \\
&= \begin{bmatrix} \mathbf{A}_j [\mathbf{R}_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + \bar{\mathbf{r}}_j] + \mathbf{E}'_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + \bar{\mathbf{e}}_{j,1} \\ \mathbf{b}_j^\top [\mathbf{R}_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + \bar{\mathbf{r}}_j] + \mathbf{e}'^\top_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + e_{sum} + \bar{e}_{j,2} + \frac{q}{4} \cdot m \end{bmatrix}.
\end{aligned}$$

As a result,

$$\begin{aligned}
b^{(j)} + \langle \mathbf{a}^{(j)}, \mathbf{s}_j \rangle =& \frac{q}{4} \cdot m + e_{sum} + \bar{e}_{j,2} + \mathbf{s}_j^\top \bar{\mathbf{e}}_{j,1} + \mathbf{s}_j^\top \mathbf{E}'_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) \\
&+ \mathbf{e}_j^\top (\mathbf{R}_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + \bar{\mathbf{r}}_j) + \mathbf{e}'^\top_{i,j} \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) \pmod{q}.
\end{aligned}$$

Then by a similar argument, the decryption algorithm recovers m as long as

$$\begin{aligned}
|e^{(j)}| :=& |e_{sum} + \bar{e}_{j,2} + \mathbf{s}_j^\top \bar{\mathbf{e}}_{j,1} + (\mathbf{s}_j^\top \mathbf{E}'_{i,j} + \mathbf{e}_j^\top \mathbf{R}_{i,j} + \mathbf{e}'^\top_{i,j}) \mathsf{BitDecomp}(\bar{\mathbf{a}}^{(i)}) + \mathbf{e}_j^\top \bar{\mathbf{r}}_j| \\
<& \frac{q}{Q} \cdot (3n + \frac{N-n}{w}) d_g B_g N \tilde{r} \log n + \frac{1 + N\hat{r} \log n}{2} + r \log n \\
&+ 2n\hat{r}r + Nl \cdot (2n \cdot \hat{r}^2 + \hat{r} \log n) < \frac{q}{8}. \qquad (1)
\end{aligned}$$

Suppose that $ct^{(j)}$ is further re-encrypted to $ct^{(k)} \leftarrow \mathsf{ReEnc}(pk^{(j)}, rk_{j \to k}, ct^{(j)})$, where $rk_{j \to k} \leftarrow \mathsf{ReKeyGen}(sk^{(j)}, pk^{(k)})$. For the correctness of $\mathsf{BlindRotate}(\check{ct}^{(j)}, \mathbf{brk}^{(j)}, \mathsf{testP})$, we further need that $|\frac{2N}{q} e^{(j)} + e^{(j)}_{ms1}| < \frac{N}{4}$, i.e.,

$$\begin{aligned}
\frac{2N}{q} \Bigg[& \frac{q}{Q} \cdot (3n + \frac{N-n}{w}) d_g B_g N \tilde{r} \log n + \frac{1 + N\hat{r} \log n}{2} + r \log n \\
& + 2n\hat{r}r + Nl \cdot (2n \cdot \hat{r}^2 + \hat{r} \log n) \Bigg] + 1 + n\hat{r} \log n < \frac{N}{4}. \qquad (2)
\end{aligned}$$

Note that the input ciphertext $ct^{(j)}$ is refreshed after the blind rotation procedure and the error $e_{br}^{(j)}$ of the ciphertext $\tilde{c}^{(j)} \leftarrow \mathsf{BlindRotate}(ct^{(j)}, \mathbf{brk}^{(j)}, \mathsf{testP})$ is independent from the input error $\frac{2N}{q}e^{(j)} + e_{ms1}^{(j)}$. By a similar analysis, the decryption recovers m with the same requirement as the single-hop ciphertext $ct^{(j)}$. Therefore, our mPRE scheme is correct for our choice of q that satisfies (2).

Below we demonstrate the adaptive HRA security of our mPRE scheme by showing the corresponding IND security, wKP security and SH security.

Theorem 2 (IND security of mPRE). *Under the LWE, RLWE and circular security assumptions, our* mPRE *scheme is* IND *secure. More precisely, for any* PPT *adversary* $\mathcal{A}$ *that makes at most* Q_{CHAL} *queries to* $\mathcal{O}_{\mathsf{CHAL}}$*, there exist* PPT *algorithm* $\mathcal{B}$ *and* $\mathcal{B}''$ *against the LWE assumption such that*

$$\mathsf{Adv}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}(n) \leq \mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^{n}_{n,q,\hat{\chi}}) + Q_{\mathsf{CHAL}} \cdot \mathsf{Adv}_{\mathcal{B}''}(\mathsf{LWE}^{n+1}_{n,q,\chi}).$$

Proof (Proof Sketch). The IND security requires the indistinguishability of the ciphertext for adversary who has no knowledge of re-encryption keys or secret keys. Therefore, it suffices to show the semantic security of the underlying scheme, i.e. the challenge ciphertext is computationally indistinguishable from a uniform distribution. A detailed proof is provided in App. B.1. □

Theorem 3 (wKP security of mPRE). *Under the LWE, RLWE and corresponding circular security assumptions, our* mFHE *scheme has* wKP *security. More precisely, for any* PPT *adversary* $\mathcal{A}$ *and for any polynomial* $\mathfrak{n}$*, there exist* PPT *algorithms* $\mathcal{B}$ *and* $\mathcal{B}'$ *against the LWE assumption such that*

$$\mathsf{Adv}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(n) \leq \mathfrak{n} \cdot (\mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^{n}_{n,q,\hat{\chi}}) + Nl \cdot \mathsf{Adv}_{\mathcal{B}'}(\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}})).$$

Proof (Proof Sketch). This proof is somewhat straightforward as a re-encryption key $rk_{i\to j}$ is "encryptions" of $\bar{\boldsymbol{z}}_i$ under public keys of an "honest" user j. Given that wKP security ensures that honestly generated re-encryption key $rk_{i\to j}$ can be indistinguishably replaced with a simulated one from the view of an adversary who have no knowledge of $sk^{(i)}$, one could demonstrate this via several hybrid games under the LWE assumption. The formal proof is detailed in App. B.2. □

Theorem 4 (SH security of mPRE). *Our* mPRE *scheme has source-hiding property, i.e., for any adversary* $\mathcal{A}$*, it holds that* $\mathsf{Adv}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}}(n) \leq \mathsf{negl}(n)$.

Proof (Proof Sketch). To prove the SH security, it is sufficient to show the honestly re-encrypted ciphertext is statistically indistinguishable from a fresh encrypted one. Note that the re-encryption from user 0 to user 1 is of the form:

$$ct^{(1)} = \begin{bmatrix} \mathbf{A}_1[\mathbf{R}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{r}}_1] + \mathbf{E}'_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{e}}_{1,1} \\ \mathbf{b}_1^{\top}[\mathbf{R}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{r}}_1] + \mathbf{e}'^{\top}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + e_{sum} + \bar{e}_{1,2} + e_r + \lfloor \frac{q}{4} \rceil m \end{bmatrix},$$

where $\mathbf{R}_{0,1}, \mathbf{E}'_{0,1} \xleftarrow{\$} \hat{\chi}^{n\times Nl}, \mathbf{e}'_{0,1} \xleftarrow{\$} \hat{\chi}^{Nl}, \bar{\mathbf{r}}_1, \bar{\mathbf{e}}_{1,1} \xleftarrow{\$} \chi^n, \bar{e}_{1,2} \xleftarrow{\$} \chi$ and $|e_r| < \frac{1}{2}$. Under our parameter selection, we can ensure that $\bar{\mathbf{r}}_1$ smudges $\mathbf{R}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)})$,

$\bar{\mathbf{e}}_{1,1}$ smudges $\mathbf{E}'_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)})$, and $\bar{e}_{1,2}$ smudges $e_{sum}+\mathbf{e}'^{\top}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)})+e_r$ by error smudging lemma. As a result, the honestly re-encrypted ciphertext is statistically indistinguishable from a fresh encrypted one. The formal proof is provided in App. B.3.

Combining Theorem 1 with Theorem 2, Theorem 3, and Theorem 4, we have the following corollary showing the HRA security of our mPRE scheme.

Corollary 1 (HRA security of mPRE). *Under the LWE, RLWE and circular security assumptions, our* mPRE *scheme is* HRA *secure. More precisely, for any* PPT *adversary* $\mathcal{A}$ *that makes at most polynomial queries to* $\mathcal{O}_{\mathsf{ReKey}}$ *and* $\mathcal{O}_{\mathsf{ReEnc}}$ *oracles, and forms a challenge graph* G *in* $\mathcal{G}(\mathfrak{n},\delta,d)$*, there exist* PPT *algorithms* $\mathcal{B},\mathcal{B}'$ *and* $\mathcal{B}''$ *against the LWE assumption such that*

$$\begin{aligned}\mathsf{Adv}^{\mathsf{HRA}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(n) \leq & \mathfrak{n}^{\sigma+\delta+1}[(2\tau\mathfrak{n}+1)\cdot\mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^{n}_{n,q,\hat{\chi}})+2\tau\mathfrak{n}Nl\cdot\mathsf{Adv}_{\mathcal{B}'}(\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}})\\ & +\mathsf{Adv}_{\mathcal{B}''}(\mathsf{LWE}^{n+1}_{n,q,\chi})]+\mathsf{negl}(n),\end{aligned}$$

where δ *denotes the outdegree,* d *the depth,* τ *and* σ *denote the time complexity and space complexity of the pebbling game for the class* $\mathcal{G}(n,\delta,d)$*, respectively.*

3.4 Complexity Analysis

The comparison of the key sizes and computational complexity between [13, 28] and ours is shown in Table 1. Since [13] only provides a framework for the PRE construction, we instantiate it using the same blind rotation techniques as ours, along with the Ducas-Stehlé washing machine [11]. The key sizes of all schemes are expressed as functions of the security parameter n. The re-encryption key in our scheme is slightly larger than those in [13, 28], due to the fact that our re-encryption algorithm operates on i's ciphertext under a secret key $\bar{\mathbf{z}}_i \in \mathbb{Z}_q^N$, rather than under $\mathbf{s}_i \in \mathbb{Z}_q^n$. In contrast, our scheme does not require the key switching algorithm used in bootstrapping, thus eliminates the need to include key switching keys in the public key, compared to [13, 28]. Additionally, [13] requires approximately $O(n^3\log^4 n)$ bits to store $O(N\log Q)$ LWE encryptions with modulus Q and dimension N in the public key, which are used as re-randomization keys for ciphertext sanitization. Overall, the public key size in our scheme outperforms that of [13, 28].

We measure the computational complexity of the re-encryption algorithm in terms of the number of the scalar multiplications between a polynomial in $\mathcal{R}_Q$ and a vector consisting of RLWE ciphertexts, denoted by "$\odot$", as the cost of these operations dominates the total computational complexity (see App. A for more details). Specially, each "$\odot$" requires exactly (d_g+1) NTT operations and we choose a large base B_g to keep the magnitude of d_g comparable to that in [13,28]. As illustrated in [18], the number of "$\odot$" in our re-encryption algorithm is less than [13, 28], thereby achieving a better computational complexity.

4 Homomorphic PRE

Homomorphic proxy re-encryption is an extension of PRE that integrates the benefits of both homomorphic encryption (HE) and PRE. Adding the following evaluation algorithm $\mathsf{HomNAND}$, our mPRE scheme can be naturally extended to a homomorphic PRE, which enables arbitrary homomorphic computations over original, re-encrypted, and evaluated ciphertexts. Note that we only present the homomorphic evaluation of a NAND gate, as it is universal and any circuit can be evaluated through NAND gates compositions.

- $\mathsf{HomNAND}(evk^{(i)}, ct_1^{(i)}, ct_2^{(i)})$: On input the evaluation key $evk^{(i)} = \{\mathbf{brk}^{(i)}, \mathbf{ksk}_{\bar{\mathbf{z}}_i \to \mathbf{s}_i}\}$, ciphertexts $ct_1^{(i)} = (\mathbf{a}_1^{(i)}, b_1^{(i)})$ and $ct_2^{(i)} = (\mathbf{a}_2^{(i)}, b_2^{(i)})$ corresponding to messages $m_1, m_2 \in \{0,1\}$, respectively. The algorithm generates a ciphertext $ct^{(i)}$ of message $m_1 \barwedge m_2$ as follows:
  - Run $\mathsf{ModSwitch}^{\mathsf{odd}}_{q,2N}(ct_1^{(i)} + ct_2^{(i)})$ to obtain an LWE ciphertext $\check{ct}^{(i)}$.
  - Let $\mathsf{testP}' = -\frac{Q}{8} \cdot X^{\frac{N}{4}}(1 + X + \cdots + X^{N-1})$, run $\mathsf{BlindRotate}(\check{ct}^{(i)}, \mathbf{brk}^{(i)}, \mathsf{testP}')$ to obtain an RLWE ciphertext $\tilde{c}^{(i)} \in \mathcal{R}_Q^2$.
  - Run $\mathsf{SampleExtract}(\tilde{c}^{(i)} + (0, \frac{Q}{8}))$ to obtain an LWE ciphertext $\hat{ct}^{(i)} \in \mathbb{Z}_Q^N \times \mathbb{Z}_Q$ under the secret key $\bar{\mathbf{z}}_i$.
  - Run $\mathsf{KeySwitch}(\hat{ct}^{(i)}, \mathbf{ksk}_{\bar{\mathbf{z}}_i \to \mathbf{s}_i})$ to obtain an LWE ciphertext $\bar{ct}^{(i)} \in \mathbb{Z}_Q^n \times \mathbb{Z}_Q$ under the secret key $\mathbf{s}_i$.
  - Run $\mathsf{ModSwitch}_{Q,q}(\bar{ct}^{(i)})$ to obtain the LWE ciphertext $ct^{(i)} \in \mathbb{Z}_q^n \times \mathbb{Z}_q$.

Correctness. The correctness of our homomorphic PRE scheme can be proved in a manner similar to that of mPRE. For more details, please refer to App. C.
Security. The circular security assumption guarantees the usage of blind rotation keys and key switching keys. Therefore, our homomorphic PRE scheme remains HRA secure under same assumptions.

5 Conclusion

In this paper, we propose a lattice-based HRA-secure unbounded multi-hop PRE scheme that supports an unbounded number of re-encryptions, making it more suitable for practical scenarios. We introduce a modified re-encryption method that reduces both storage and computation costs. Moreover, our scheme can be extended to an unbounded HPRE scheme by adding a homomorphic NAND gate evaluation algorithm. An interesting open problem is how to construct an unbounded multi-hop fine-grained PRE scheme with HRA security to achieve fine-grained re-encryption capabilities.

References

1. Alperin-Sheriff, J., Peikert, C.: Faster bootstrapping with polynomial error. In: Garay, J.A., Gennaro, R. (eds.) CRYPTO 2014, Part I. LNCS, vol. 8616, pp. 297–314. Springer, Heidelberg (2014)

2. Asharov, G., Jain, A., López-Alt, A., Tromer, E., Vaikuntanathan, V., Wichs, D.: Multiparty computation with low communication, computation and interaction via threshold fhe. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 483–501. Springer, Heidelberg (2012)
3. 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 (2006)
4. Bennett, C.H.: Time/space trade-offs for reversible computation. SIAM J. Comput. **18**(4), 766–776 (1989)
5. Blaze, M., Bleumer, G., Strauss, M.: Divertible protocols and atomic proxy cryptography. In: Nyberg, K. (ed.) EUROCRYPT 1998. LNCS, vol. 1403, pp. 127–144. Springer, Heidelberg (1998)
6. Chandran, N., Chase, M., Liu, F.H., Nishimaki, R., Xagawa, K.: Re-encryption, functional re-encryption, and multi-hop re-encryption: a framework for achieving obfuscation-based security and instantiations from lattices. In: Krawczyk, H. (ed.) PKC 2014. LNCS, vol. 8383, pp. 95–112. Springer, Heidelberg (2014)
7. Chillotti, I., Gama, N., Georgieva, M., Izabachene, M.: Faster fully homomorphic encryption: Bootstrapping in less than 0.1 seconds. In: Cheon, J.H., Takagi, T. (eds.) ASIACRYPT 2016, Part I. LNCS, vol. 10031, pp. 3–33. Springer, Heidelberg (2016)
8. Chillotti, I., Gama, N., Georgieva, M., Izabachène, M.: Tfhe: fast fully homomorphic encryption over the torus. J. Cryptol. **33**(1), 34–91 (2020)
9. Cohen, A.: What about bob? the inadequacy of cpa security for proxy reencryption. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 287–316. Springer, Cham (2019)
10. Ducas, L., Micciancio, D.: Fhew: bootstrapping homomorphic encryption in less than a second. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015, Part I. LNCS, vol. 9056, pp. 617–640. Springer, Heidelberg (2015)
11. Ducas, L., Stehlé, D.: Sanitization of fhe ciphertexts. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part I. LNCS, vol. 9665, pp. 294–310. Springer, Heidelberg (2016)
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.) ACNS 2019. LNCS, vol. 11464, pp. 363–382. Springer, Cham (2019)
13. Fuchsbauer, G., Kamath, C., Klein, K., Pietrzak, K.: Adaptively secure proxy re-encryption. In: Lin, D., Sako, K. (eds.) PKC 2019, Part II. LNCS, vol. 11443, pp. 317–346. Springer, Cham (2019)
14. Gama, N., Izabachène, M., Nguyen, P.Q., Xie, X.: Structural lattice reduction: generalized worst-case to average-case reductions and homomorphic cryptosystems. In: Fischlin, M., Coron, J.S. (eds.) EUROCRYPT 2016, Part II. LNCS, vol. 9666, pp. 528–558. Springer, Heidelberg (2016)
15. Jafargholi, Z., Kamath, C., Klein, K., Komargodski, I., Pietrzak, K., Wichs, D.: Be adaptive, avoid overcommitting. In: Katz, J., Shacham, H. (eds.) CRYPTO 2017, Part I. LNCS, vol. 10401, pp. 133–163. Springer, Cham (2017)
16. Lai, J., Huang, Z., Au, M.H., Mao, X.: Constant-size cca-secure multi-hop unidirectional proxy re-encryption from indistinguishability obfuscation. Theor. Comput. Sci. **847**, 1–16 (2020)
17. Lee, S., Park, H., Kim, J.: A secure and mutual-profitable drm interoperability scheme. In: ISCC 2010. pp. 75–80. IEEE (2010)

18. Lee, Y., Micciancio, D., Kim, A., Choi, R., Deryabin, M., Eom, J., Yoo, D.: Efficient fhew bootstrapping with small evaluation keys, and applications to threshold homomorphic encryption. In: Hazay, C., Stam, M. (eds.) EUROCRYPT 2023, Part III. LNCS, vol. 14006, pp. 227–256. Springer, Cham (2023)
19. Li, J., Qiao, Z., Zhang, K., Cui, C.: A lattice-based homomorphic proxy re-encryption scheme with strong anti-collusion for cloud computing. Sensors **21**(1), 288 (2021)
20. Li, Z., Ma, C., Wang, D.: Towards multi-hop homomorphic identity-based proxy re-encryption via branching program. IEEE Access **5**, 16214–16228 (2017)
21. Ma, C., Li, J., Ouyang, W.: A homomorphic proxy re-encryption from lattices. In: Chen, L., Han, J. (eds.) ProvSec 2016. LNCS, vol. 10005, pp. 353–372. Springer, Cham (2016)
22. Miao, P., Patranabis, S., Watson, G.: Unidirectional updatable encryption and proxy re-encryption from ddh. In: Boldyreva, A., Kolesnikov, V. (eds.) PKC 2023, Part II. LNCS, vol. 13941, pp. 368–398. Springer, Cham (2023)
23. Micciancio, D., Peikert, C.: Trapdoors for lattices: Simpler, tighter, faster, smaller. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 700–718. Springer, Heidelberg (2012)
24. Micciancio, D., Polyakov, Y.: Bootstrapping in fhew-like cryptosystems. In: WAHC 2021. pp. 17–28. ACM (2021)
25. Polyakov, Y., Rohloff, K., Sahu, G., Vaikuntanathan, V.: Fast proxy re-encryption for publish/subscribe systems. ACM Trans. Priv. Secur. **20**(4), 1–31 (2017)
26. Regev, O.: On lattices, learning with errors, random linear codes, and cryptography. In: Gabow, H.N., Fagin, R. (eds.) STOC 2005. pp. 84–93. ACM (2005)
27. Wang, L., Aono, Y., Nguyen, M.H., Boyen, X., et al.: Proxy re-encryption schemes with key privacy from lwe. Cryptology ePrint Archive (2016)
28. Zhao, F., Wang, H., Weng, J.: Constant-size unbounded multi-hop fully homomorphic proxy re-encryption from lattices. In: Garcia-Alfaro, J., Kozik, R., Choraś, M., Katsikas, S. (eds.) ESORICS 2024, Part III. LNCS, vol. 14984, pp. 238–258. Springer, Cham (2024)
29. Zhou, Y., Liu, S., Han, S.: Multi-hop fine-grained proxy re-encryption. In: Tang, Q., Teague, V. (eds.) PKC 2024, Part IV. LNCS, vol. 14604, pp. 161–192. Springer, Cham (2024)

Appendix

A Additional Preliminaries

A.1 Basic Lattice-based Encryption

For positive integers Q and a power of two N, the basic RLWE encryption of $m \in \mathcal{R}$ under the secret key z is defined as $\mathrm{RLWE}_z = (a, -a \cdot z + m + e) \in \mathcal{R}_Q^2$. For a base integer B_g and a degree $d_g \geq 1$, the gadget vector is defined by $\mathbf{g} = (g_0, \cdots, g_{d_g-1})$. A gadget decomposition of $t \in \mathcal{R}_Q$ is defined as $(t_0, \cdots, t_{d_g-1})$ if $t = \sum_{i=0}^{d_g-1} g_i \cdot t_i$ where $\|t_i\|_\infty < B_g$ for $i \in [0, d_g - 1]$. For a gadget vector $\mathbf{g}$, we define $\mathrm{RLWE}'_z(m)$ and $\mathrm{RGSW}_z(m)$ as follows [24]:

$$\mathrm{RLWE}'_z(m) \coloneqq (\mathrm{RLWE}_z(g_0 \cdot m), \mathrm{RLWE}_z(g_1 \cdot m), \cdots, \mathrm{RLWE}_z(g_{d_g-1} \cdot m)) \in \mathcal{R}_Q^{2d_g},$$
$$\mathrm{RGSW}_z(m) \coloneqq (\mathrm{RLWE}'_z(z \cdot m), \mathrm{RLWE}'_z(m)) \in \mathcal{R}_Q^{2 \times 2d_g}.$$

The scalar multiplication " $\odot$ " between a polynomial in $\mathcal{R}_Q$ and RLWE' ciphertext is defined as

$$\begin{aligned} t \odot \mathrm{RLWE}'_z(m) &= \langle (t_0, \cdots, t_{d_g-1}), (\mathrm{RLWE}_z(g_0 \cdot m), \cdots, \mathrm{RLWE}_z(g_{d_g-1} \cdot m)) \rangle \\ &= \sum_{i=0}^{d_g-1} t_i \cdot \mathrm{RLWE}_z(g_i \cdot m) = \mathrm{RLWE}_z(t \cdot m) \in \mathcal{R}_Q^2. \end{aligned}$$

The error introduced by " $\odot$ " is $e_\odot = \sum_{i=0}^{d_g-1} t_i \cdot e_i$, where e_i is the error term in $\mathrm{RLWE}_z(g_i \cdot m)$.

The multiplication " $\circledast$ " between RLWE ciphertext and RGSW ciphertext is defined as

$$\begin{aligned} \mathrm{RLWE}_z(m_1) \circledast \mathrm{RGSW}_z(m_2) &= (a, b) \circledast (\mathrm{RLWE}'_z(z \cdot m_2), \mathrm{RLWE}'_z(m_2)) \\ &= a \odot \mathrm{RLWE}'_z(z \cdot m_2) + b \odot \mathrm{RLWE}'_z(m_2) \\ &= \mathrm{RLWE}_z(m_1 \cdot m_2 + e_1 \cdot m_2) \\ &\approx \mathrm{RLWE}_z(m_1 \cdot m_2) \in \mathcal{R}_Q^2. \end{aligned}$$

The error term $e_1 \cdot m_2$ will be sufficiently small if using $m_2 = \pm X^v$ as messages, then the error term after " $\circledast$ " is $2e_\odot + e_1$, where e_1 is the error variance of the input RLWE ciphertext.

A.2 The LMKC+ Blind Rotation Algorithm

The LMKC+ blind rotation algorithm [18] utilizes the automorphism algorithm, which can apply the automorphism $\psi : \mathcal{R} \to \mathcal{R}$ given by $a(X) \mapsto a(X^t)$ for $t \in \mathbb{Z}_{2N}^*$ and transform an RLWE ciphertext $\mathrm{RLWE}_z(m(X))$ into the RLWE ciphertext $\mathrm{RLWE}_z(m(X^t))$.

– $\mathsf{EvalAuto}_t(\mathrm{RLWE}_z(m), \mathbf{ak}_t)$: Given a ciphertext $\mathrm{RLWE}_z(m) = (a(X), b(X))$ and an automorphism key $\mathbf{ak}_t = \mathrm{RLWE}'_z(z(X^t))$, it first applies ψ_t to $\mathrm{RLWE}_z(m)$ to obtain a ciphertext $\mathrm{RLWE}_{z(X^t)}(m(X^t)) = (a(X^t), b(X^t))$, and then, it outputs $\mathrm{RLWE}_z(m(X^t)) = a(X^t) \odot \mathbf{ak}_t + (0, b(X^t)) \pmod{Q}$.

We give a detailed LMKC+ blind rotation algorithm in Alg. 1.

Algorithm 1 LMKC+ Blind Rotation with odd Input: $\mathsf{BlindRotate}$ [18]

Input: A test polynomial $\mathsf{testP} \in \mathcal{R}_Q$, an all-odd LWE ciphertext $(\mathbf{a}, b) \in \mathbb{Z}_{2N}^n \times \mathbb{Z}_{2N}$ under the secret $\mathbf{s}$, and blind rotation keys $\mathbf{brk} = \{\mathbf{ak}_{-g}, \{\mathbf{ak}_{g^u}\}_{u\in[w]}, \{\mathbf{brk}_j = \mathrm{RGSW}_z(X^{s_j})\}_{j\in[0,n-1]}\}$.

Output: An RLWE ciphertext $\mathrm{RLWE}_z(\mathsf{testP} \cdot X^{b+\langle \mathbf{a},\mathbf{s}\rangle})$.

$\mathbf{acc} \leftarrow (0, \mathsf{testP}(X^{-g}) \cdot X^{-gb}))$
$v \leftarrow 0$
for $(l = N/2 - 1; l > 0; l = l - 1)$ **do**
  for $i \in I_l^-$ **do**
    $\mathbf{acc} \leftarrow \mathbf{acc} \circledast \mathbf{brk}_i$
  $v \leftarrow v + 1$
  if $(I_{l-1}^- \neq \emptyset$ or $v = w$ or $l = 1)$ **then**
    $\mathbf{acc} \leftarrow \mathsf{EvalAuto}_{g^v}(\mathbf{acc}, \mathbf{ak}_{g^v})$
    $v \leftarrow 0$
for $i \in I_0^-$ **do**
  $\mathbf{acc} \leftarrow \mathbf{acc} \circledast \mathbf{brk}_i$
$\mathbf{acc} \leftarrow \mathsf{EvalAuto}_{-g}(\mathbf{acc}, \mathbf{ak}_{-g})$
for $(l = N/2 - 1; l > 0; l = l - 1)$ **do**
  for $i \in I_l^+$ **do**
    $\mathbf{acc} \leftarrow \mathbf{acc} \circledast \mathbf{brk}_i$
  $v \leftarrow v + 1$
  if $(I_{l-1}^+ \neq \emptyset$ or $v = w$ or $l = 1)$ **then**
    $\mathbf{acc} \leftarrow \mathsf{EvalAuto}_{g^v}(\mathbf{acc}, \mathbf{ak}_{g^v})$
    $v \leftarrow 0$
for $i \in I_0^+$ **do**
  $\mathbf{acc} \leftarrow \mathbf{acc} \circledast \mathbf{brk}_i$
return: $\mathbf{acc} = \mathrm{RLWE}_z(\mathsf{testP} \cdot X^{b+\langle \mathbf{a},\mathbf{s}\rangle})$

A.3 Pebbling Game

The reversible pebbling game on DAG (Directed Acyclic Graph) was originally introduced in [4] to model reversible computation. Below we recall a variant proposed in [15] for its application to PRE.

Definition 5 (Pebbling Game [15]). *A reversible pebbling of a directed acyclic graph $G = (\mathcal{V}, \mathcal{E})$ with a unique source vertex i^* is a sequence $\mathcal{P} := (\mathcal{P}_0, \cdots, \mathcal{P}_\tau)$ of pebbling configurations $\mathcal{P}_t \subseteq \mathcal{V}$ with $t \in [0, \tau]$. Two subsequent configurations differ only in one vertex and the following rule is respected in a move: a pebble*

can be placed on or removed from a vertex if and only if all its children carry a pebble. That is, $\mathcal{P}$ is a valid sequence iff

$$\forall t \in [\tau], \exists! k \in \mathcal{P}_{t-1} \triangle \mathcal{P}_t \text{ and } \mathsf{children}(k, G) \subseteq \mathcal{P}_{t-1}.$$

Starting with an empty graph (i.e., $\mathcal{P}_0 = \emptyset$), the goal of the game is to place a pebble on the source (i.e., $i^* \in \mathcal{P}_\tau$).

For a DAG G, let $\mathcal{P}_G$ denote the set of all valid reversible pebbling sequences for G. The time complexity of a particular sequence $\mathcal{P} = (\mathcal{P}_0, \cdots, \mathcal{P}_\tau)$ for a DAG G is defined as $\tau_G(\mathcal{P}) := \tau$, and its space complexity is defined as $\sigma_G(\mathcal{P}) := \max_{t \in [0,\tau]} |\mathcal{P}_t|$.

For a class of DAGs $\mathcal{G}$, it has time complexity τ and space complexity σ if for all $G \in \mathcal{G}$, there exists a sequence $\mathcal{P} \in \mathcal{P}_G$ such that $\tau_G(\mathcal{P}) \leq \tau$ and $\sigma_G(\mathcal{P}) \leq \sigma$.

B Omitted Proofs

Note that the circular security assumption guarantees the security with the usage of blind rotation keys, as in [28], and we will no longer consider blind rotation keys in the following proofs for simplicity.

B.1 Proof of Theorem 2

Proof (Proof of Theorem 2). We prove the theorem via games G_0, G_1 and G_2.

Game G_0: This is the IND experiment (cf. Fig. 5). Let Win denote the event that $\beta' = \beta$. By definition, $\mathsf{Adv}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}(n) = |\mathsf{Pr}_0[\mathsf{Win}] - \frac{1}{2}|$. [6]

In this game, the challenger runs the KeyGen algorithm to generate (pk, sk), where $pk = \widetilde{\mathbf{pk}} = (\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$ for $\mathbf{b} = -\mathbf{A}^\top \mathbf{s} + \mathbf{e}$ and $sk = (\mathbf{s}, \bar{\mathbf{z}})$. Then the challenger chooses a random bit $\beta \xleftarrow{\$} \{0,1\}$ and answers $\mathcal{A}$'s $\mathcal{O}_{\mathsf{CHAL}}$ queries (m_0, m_1) with $ct \leftarrow \mathsf{Enc}(pk, m_\beta)$, i.e. $ct = (\mathbf{a}, b) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where $\mathbf{a} = \mathbf{A}\mathbf{r} + \mathbf{e}_1 \in \mathbb{Z}_q^n$ and $b = \mathbf{b}^\top \mathbf{r} + e_2 + \lfloor \frac{q}{4} \rceil \cdot m_\beta \in \mathbb{Z}_q$.

Game G_1: It's the same as G_0, except that the public key $pk = (\mathbf{A}, \mathbf{b})$ is sampled from the uniform distribution over $\mathbb{Z}_q^{n \times n} \times \mathbb{Z}_q^n$. Note that the public key pk in G_0 is $(\mathbf{A}, \mathbf{b})$, where $\mathbf{b} = -\mathbf{A}^\top \mathbf{s} + \mathbf{e}$. Under the $\mathsf{LWE}^n_{n,q,\hat{\chi}}$ assumption, we get that $|\mathsf{Pr}_1[\mathsf{Win}] - \mathsf{Pr}_0[\mathsf{Win}]| \leq \mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^n_{n,q,\hat{\chi}})$.

Game G_2: It's the same as G_1, except for the reply to $\mathcal{O}_{\mathsf{CHAL}}(m_0, m_1)$, now the challenger returns a uniformly sampled $ct \xleftarrow{\$} \mathbb{Z}_q^n \times \mathbb{Z}_q$ to $\mathcal{A}$, which is independent of the bit β. Therefore, $\mathsf{Pr}_2[\mathsf{Win}] = \frac{1}{2}$.

Now we construct a PPT adversary $\mathcal{B}''$ against the $\mathsf{LWE}^{n+1}_{n,q,\chi}$ assumption to show that $|\mathsf{Pr}_2[\mathsf{Win}] - \mathsf{Pr}_1[\mathsf{Win}]| \leq Q_{\mathsf{CHAL}} \cdot \mathsf{Adv}_{\mathcal{B}''}(\mathsf{LWE}^{n+1}_{n,q,\chi})$.

[6] In this paper, we use $\mathsf{Pr}_i[\mathsf{Win}]$ to denote the probability of a particular event occurring in game G_i.

Algorithm $\mathcal{B}''$: Given a challenge $(\bar{\mathbf{A}}, \mathbf{U})$, $\mathcal{B}''$ wants to distinguish $\mathbf{U} = \bar{\mathbf{A}}\mathbf{S} + \mathbf{E}$ from $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Q_{\mathsf{CHAL}}}$, where $\bar{\mathbf{A}} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times n}$, $\mathbf{S} \xleftarrow{\$} \chi^{n\times Q_{\mathsf{CHAL}}}$ and $\mathbf{E} \xleftarrow{\$} \chi^{(n+1)\times Q_{\mathsf{CHAL}}}$.

$\mathcal{B}''$ is constructed by simulating $\mathsf{G}_1/\mathsf{G}_2$ as follows. $\mathcal{B}''$ sets $pk = (\mathbf{A}, \mathbf{b}) \in \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^n$ where $(\mathbf{A}^\top, \mathbf{b})^\top = \bar{\mathbf{A}}$, and returns pk to $\mathcal{A}$. Then $\mathcal{B}''$ chooses a random bit $\beta \xleftarrow{\$} \{0,1\}$ and parses $\mathbf{U} = (\mathbf{u}_1|\cdots|\mathbf{u}_{Q_{\mathsf{CHAL}}})$ with $\mathbf{u}_i \in \mathbb{Z}_q^{n+1}$. On $\mathcal{A}$'s i-th $\mathcal{O}_{\mathsf{CHAL}}(m_0, m_1)$ query, $\mathcal{B}''$ computes $ct := \mathbf{u}_i + (\mathbf{0}, \lfloor \frac{q}{4} \rceil \cdot m_\beta)^\top$ and returns ct to $\mathcal{A}$. Finally, $\mathcal{A}$ returns a bit β' and $\mathcal{B}''$ outputs 1 to its challenger iff $\beta' = \beta$.

Now we analyze the advantage of $\mathcal{B}''$. In the case of $\mathbf{U} = \bar{\mathbf{A}}\mathbf{S} + \mathbf{E}$, we have $\mathbf{u}_i = \bar{\mathbf{A}}\mathbf{s}_i + \mathbf{e}_i$, where $\mathbf{s}_i \in \mathbb{Z}_q^n$ and $\mathbf{e}_i \in \mathbb{Z}_q^{n+1}$ are the i-th column vector of $\mathbf{S}$ and $\mathbf{E}$ respectively. Then the ciphertext $ct = \bar{\mathbf{A}}\mathbf{s}_i + \mathbf{e}_i + (\mathbf{0}, \lfloor \frac{q}{4} \rceil \cdot m_\beta)^\top$, which is the same as G_1. In the case of $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Q_{\mathsf{CHAL}}}$, we have $ct = \mathbf{u}_k + (\mathbf{0}, \lfloor \frac{q}{4} \rceil \cdot m_\beta)^\top$ for $\mathbf{u}_k \xleftarrow{\$} \mathbb{Z}_q^{(n+1)}$, which is the same as G_2. Consequently, $\mathcal{B}''$ simulates G_1 in the case of $\mathbf{U} = \bar{\mathbf{A}}\mathbf{S} + \mathbf{E}$ and simulates G_2 in the case of $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Q_{\mathsf{CHAL}}}$, then it's easy to see that $|\mathsf{Pr}_2[\mathsf{Win}] - \mathsf{Pr}_1[\mathsf{Win}]| \le Q_{\mathsf{CHAL}} \cdot \mathsf{Adv}_{\mathcal{B}''}(\mathsf{LWE}_{n,q,\chi}^{n+1})$.

Therefore, $\mathsf{Adv}^{\mathsf{IND}}_{\mathsf{mPRE},\mathcal{A}}(n) \le \mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^n_{n,q,\hat{\chi}}) + Q_{\mathsf{CHAL}} \cdot \mathsf{Adv}_{\mathcal{B}''}(\mathsf{LWE}^{n+1}_{n,q,\chi})$, and Theorem 2 follows. □

B.2 Proof of Theorem 3

Proof (Proof of Theorem 3). We prove the theorem via a sequence of Game $\mathsf{G}_0 - \mathsf{G}_{\mathfrak{n}}$, where G_0 is the wKP experiment, and in $\mathsf{G}_{\mathfrak{n}}$, $\mathcal{A}$ has a negligible advantage.

Game G_0: This is the wKP experiment (cf. Fig. 6). Let Win denote the event that $\beta' = \beta$. By definition, $\mathsf{Adv}^{\mathsf{wKP}}_{\mathsf{mPRE},\mathcal{A},\mathfrak{n}}(n) = |\mathsf{Pr}_0[\mathsf{Win}] - \frac{1}{2}|$.

Let $pk^{(i)} = \widetilde{\mathbf{pk}}^{(i)}$ and $sk^{(i)} = (\mathbf{s}_i, \bar{\mathbf{z}}_i)$ denote the public key and secret key of user $i \in [0, \mathfrak{n}]$, respectively. In this game, the adversary $\mathcal{A}$ is given $\{pk^{(i)}\}_{i\in[0,\mathfrak{n}]}$, the challenger chooses a random bit $\beta \xleftarrow{\$} \{0,1\}$ and answers $\mathcal{A}$'s $\mathcal{O}_{\mathsf{ReKey}}$ queries ($j \in [\mathfrak{n}]$) as follows: If there exists $(j, \cdot)$ in $\mathcal{Q}_{rk}$, then the challenger retrieves $(j, rk_{0\to j})$ and returns $rk_{0\to j}$ to $\mathcal{A}$. Otherwise,

- If $\beta = 0$, the challenger invokes $rk_{0\to j} \leftarrow \mathsf{ReKeyGen}(sk^{(0)}, pk^{(j)})$ and returns $rk_{0\to j}$ to $\mathcal{A}$. More precisely, it samples $\mathbf{R}_{0,j}, \mathbf{E}'_{0,j} \xleftarrow{\$} \hat{\chi}^{n\times Nl}, \mathbf{e}'_{0,j} \xleftarrow{\$} \hat{\chi}^{Nl}$ and returns
$$rk_{0\to j} = \begin{bmatrix} \mathbf{A}_j\mathbf{R}_{0,j} + \mathbf{E}'_{0,j} & \mathbf{0}_{n\times 1} \\ \mathbf{b}_j^\top\mathbf{R}_{0,j} + \mathbf{e}'^\top_{0,j} + \mathsf{Powersof2}(\bar{\mathbf{z}}_0)^\top & 1 \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times(Nl+1)}.$$
- If $\beta = 1$, the challenger invokes $rk_{0\to j} \leftarrow \mathsf{ReKeyGen}^*(pk^{(j)})$ which is defined as
$$\mathsf{ReKeyGen}^* : \mathbf{R} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Nl} \text{ and } rk_{0\to j} = \left[\mathbf{R} \;\middle|\; \begin{matrix}\mathbf{0}_{n\times 1} \\ 1\end{matrix}\right] \in \mathbb{Z}_q^{(n+1)\times(Nl+1)}.$$

Then the challenger records $(j, rk_{0\to j})$ in $\mathcal{Q}_{rk}$ and returns $rk_{0\to j}$ to $\mathcal{A}$.

Game $\mathsf{G}_t, t \in [\mathfrak{n}]$: It's the same as G_0, except for the generation of public keys $\{pk^{(i)}\}_{i\in[t]}$ and the reply to $\mathcal{A}$'s $\mathcal{O}_{\mathsf{ReKey}}(j)$ query when $\beta = 0$,

1) For all $i \in [t]$, the challenger now generates the public key $pk^{(i)} = (\mathbf{A}_i, \mathbf{b}_i)$ by uniformly sampling $\mathbf{A}_i \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$ and $\mathbf{b}_i \xleftarrow{\$} \mathbb{Z}_q^n$.
2) The challenger answers $\mathcal{A}$'s $\mathcal{O}_{\mathsf{ReKey}}(j)$ query when $\beta = 0$ as follows:
   - For $j \le t$, the challenger uniformly samples $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Nl}$ to get the re-encryption key $rk_{0\to j} = \left[\mathbf{R} \middle| \begin{smallmatrix}\mathbf{0}_{n\times 1}\\ 1\end{smallmatrix}\right] \in \mathbb{Z}_q^{(n+1)\times(Nl+1)}$.
   - For $j > t$, the challenger answers the query just like G_0.

Now we show that for $t \in [\mathfrak{n}]$, G_{t-1} and G_t are computationally indistinguishable under the LWE assumption.

Lemma 3. $|\mathsf{Pr}_{t-1}[\mathsf{Win}] - \mathsf{Pr}_t[\mathsf{Win}]| \le \mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^n_{n,q,\hat{\chi}}) + Nl \cdot \mathsf{Adv}_{\mathcal{B}'}(\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}})$ *for all* $t \in [\mathfrak{n}]$.

Proof. We prove the lemma by defining a new experiment $\mathsf{G}_{t-1,1}$.

Game $\mathsf{G}_{t-1,1}$*: It's the same as* G_{t-1}*, except that the public key* $pk^{(t)}$ *is changed into* $(\mathbf{A}_t, \mathbf{b}_t)$ *where* $\mathbf{A}_t \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$ *and* $\mathbf{b}_t \xleftarrow{\$} \mathbb{Z}_q^n$.

Firstly, we show that G_{t-1} *and* $\mathsf{G}_{t-1,1}$ *is computationally indistinguishable under the LWE assumption. Note that the only difference between* G_{t-1} *and* $\mathsf{G}_{t-1,1}$ *is the generation of the public key* $pk^{(t)}$ *of user* t*. In* G_{t-1}, $pk^{(t)} = (\mathbf{A}_t, \mathbf{b}_t)$ *where* $\mathbf{A}_t \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$ *and* $\mathbf{b}_t = -\mathbf{A}_t^\top \mathbf{s}_t + \mathbf{e}_t \in \mathbb{Z}_q^n$ *for* $\mathbf{s}_t, \mathbf{e}_t \in \hat{\chi}^n$*. In* $\mathsf{G}_{t-1,1}$, $(\mathbf{A}_t, \mathbf{b}_t)$ *is uniformly distributed over* $\mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^n$*. Therefore, there exists a* PPT *adversary* $\mathcal{B}$ *against the* $\mathsf{LWE}^n_{n,q,\hat{\chi}}$ *assumption such that*

$$|\mathsf{Pr}_{t-1,1}[\mathsf{Win}] - \mathsf{Pr}_{t-1}[\mathsf{Win}]| \le \mathsf{Adv}_{\mathcal{B}}(\mathsf{LWE}^n_{n,q,\hat{\chi}}). \tag{3}$$

Then we construct a PPT *adversary* $\mathcal{B}'$ *against the* Nl-$\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}}$ *assumption, such that* $|\mathsf{Pr}_{t-1,1}[\mathsf{Win}] - \mathsf{Pr}_t[\mathsf{Win}]| \le Nl \cdot \mathsf{Adv}_{\mathcal{B}'}(\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}})$.

Algorithm $\mathcal{B}'$*. Given a challenge* $(\mathbf{A}, \mathbf{U})$*,* $\mathcal{B}'$ *wants to distinguish* $\mathbf{U} = \mathbf{AS} + \mathbf{E}$ *from* $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Nl}$*, where* $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times n}$*,* $\mathbf{S} \xleftarrow{\$} \hat{\chi}^{n\times Nl}$ *and* $\mathbf{E} \xleftarrow{\$} \hat{\chi}^{(n+1)\times Nl}$.

$\mathcal{B}'$ *is constructed by simulating* $\mathsf{G}_{t-1,1}/\mathsf{G}_t$ *for* $\mathcal{A}$ *as follows. Firstly,* $\mathcal{B}'$ *sets* $pk^{(t)} = (\mathbf{A}_t, \mathbf{b}_t) \in \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^n$ *where* $(\mathbf{A}_t^\top, \mathbf{b}_t)^\top = \mathbf{A}$ *for the user* t*. For the users* $i \in [t-1]$*,* $\mathcal{B}'$ *sets* $pk^{(i)} = (\mathbf{A}_i, \mathbf{b}_i)$ *where* $(\mathbf{A}_i, \mathbf{b}_i) \xleftarrow{\$} \mathbb{Z}_q^{n\times n} \times \mathbb{Z}_q^n$*, and honestly generates secret key* $sk^{(i)}$*. Then* $\mathcal{B}'$ *invokes* KeyGen *algorithm to honestly generate* $(pk^{(i)}, sk^{(i)})$ *for users* $i \in [0, \mathfrak{n}]\backslash[t]$ *and sends* $\{pk^{(i)}\}_{i\in[0,\mathfrak{n}]}$ *to* $\mathcal{A}$*. After that,* $\mathcal{B}'$ *initializes an empty set* $\mathcal{Q}_{rk}$ *and chooses a random bit* $\beta \xleftarrow{\$} \{0,1\}$*. On receiving the* $\mathcal{O}_{\mathsf{ReKey}}(j \in [\mathfrak{n}])$ *query from* $\mathcal{A}$*, if there exists* $(j, \cdot)$ *in* $\mathcal{Q}_{rk}$*, then the challenger retrieves* $(j, rk_{0\to j})$ *and returns* $rk_{0\to j}$ *to* $\mathcal{A}$*. Else, if* $\beta = 1$*,* $\mathcal{B}'$ *invokes* $\mathsf{ReKeyGen}^*$ *to obtain* $rk_{0\to j}$ *and gives it to* $\mathcal{A}$*. Otherwise,* $\mathcal{B}'$ *answers the* $\mathcal{O}_{\mathsf{ReKey}}(j \in [\mathfrak{n}])$ *as follows:*

- *For* $j \le t-1$*,* $\mathcal{B}'$ *samples* $\mathbf{R} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Nl}$ *to get the re-encryption key* $rk_{0\to j} = \left[\boldsymbol{R} \middle| \begin{smallmatrix}\mathbf{0}_{n\times 1}\\ 1\end{smallmatrix}\right]$.
- *For* $j = t$*,* $\mathcal{B}'$ *returns re-encryption key:* $rk_{0\to j} = \left[\mathbf{U} + \begin{pmatrix}\mathbf{0}_{n\times 1}\\ \mathsf{Powersof2}(\bar{\mathbf{z}}_0)^\top\end{pmatrix} \middle| \begin{smallmatrix}\mathbf{0}_{n\times 1}\\ 1\end{smallmatrix}\right]$.

- *For $j > t$, $\mathcal{B}'$ samples $\mathbf{R}_{0,j}, \mathbf{E}'_{0,j} \xleftarrow{\$} \hat{\chi}^{n \times Nl}$ and $\mathbf{e}'_{0,j} \xleftarrow{\$} \hat{\chi}^{Nl}$ and returns the re-encryption key:*

$$rk_{0\to j} = \begin{bmatrix} \mathbf{A}_j \mathbf{R}_{0,j} + \mathbf{E}'_{0,j} & \mathbf{0}_{n\times 1} \\ \mathbf{b}_j^\top \mathbf{R}_{0,j} + \mathbf{e}'^\top_{0,j} + \mathsf{Powersof2}(\bar{\mathbf{z}}_0)^\top & 1 \end{bmatrix} \in \mathbb{Z}_q^{(n+1)\times(Nl+1)}.$$

Finally, $\mathcal{A}$ outputs a bit β', and $\mathcal{B}'$ outputs 1 to its challenger iff $\beta' = \beta$.

Now we analyze the advantage of $\mathcal{B}'$. In case of $\mathbf{U} = \mathbf{AS} + \mathbf{E}$, $\mathcal{B}''$ perfectly simulates $\mathsf{G}_{t-1,1}$. In case of $\mathbf{U} \xleftarrow{\$} \mathbb{Z}_q^{(n+1)\times Nl}$, $\mathbf{U} + \begin{pmatrix} \mathbf{0}_{n\times 1} \\ \mathsf{Powersof2}(\bar{\mathbf{z}}_0)^\top \end{pmatrix}$ is also uniformly distributed over $\mathbb{Z}_q^{(n+1)\times Nl}$, then $\mathcal{B}'$ perfectly simulates G_t. Therefore,

$$|\mathsf{Pr}_t[\mathsf{Win}] - \mathsf{Pr}_{t-1,1}[\mathsf{Win}]| \le \mathsf{Adv}_{\mathcal{B}'}(Nl\text{-}\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}}) \le Nl \cdot \mathsf{Adv}_{\mathcal{B}'}(\mathsf{LWE}^{n+1}_{n,q,\hat{\chi}}). \quad (4)$$

Combining (3) and (4), we complete the proof of Lemma 3. □

Finally, in $\mathsf{G}_{\mathfrak{n}}$, the challenger's reply to $\mathcal{A}$'s $\mathcal{O}_{\mathsf{ReKey}}$ query in $\beta = 0$ is identical to that in the case of $\beta = 1$. Therefore we have $\mathsf{Pr}_{\mathfrak{n}}[\mathsf{Win}] = \frac{1}{2}$. Combining this with Lemma 3, we complete the proof of Theorem 3. □

B.3 Proof of Theorem 4

Proof (Proof of Theorem 4). In the SH experiment (cf. Fig. 7), let $sk^{(i)} = (\mathbf{s}_i, \bar{\mathbf{z}}_i)$ and $pk^{(i)} = (\mathbf{A}_i, \mathbf{b}_i)$ where $\mathbf{b}_i = -\mathbf{A}_i^\top \mathbf{s}_i + \mathbf{e}_i$ for the user $i \in \{0, 1\}$. The challenger invokes $rk_{0\to 1} \leftarrow \mathsf{ReKeyGen}(sk^{(0)}, pk^{(1)})$ and initializes $\mathcal{L} := \bot, \mathsf{ctr} := 0$. Then the challenger chooses a random bit $\beta \xleftarrow{\$} \{0, 1\}$ and answers $\mathcal{A}$'s $\mathcal{O}_{\mathsf{Enc}}$ and $\mathcal{O}_{\mathsf{CHAL}}$ queries as follows:

- On receiving the encryption query $\mathcal{O}_{\mathsf{Enc}}(m)$, the challenger increments ctr and runs $\mathsf{Enc}(pk^{(0)}, m)$ to generate the ciphertext $ct^{(0)}$. Then the challenger records $(\mathsf{ctr}, m, ct^{(0)})$ in $\mathcal{L}$ and returns $(\mathsf{ctr}, ct^{(0)})$ to $\mathcal{A}$.
- On receiving the challenge query $\mathcal{O}_{\mathsf{CHAL}}(k)$, the challenger first retrieves $(k, m, ct^{(0)})$ and returns $\bot$ if fails; otherwise, the challenger answers the query as follows:
  - If $\beta = 0$, the challenger runs $\mathsf{ReEnc}(pk^{(0)}, rk_{0\to 1}, ct^{(0)})$ to generate the re-encryption ciphertext $ct^{(1)}$, i.e.,

$$ct^{(1)} = \begin{bmatrix} \mathbf{A}_1[\mathbf{R}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{r}}_1] + \mathbf{E}'_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{e}}_{1,1} \\ \mathbf{b}_1^\top[\mathbf{R}_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + \bar{\mathbf{r}}_1] + \mathbf{e}'^\top_{0,1}\mathsf{BitDecomp}(\bar{\mathbf{a}}^{(0)}) + e_{sum} + \bar{e}_{1,2} + \frac{q}{4}m \end{bmatrix} \quad (5)$$

    where $\mathbf{R}_{0,1}, \mathbf{E}'_{0,1} \xleftarrow{\$} \hat{\chi}^{n\times Nl}, \mathbf{e}'_{0,1} \xleftarrow{\$} \hat{\chi}^{Nl}, \bar{\mathbf{r}}_1, \bar{\mathbf{e}}_{1,1} \xleftarrow{\$} \chi^n$ and $\bar{e}_{1,2} \xleftarrow{\$} \chi$.
  - If $\beta = 1$, the challenger runs $\mathsf{Enc}(pk^{(1)}, m)$ to generate the fresh ciphertext $ct^{(1)}$, i.e.,

$$ct^{(1)} = \begin{bmatrix} \mathbf{A}_1\mathbf{r}_1 + \mathbf{e}_{1,1} \\ \mathbf{b}_1^\top \mathbf{r}_1 + e_{1,2} + \lfloor \frac{q}{4} \rceil \cdot m \end{bmatrix}, \quad (6)$$

    where $\mathbf{r}_1, \mathbf{e}_{1,1} \xleftarrow{\$} \chi^n$ and $e_{1,2} \xleftarrow{\$} \chi$.

By the smudging lemma (cf. Lemma 1), as long as $[\frac{q}{Q}(3n+\frac{N-n}{w})d_gB_gN\tilde{r}\log n + \frac{1+N\hat{r}\log n}{2} + Nl\hat{r}\log n + \frac{1}{2}]/(r\log n) = \mathsf{negl}(n)$, the re-encryption ciphertext $ct^{(1)}$ in (5) and the fresh encryption $ct^{(1)}$ in (6) are statistically indistinguishable. Combining with the fact that the number of $\mathcal{O}_{\mathsf{CHAL}}$ queries from $\mathcal{A}$ is at most polynomial, $\mathcal{A}$ has a negligible advantage in distinguishing $\beta = 0$ and $\beta = 1$. Therefore, with our proper choice of parameters, $\mathsf{Adv}^{\mathsf{SH}}_{\mathsf{mPRE},\mathcal{A}}(n) \leq \mathsf{negl}(n)$, and Theorem 4 follows. □

C The Correctness of Our Homomorphic PRE

For ciphertext $ct_j^{(i)} = (\mathbf{a}_j^{(i)}, b_j^{(i)})$ satisfying that $b_j^{(i)} + \langle \mathbf{a}_j^{(i)}, \mathbf{s}_i \rangle = \frac{q}{4} \cdot m_j + e_j^{(i)}$ for $j \in [2]$, we will show the decryption of $ct^{(i)}$ results in $m_1 \barwedge m_2$.

More precisely, the re-encryption algorithm first computes an all-odd LWE ciphertext $\check{ct}^{(i)} = (\check{\mathbf{a}}^{(i)}, \check{b}^{(i)})$ with modulus $2N$ satisfying that

$$\begin{aligned}\check{b}^{(i)} + \langle \check{\mathbf{a}}^{(i)}, \mathbf{s}_i \rangle &= \lfloor \frac{2N}{q}(b_1^{(i)} + b_2^{(i)}) \rceil_{\mathsf{odd}} + \langle \lfloor \frac{2N}{q}(\mathbf{a}_1^{(i)} + \mathbf{a}_2^{(i)}) \rceil_{\mathsf{odd}}, \mathbf{s}_i \rangle \\ &= \frac{N}{2}(m_1 + m_2) + \frac{2N}{q}(e_1^{(i)} + e_2^{(i)}) + e_{ms1}^{(i)},\end{aligned}$$

where $e_{ms1}^{(i)}$ is the rounding error introduced by $\mathsf{ModSwitch}^{\mathsf{odd}}_{q,2N}$. After the blind rotation algorithm, we have

$$\tilde{c}^{(i)} = (\tilde{a}^{(i)}, -\tilde{a}^{(i)} \cdot z_i + e_{br}^{(i)} + \mathsf{testP}' \cdot X^{\frac{N}{2}(m_1+m_2) + \frac{2N}{q}(e_1^{(i)} + e_2^{(i)}) + e_{ms1}^{(i)}}) \in \mathcal{R}_Q^2.$$

As long as $|\frac{2N}{q}(e_1^{(i)} + e_2^{(i)}) + e_{ms1}^{(i)}| < \frac{N}{4}$, the constant term of the underlying message of $\tilde{c}^{(i)}$ is equal to $-\frac{Q}{8}$ if $m_1 = m_2 = 1$; otherwise, it's equal to $\frac{Q}{8}$. According to the sample extraction algorithm, the LWE ciphertext $\hat{ct}^{(i)} = (\hat{\mathbf{a}}^{(i)}, \hat{b}^{(i)}) \leftarrow \mathsf{SampleExtract}(\tilde{c}^{(i)} + (0, \frac{Q}{8}))$ satisfies that

$$\hat{b}^{(i)} + \langle \hat{\mathbf{a}}^{(i)}, \bar{\mathbf{z}}_i \rangle = \frac{Q}{4} \cdot (m_1 \barwedge m_2) + e_{br,0}^{(i)},$$

where $e_{br,0}^{(i)}$ is the constant term of blind rotation error $e_{br}^{(i)}$.

After key switching and modulus switching algorithms, we finally obtain a ciphertext $ct^{(i)} = (\mathbf{a}^{(i)}, b^{(i)}) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ satisfying that

$$b^{(i)} + \langle \mathbf{a}^{(i)}, \mathbf{s}_i \rangle = \frac{q}{4} \cdot (m_1 \barwedge m_2) + e^{(i)},$$

where $e^{(i)} = \frac{q}{Q} \cdot (e_{br,0}^{(i)} + e_{ks}^{(i)}) + e_{ms2}^{(i)}$, with $e_{ks}^{(i)}$ and $e_{ms2}^{(i)}$ being the error terms introduced by $\mathsf{KeySwitch}$ and $\mathsf{ModSwitch}_{Q,q}$ algorithms, respectively. As a result, the decryption algorithm recovers $m_1 \barwedge m_2$ as long as

$$|e^{(i)}| < \frac{q}{Q} \cdot \left[(3n + \frac{N-n}{w}) d_g B_g N \tilde{r} \log n + N d_{ks} \hat{r} \log n \right] + \frac{1 + n\hat{r}\log n}{2} < \frac{q}{8}.$$

Note that the input ciphertexts can be fresh, evaluated, or re-encrypted ciphertexts, it's evident to see that re-encrypted ciphertexts have the largest error terms. To support further re-encryptions or homomorphic evaluations, and as shown in (1), our unbounded multi-hop HPRE scheme is correct as long as $|\frac{2N}{q}(e_1 + e_2) + e_{ms1}| < \frac{N}{4}$ for two re-encrypted ciphertexts with errors e_1 and e_2, i.e.,

$$\frac{4N}{q} \cdot \Bigg[\frac{q}{Q} \cdot (3n + \frac{N-n}{w}) d_g B_g N \tilde{r} \log n + \frac{1 + N\hat{r} \log n}{2} + r \log n \\ + 2n\hat{r}r + Nl \cdot (2n \cdot \hat{r}^2 + \hat{r} \log n) \Bigg] + 1 + n\hat{r} \log n < \frac{N}{4}.$$