Relaxed Lattice-Based Programmable Hash Functions: New Efficient Adaptively Secure IBEs

Xingye Lu[1], Jingjing Fan[2], and Man Ho Au[1]

The Hong Kong Polytechnic University, Hong Kong
{xing-ye.lu,mhhau}@polyu.edu.hk
[2] The University of Hong Kong, Hong Kong
jjfan@cs.hku.hk

Abstract. In this paper, we introduce the notion of relaxed lattice-based programmable hash function (rPHF), which is a novel variant of lattice-based programmable hash functions (PHFs). Lattice-based PHFs, together with preimage trapdoor functions (TDFs), have been widely utilized (implicitly or explicitly) in the construction of adaptively secure identity-based encryption (IBE) schemes. The preimage length and the output length of the underlying PHF and TDF together determine the user secret key and ciphertext lengths of the IBE schemes. However, the current lattice-based PHF definition imposes the restriction that the preimage length of TDF in the IBE schemes cannot be too short, hindering the utilization of size-efficient NTRU TDF. To overcome this hurdle, rPHF relaxes the hash key distribution requirement in the definition of PHF from statistical indistinguishability to computational indistinguishability. This relaxation eliminates limitations on the preimage length of underlying TDFs in IBE, enabling the construction of IBEs from NTRU TDFs.

We introduce two instantiations of rPHF: the first produces a hash output length of 2 ring elements, with a hash key size linear to the input length, and the second yields an output length of 14 ring elements, with a hash key size proportional to the square root of the input length. Building upon these rPHF instantiations, we propose two adaptively secure lattice-based IBE schemes with ciphertext lengths of 5 and 17 ring elements and user secret key lengths of 4 and 16 ring elements, respectively. The length of the IBE master public key is roughly equivalent to the size of the hash key of the underlying rPHF. In comparison to existing IBE constructions, our proposed schemes achieve a significant reduction (over an order of magnitude) in ciphertext and secret key sizes. Notably, state-of-the-art constructions from ideal lattices exhibit secret key and ciphertext sizes over 100 ring elements, making our proposed schemes highly efficient. Moreover, the master public key sizes of our IBEs remain comparable.

Keywords: lattice-based cryptography, programmable hash function, identity-based encryption

1 Introduction

In 2008, Hofheinz and Kiltz introduced the concept of programmable hash function (PHF) [12] with the aim of capturing the partitioning technique essential for security proofs in the standard model. A PHF refers to a keyed hash function that exhibits two behaviours based on how hash keys are generated. Specifically, there are two distinct algorithms for generating these hash keys: the normal key generation algorithm and the trapdoor key generation algorithm. Both algorithms produce statistically indistinguishable hash keys. Hash keys generated using the trapdoor mode contain secret trapdoor information, allowing the secret partitioning of the hash input into two separate sets. In contrast, in the normal mode, hash keys have no such trapdoor information. In one set, we possess the knowledge to solve a challenging problem, while in the other set, such knowledge is absent.

The original PHF introduced in [12] and its subsequent works rely on "DL groups", which are vulnerable to attacks from quantum computers. To address this vulnerability, Zhang et al. [28] proposed the concept of lattice-based PHF. This notion can be seen as the instantiation of traditional PHF on lattices, a promising candidate for post-quantum cryptography. Lattice-based PHFs serve as fundamental components for adaptively secure lattice-based short signature and identity-based encryption (IBE) schemes.

1.1 Lattice-Based PHF and Adaptively Secure IBEs

In this section, we briefly review the constructions of adaptively secure IBEs employing lattice-based PHFs. For ease of explanation, we consider the construction from standard lattices. Let $\mathcal{H}_K$ be a lattice-based PHF, where K represents the hash key. $\mathcal{H}_K$ maps an input string to a matrix. The hash key K can be generated in two modes: the normal mode and the trapdoor mode.

In the normal mode, a PHF hash key K comprises matrices $\mathbf{H}_0, \mathbf{H}_1, ..., \mathbf{H}_k \in \mathbb{Z}_q^{n\times m}$ chosen uniformly at random, where k can be linear, square-root, or logarithmic with respect to the input string length ℓ, depending on the specific construction. The PHF will map a input $X \in \{0,1\}^\ell$ to a matrix $\mathbf{H}_\mathrm{X}$ such that $\mathcal{H}_K(\mathrm{X}) = \mathbf{H}_\mathrm{X} \in \mathbb{Z}_q^{n\times m}$. For concreteness, the lattice-based PHF defines $\mathcal{H}_K(X)$ as $\mathbf{H}_0 + \sum_i (-1)^{X[i]} \cdot \mathbf{H}_i$.

In contrast, the trapdoor mode generates hash key K' in the form of $\mathbf{H}_i = \mathbf{A} \cdot \mathbf{R}_i + \mathbf{S}_i \cdot \mathbf{G}$, where $\mathbf{A} \in \mathbb{Z}_q^{n\times m'}$ and $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ are pre-defined matrices, and $\{\mathbf{R}_i, \mathbf{S}_i\}$ are the trapdoor. The trapdoor information allows us to establish the relationship between the hash output of any input X through the equation: $\mathcal{H}_{K'}(X) = \mathbf{H}_\mathrm{X} = \mathbf{A}\cdot\mathbf{R}_\mathrm{X} + \mathbf{S}_\mathrm{X}\cdot\mathbf{G}$, where $\mathbf{R}_\mathrm{X} \in \mathbb{Z}_q^{m'\times m}$ and $\mathbf{S}_\mathrm{X} \in \mathbb{Z}_q^{n\times n}$ are two matrices that can be computed from the trapdoor.

For a lattice-based PHF, it partitions the hash inputs into two categories based on whether $\mathbf{S}_\mathrm{X} = 0$ or not. The category $\mathbf{S}_\mathrm{X} = 0$ allows the ability to use adversaries to solve specific hard problems in security proofs. Specifically, if a PHF is (u, v)-programmable, then given any inputs $X_1, \cdots, X_u$ and $Y_1, \cdots, Y_v$

such that $X_i \neq Y_j$ for all i, j, the probability:

$$\Pr[\mathbf{S}_{\mathrm{X}_1} = \cdots = \mathbf{S}_{\mathrm{X}_u} = 0 \ \wedge \ \mathbf{S}_{\mathrm{Y}_1} \neq 0, \cdots, \mathbf{S}_{\mathrm{Y}_v} \neq 0] \geq 1/\mathsf{poly}.$$

Lattice-based PHFs, when combined with "preimage sampleable" trapdoor functions (TDFs) [11], form the foundation for constructing lattice-based adaptively secure IBEs in the standard model. In such IBE schemes, each user id is associated with a matrix $[\mathbf{A}|\mathbf{H}_{\mathsf{id}}]$. Matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m'}$ is the public key for a lattice-based TDF $f_{\mathbf{A}}$. $\mathbf{A}$ is equipped with a trapdoor that facilitates the sampling of a short vector $\mathbf{t}$ such that $f_{\mathbf{A}}(\mathbf{t}) = \mathbf{A}\mathbf{t} = \mathbf{w} \mod q$ for any $\mathbf{w}$. $\mathbf{H}_{\mathsf{id}} \in \mathbb{Z}_q^{n \times m}$ represents the hash output of a rPHF $\mathcal{H}_K(\mathsf{id})$ with hash key K in normal mode. Extracting the secret key for a user id involves utilizing the trapdoor of $\mathbf{A}$ to generate a short vector $\mathsf{sk}_{\mathsf{id}} \in \mathbb{Z}_q^{m'+m}$ such that $[\mathbf{A}|\mathbf{H}_{\mathsf{id}}] \cdot \mathsf{sk}_{\mathsf{id}} = \mathbf{u} \mod q$, where $\mathbf{u}$ is another vector included in the master public key. To encrypt a message for id, one utilizes $[\mathbf{A}|\mathbf{H}_{\mathsf{id}}]$ and $\mathbf{u}$ as the public key of the Dual Regev PKE [11].

The main challenge of achieving adaptive security in the security proof is to make the simulator respond to key extraction queries without knowing $\mathbf{A}$'s trapdoor. To tackle this challenge, the hash key K in the security proof is generated in the trapdoor mode, incorporating with matrix $\mathbf{A}$ and another matrix $\mathbf{G}$. This $\mathbf{G}$ is the public key for TDF $f_{\mathbf{G}}$ and possesses its own trapdoor. When confronted with a key extraction query for a specific identity id, the simulator can relate $\mathbf{H}_{\mathsf{id}} = \mathcal{H}_K(\mathsf{id})$ to $\mathbf{A}$ and $\mathbf{G}$ in the form of $\mathbf{H}_{\mathsf{id}} = \mathbf{A} \cdot \mathbf{R}_{\mathsf{id}} + \mathbf{S}_{\mathsf{id}} \cdot \mathbf{G}$. When $\mathbf{S}_{\mathsf{id}} \neq 0$, the simulator can leverage the trapdoor of $\mathbf{G}$ to sample $\mathsf{sk}_{\mathsf{id}}$, effectively responding to the key extraction query.

The space complexity of the IBE scheme is intricately tied to the properties of both the underlying TDF and rPHF. Specifically, in the IBE scheme, the lengths of the secret key and ciphertext are determined by the dimensions of the lattice associated with $[\mathbf{A}|\mathbf{H}_{\mathsf{id}}]$ $(m' + m)$, where m' is the preimage length of $f_{\mathbf{A}}$, and m is the hash output length of rPHF. Thus, in the pursuit of designing size-efficient IBE schemes, the selection of TDF with short public keys is paramount. One of the most promising candidates is the NTRU TDF. NTRU stands out by offering a TDF with a preimage length of only 2 ring elements (equivalent to length $2n$), which is significantly shorter than its counterparts in standard and ideal lattices. By utilizing the NTRU TDF as the $\mathbf{A}$ and $\mathbf{G}$ components, it is hopeful to reduce the ciphertext to a few ring elements.

This idea, however, is difficult to realize. The security proof relies on the fact that the hash keys of rPHF generated from normal and trapdoor modes are statistically indistinguishable. Existing approaches utilize the leftover hash lemma to argue that $\mathbf{A} \cdot \mathbf{R}_i$ is close to uniform, thereby making the trapdoor mode hash key $\mathbf{H}_i = \mathbf{A} \cdot \mathbf{R}_i + \mathbf{S}_i \cdot \mathbf{G}$ also close to uniform. Unfortunately, utilizing leftover hash lemma has requirements on the dimension of $\mathbf{A} \in \mathbb{Z}_q^{n \times m'}$, and the entropy of $\mathbf{R}_i$, which renders the resulting IBE has secret key and ciphertext length of $O(n \log q)$, even though more size efficient NTRU TDF exists.

In light of this challenge, our paper aims to revisit the notion of lattice-based PHF and remove the dimension restriction. By addressing this limitation, we seek to pave the way for more efficient IBE schemes based on the proposed lattice-based rPHF.

1.2 Technical Overview

In our approach, we address the challenge inherent in the use of the leftover hash lemma. To overcome this, we first relax the requirement that the hash keys in the definition of $r\mathsf{PHF}$ in normal and trapdoor modes to be computationally close. This relaxaion gives rise to our $r\mathsf{PHF}$. We detail how to create $r\mathsf{PHF}$ from computational assumptions. For clarity, let's consider our $r\mathsf{PHF}$ from standard lattices (although in our actual construction, we employ the more efficient NTRU lattices).

Similar to lattice-based PHFs, hash keys for $r\mathsf{PHF}$s consist of a collection of matrices $\{\mathbf{H}_0, \mathbf{H}_1, \cdots, \mathbf{H}_\kappa\}$ sampled uniformly at random from $\mathbb{Z}_q^{n\times m}$. However, unlike original PHFs, in the trapdoor mode, each hash key matrix $\mathbf{H}_i$ is constructed as $\mathbf{H}_i = \mathbf{A}\cdot\mathbf{R}_i + \mathbf{E}_i + \mathbf{S}_i\cdot\mathbf{G}$, where matrices $\mathbf{A} \in \mathbb{Z}_q^{n\times n}$ and $\mathbf{G} \in \mathbb{Z}_q^{n\times m}$ are user specified, $\mathbf{R}_i, \mathbf{E}_i \in \mathbb{Z}_q^{n\times m}$ are sampled from small error distribution. By adopting this approach, we move from statistical arguments to computational ones. Specifically, we leverage the LWE (Ring LWE) assumptions to make the hash keys generated in normal mode to be computationally indistinguishable from those generated in trapdoor mode. Moreover, we achieve a reduction in the dimension of $\mathbf{A}$ from $n \times m'$ (where $m' = O(n \log q)$) to $n \times n$. In the trapdoor mode, each input X can be associated with the hash output $\mathcal{H}_K(X) = \mathbf{A}\cdot\mathbf{R}_{\mathrm{x}} + \mathbf{E}_{\mathrm{x}} + \mathbf{S}_{\mathrm{x}}\cdot\mathbf{G}$. The remaining part is to deal with the additional subtleties introduced from the error terms.

Replacing the lattice-based PHF used in the IBEs (e.g., [1, 28]) with our $r\mathsf{PHF}$ already allows for improvements in space complexity. The significance of our relaxation lies in the ability to remove the dimension limit. This key relaxation allows us to seamlessly combine the very efficient TDF from the NTRU lattice, and we will briefly outline it below.

NTRU lattice associated with modulus q and $\mathrm{h} \in \mathcal{R}_q$ is defined as $\Lambda_{\mathrm{h},q} = \{(\mathrm{u}, \mathrm{v}) \in \mathcal{R}_q^2 | \mathrm{u} + \mathrm{v} * \mathrm{h} = 0 \mod q\}$, where $\mathrm{h} = \mathrm{g}\cdot\mathrm{f}^{-1}$ and g, f are ring elements with small coefficients. With g, f, we are able to sample short $(\mathrm{s}_1, \mathrm{s}_2)$ such that $\mathrm{s}_1 + \mathrm{h}\cdot\mathrm{s}_2 = \mathrm{u} \mod q$ for a given u. By utilizing this NTRU TDF to instantiate our $r\mathsf{PHF}$, we can construct new IBE schemes featuring exceptionally short key and ciphertext sizes.

In this approach, each identity id corresponds to a vector $\mathbf{f}_{\mathsf{id}} = [1, \mathrm{h}, \mathbf{h}_{\mathsf{id}}]$ where $\mathbf{h}_{\mathsf{id}} \in \mathcal{R}_q^d$ is the $r\mathsf{PHF}$ hash output for id. In the IBE scheme, the NTRU trapdoor within h allows for the extraction of the secret key $\mathbf{s}$ satisfying $\mathbf{f}_{\mathsf{id}}\cdot\mathbf{s}^\intercal = \mathrm{u} \mod q$ for $\mathrm{u} \in \mathcal{R}_q$ in the master public key. With $\mathbf{f}_{\mathsf{id}}$ and $\mathrm{u} \in \mathcal{R}_q$ as the user public key, a message m can be encrypted. In the security proof, h will be sampled uniformly at random from its range. The simulator generates trapdoor mode hash key $\mathbf{h}_i = \mathrm{h}\cdot\mathbf{r}_i + \mathbf{e}_i + \mathrm{s}_i\cdot\mathbf{g}$, using this sampled h and a vector $\mathbf{g} \in \mathcal{R}_q^d$ equipped with a trapdoor. Each id is associated with a hash output $\mathbf{h}_{\mathsf{id}}$ in the form of $\mathbf{h}_{\mathsf{id}} = \mathrm{h}\cdot\mathbf{r}_{\mathsf{id}} + \mathbf{e}_{\mathsf{id}} + \mathrm{s}_{\mathsf{id}}\cdot\mathbf{g}$. When $\mathrm{s}_{\mathsf{id}} \neq 0$, we are able to construct an algorithm that extracts secret keys $\mathbf{s}$ for $[1, \mathrm{h}, \mathrm{h}\cdot\mathbf{r}_{\mathsf{id}} + \mathbf{e}_{\mathsf{id}} + \mathrm{s}_{\mathsf{id}}\cdot\mathbf{g}]$ with the trapdoor within $\mathbf{g}$. During the challenge phase, if $\mathrm{s}_i = 0$, the simulator leverages the adversary's response to solve the decisional RLWE problem.

1.3 Contributions

In this paper, our primary focus is on the development of adaptively secure lattice-based IBE with a core objective: enhancing their efficiency. Our goal centres on designing schemes with notably compact ciphertext sizes, aiming to reduce them to a minimal constant number of ring elements. Specifically, we obtain the following results.

- We first introduce and formalize the notion of the relaxed lattice-based programmable hash function ($r\mathsf{PHF}$). Based on this, we provide two instantiations of $r\mathsf{PHF}$.
  - Type-I-$r\mathsf{PHF}$ features a compact hash output comprising only 2 ring elements, with the hash key length being linear concerning the input size.
  - Type-II-$r\mathsf{PHF}$ is designed to balance hash key length and hash output size. Building upon the approach proposed in [14], this instantiation achieves a hash key length proportional to the square root of the input length. Consequently, the hash output size increases to a larger constant value in terms of ring elements.
- Based on our $r\mathsf{PHF}$, we propose novel adaptively secure lattice-based IBE schemes using the NTRU TDF.
  - Type-I Instantiation IBE: Based on the Type-I $r\mathsf{PHF}$, this scheme features a remarkably short ciphertext length of only 5 ring elements, with user secret key length of 4 ring elements and master public key length linear to the identity length. It sets a new standard for minimal ciphertext size among previously proposed adaptively secure IBE schemes.
  - Type-II Instantiation IBE: This scheme offers a ciphertext length of approximately 17 ring elements and a user secret key length of 16 ring elements. The length of the master public key is proportional to the square root of the identity length.

  A detailed comparison between our IBE schemes and prior works is presented in Table 1.
- Additionally, we propose a short signature scheme secure against universal forgery under chosen-message attack (UF-CMA) based on our $r\mathsf{PHF}$ in Section 5. The resulting two schemes have signature sizes of 4 ring elements and 14 ring elements, respectively. To the best of our knowledge, existing lattice signatures in the standard model typically have a shortest signature size of $O(\log q)$ ring elements ($O(n \log q)$ $\mathbb{Z}_q$ elements) [10, 2, 6, 18, 28].

1.4 Related Work

IBE, initially introduced by Shamir [22] in 1984, is a public key encryption scheme that allows the use of arbitrary strings as public keys, such as user's email address or telephone number. Boneh and Franklin [5] defined a security model

Table 1. Comparison of Lattice IBE in Standard Model[1]

Schemes	Ciphertext Length[2]	mpk Length	LWE param $1/\alpha$	Lattice
ABB10 [1]+Boy10 [6]	$O(n\log^2 q)$	$O(n^2\ell\cdot\log^2 q)$	$\tilde{O}(n^{5.5})$	Standard
KY16 [14]: Type I	$O(n\log^2 q)$	$O(n\ell^{\frac{1}{d}}\cdot\log^2 q)$	$\tilde{O}(n^{2d+1.5})$	Ideal
KY16 [14]: Type II[3]	$(32d+48)n\log q$	$O(\ell^{\frac{1}{d}}n\log q)$	$\tilde{O}(n^{2d+2.5})$	Ideal
	$113n\log q$	$113+112\ell^{\frac{1}{2}}n\log q$	$\tilde{O}(n^{6.5})$	Ideal
ZCZ16 [28]	$O(n\log^2 q)$	$O(n^2\log\ell\log^2 q)$	$\tilde{O}(Q^2\cdot n^{6.5})$	Standard
Yam17 ($\mathsf{F}_{\mathsf{MAH}}$) [27]	$O(n\log^2 q)$	$O(n^2\log^2\ell\log^2 q)$	$\tilde{O}(n^{11})$	Standard
JKN21 [13]	$O(n\log^2 q)$	$O(n^2\log\ell\log^2 q)$	$\tilde{O}(n^6)$	Standard
This paper (Type-I)	$5n\log q$	$(5+4\ell)n\log q$	$\tilde{O}(n^{3.75})$	NTRU
This paper (Type-II)	$17n\log q$	$(17+28\ell^{\frac{1}{2}})n\log q$	$\tilde{O}(n^{10.5})$	NTRU

[1] ℓ is the length of identity. Q is the total number of query in the security proof.
[2] User secret key length is $cl - n\log q$ where cl is the ciphertext length.
[3] The length given here is different from the original paper according to our analysis. The parameter given in the paper does not satisfy their security proof. Specifically, according to their proof, in [14] Type II, m should satisfy $m \geq 2\log_\rho q$. Thus, for $q = n^{2d+3}$, $\rho = n^{\frac{1}{4}}$, $m \geq 16d + 24$ should hold. The ciphertext length is $2m + 1$ ring elements. d is a flexible small constant equal to 2 or 3. The second row is the parameter when $d = 2$.

for identity-based encryption and proposed the first efficient and secure IBE based on Bilinear Diffie-Hellman (BDH) problem secure in the random oracle model. Since then, IBE has been a vibrant area of research in cryptography. In subsequent works, there have been pairing-based IBE schemes shown to be secure without random oracle, either in a weaker selective-ID model [7, 3] or in fully secure adaptive-ID model [4, 24, 25].

An important research direction of IBE is the design of schemes based on lattice-based cryptography, a promising candidate of post-quantum cryptography. Gentry et al. [11] first proposed a lattice-based IBE scheme in the random oracle model. Subsequently, lattice-based schemes secure in the standard model were introduced [8, 1, 26, 14, 27]. Cash et al. [8] leverage the admissible hash functions and assign a matrix to each bit of identity in their construction, resulting in a large user public key formed by concatenating these matrices. To enhance the efficiency, Agrawal et al. [1] processes identities as a chunk using a map function that maps an identity to a matrix, resulting in a lattice with dimension similar to IBE schemes in the random oracle model. To build such a map, it requires $O(\ell)$ basic matrices in the public parameters where ℓ is the identity length. Later, by altering the mapping approaches for identities, constructions with shorter public parameter sizes were proposed [26, 14, 28, 27, 13]. Despite these efforts, the typical ciphertext lengths of these adaptively secure IBEs are often bound by $O(\log q)$ ring elements or $O(n \log q)$ elements in $\mathbb{Z}_q$. In this paper, we aim to construct adaptively secure IBEs with a significantly reduced ciphertext length, restricted to just a small constant number of ring elements, using the NTRU lattice.

Current lattice-based IBEs from NTRU [9, 17] are both from the random oracle model. It is worth noting that building an adaptively secure NTRU IBE scheme from the standard model is not a mere adaptation of existing counterparts from standard lattice or ideal lattice to the NTRU lattice. This is due to the fact that the leftover hash lemma, which is employed in previous works to maintain the indistinguishability of IBE master public keys in the construction and security proofs, is not applicable to the NTRU preimage sampleable function.

2 Preliminaries

Elements in $\mathbb{Z}_q$ are represented by integers in $[-\frac{q}{2}, \frac{q}{2})$. For a ring $\mathcal{R}$ we define $\mathcal{R}_q$ to be the quotient ring $\mathbb{Z}_q[x]/(x^n + 1)$ with n being a power of 2 and q being a prime.

Vector and Matrix in $\mathcal{R}$: We use $\mathcal{R}_q^\times$ to denote the set of ring elements that have an inverse in $\mathcal{R}_q$. Row vectors in $\mathbb{Z}_q^m$ are denoted by italic lower-case bold letters (e.g. $\boldsymbol{x}$). Elements in $\mathcal{R}_q$ are denoted by lower-case letters (e.g. x). Row vectors with entries from $\mathcal{R}_q$ are denoted by lower-case bold letters (e.g. $\mathbf{x}$). Matrices are denoted by upper-case bold letters (e.g. $\mathbf{X}$). We denote $[\mathbf{x}]_j$ as the j-th entry of vector $\boldsymbol{x}$ and $[\mathbf{X}]_j$ as the j-th column of matrix $\mathbf{X}$. We use $\phi : \mathcal{R}_q^k \to \mathbb{Z}_q^{kn}$

to denote the transformation from a vector of ring elements to the vector of its coefficient embedding and $[\phi(\mathrm{a})]_j$ to denote the coefficient of term x^j in a. We use $[a,b]_{\mathcal{R},\kappa}$ to represent a polynomial in $\mathcal{R}$ with all of its coefficients in the interval $[a,b]$ and degree no larger than $\kappa-1$.

For two row vectors $\mathbf{v}$, $\mathbf{u}$ (two matrices $\mathbf{A}$, $\mathbf{B}$), we use $[\mathbf{v},\mathbf{u}]$, $[\mathbf{A}|\mathbf{B}]$ (resp. $[\mathbf{v};\mathbf{u}]$, $[\mathbf{A};\mathbf{B}]$) to denote the horizontal (resp. vertical) concatenation of them.

Norms: $\|\boldsymbol{v}\|_1$ is the ℓ_1 norm of vector $\boldsymbol{v}$ and $\|\boldsymbol{v}\|$ is the ℓ_2 norm of $\boldsymbol{v}$. For $\mathbf{v} = (\mathrm{v}_1, \cdots, \mathrm{v}_n)$, we define $\|\mathbf{v}\| = \sqrt{\sum_{i=1}^{n} \|\mathrm{v}_i\|^2}$. For a matrix $\mathbf{V}$, we define $s_1(\mathbf{V})$ as the spectral norm and $\|\tilde{\mathbf{V}}\|$ as the Gram-Schmidt norm.

Distributions: For distribution $\mathcal{D}$, $x \leftarrow \mathcal{D}$ means sampling x according to distribution $\mathcal{D}$. Given two distributions $\mathcal{D}_1$ and $\mathcal{D}_2$, $\mathcal{D}_1 \overset{c}{\approx} \mathcal{D}_2$ means that these two distributions are computationally indistinguishable. The continuous normal distribution over $\mathbb{R}^n$ centered at $\boldsymbol{v}$ with standard deviation σ is defined as $\rho_{\sigma,\boldsymbol{v}}(\boldsymbol{x}) = (\frac{1}{\sqrt{2\pi\sigma^2}}) e^{\frac{-\|x-v\|^2}{2\sigma^2}}$. For simplicity, when $\boldsymbol{v}$ is the zero vector, we use $\rho_\sigma(\boldsymbol{x})$. We also use $\rho_{\sigma,\boldsymbol{v}}(\mathrm{a})$ to denote the Gaussian distribution of $\mathrm{a} = \sum_{i=0}^{n-1} a_i X^i \in \mathcal{R}$ where the entry of coefficient vector $\phi(\mathrm{a}) \in \mathbb{R}^n$ is sampled from $\rho_{\sigma,\boldsymbol{v}}$.

The discrete normal distribution over $\mathbb{Z}^n$ centered at $\boldsymbol{v} \in \mathbb{Z}^n$ with standard deviation σ is defined as $\mathcal{D}_{\mathbb{Z}^n,\sigma,\boldsymbol{v}}(\boldsymbol{x}) = \frac{\rho_{\sigma,v}(\boldsymbol{x})}{\rho_{\sigma,v}(\mathbb{Z}^n)}$. We use $\mathcal{D}_{\mathcal{R},\sigma}$ to denote sampling the coefficient of a ring element as the distribution $\mathcal{D}_{\mathbb{Z}^n,\boldsymbol{s}}$.

Lemma 1. *Let b be a positive integer and $\mathbf{R} \in \mathcal{R}_q^{s\times t}$ be a matrix with coefficient independently sampled from $\mathcal{D}_\alpha$. Then, with overwhelming probability we have*

$$\Pr[s_1(\mathbf{R}) \le \alpha\sqrt{n} \cdot (\sqrt{s} + \sqrt{t} + \omega(\sqrt{\log n}))]$$

Let b be a positive integer and $\mathbf{R} \in \mathcal{R}_q^{s\times t}$ be a matrix chosen uniformly at random from $[-b,b]_{\mathcal{R}}^{s\times t}$. Then, there exists a universal constant $C(\approx 1/\sqrt{2\pi})$ such that

$$\Pr[s_1(\mathbf{R}) \le C\cdot\beta\sqrt{n} \cdot (\sqrt{s} + \sqrt{t} + \omega(\sqrt{\log n}))]$$

2.1 Lattices and Hardness Assumptions

A lattice in m-dimension Euclidean space $\mathbb{R}^m$ is a discrete set

$$\Lambda(\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n) = \left\{ \sum_{i=1}^{n} x_i \boldsymbol{b}_i | x_i \in \mathbb{Z} \right\}$$

of all integral combinations of n linear independent vectors $\boldsymbol{b}_1, \cdots, \boldsymbol{b}_n$ in $\mathbb{R}^m$ ($n \le m$). We call matrix $\mathbf{B} = [\boldsymbol{b}_1; \cdots; \boldsymbol{b}_n] \in \mathbb{R}^{n\times m}$ a basis of lattice Λ. Using matrix notation, a lattice can be defined as $\Lambda(\mathbf{B}) = \{\boldsymbol{x}\mathbf{B} | \boldsymbol{x} \in \mathbb{Z}^n\}$.

The discrete Gaussian distribution of a lattice Λ, parameter σ and center $\boldsymbol{v}$ is defined as $\mathcal{D}_{\Lambda,\sigma,\boldsymbol{v}}(\mathbf{x}) = \frac{\rho_{v,s}(\mathbf{x})}{\rho_{v,s}(\Lambda)}$.

Definition 1 *Let $m \geq n \geq 1$ and $q \geq 2$. For arbitrary matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ and vector $\boldsymbol{u} \in \mathbb{Z}_q^n$ define m-dimensional full-rank integer lattices and its shift:*

$$\Lambda^{\perp}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}_q^m : \mathbf{A}\mathbf{z}^{\intercal} = \mathbf{0} \mod q\},$$

$$\Lambda_{\boldsymbol{u}}^{\perp}(\mathbf{A}) = \{\mathbf{z} \in \mathbb{Z}_q^m : \mathbf{A}\mathbf{z}^{\intercal} = \boldsymbol{u} \mod q\}.$$

Lemma 2 ([19], Lemma 4.4). *For any n-dimensional lattice Λ, vector $\boldsymbol{c} \in \mathbb{R}^n$, and reals $0 < \epsilon < 1$, $s \geq \eta(n)$, we have*

$$\Pr_{x \sim \mathcal{D}_{\Lambda,s,c}}[\|\boldsymbol{x}\| > s\sqrt{n}] \leq \frac{1+\epsilon}{1-\epsilon} \cdot 2^{-n}.$$

Lemma 3 ([14], Lemma 20). *Let $\boldsymbol{e} \in \mathbb{Z}^n$ and let $\boldsymbol{x} \leftarrow \mathcal{D}_{\mathbb{Z}^n,\alpha q}$, for some $\alpha q > \omega(\sqrt{n})$. Then the quantity $|\boldsymbol{e} \cdot \boldsymbol{x}^{\intercal}|$ treated as an integer in $[0, q-1]$ satisfies $|\boldsymbol{e} \cdot \boldsymbol{x}^{\intercal}| \leq \|\boldsymbol{e}\| \alpha q \cdot \omega(\sqrt{\log n})$ with all but negligible probability in n.*

Learning With Error Problem Learning with error (LWE) problem was introduced by Regev [21] in 2005. It is a classic hard problem on lattices, that solving it is at least as hard as solving standard problems on lattices in the worst case. The Ring LWE (RLWE) problem [15] is a more compact variant of LWE and its provable hardness assumes the intractability of worst-case ideal lattice problems.

Definition 2 (Ring-LWE$_{n,q,\chi}$ $\mathsf{RLWE}_{n,q,\chi}$ Assumption) *Given ring-LWE distribution $A_{\boldsymbol{s},\chi}$ with a secret $s \in \mathcal{R}_q$, a_i uniform in $\mathcal{R}_q$, the "error polynomials" e_i sampled from the error distribution χ, for $\ell = poly(n)$, it holds that*

$$\{(a_i, a_i \cdot s + e_i)\}_{i \in [\ell]} \overset{c}{\approx} \{(a_i, u_i)\}_{i \in [\ell]},$$

where u_i is uniformly sampled from $\mathcal{R}_q$.

The search Ring-LWE assumption states that no PPT adversary can recover $\boldsymbol{s}$ from a polynomial number of samples $\{(\mathbf{a}_i, \mathbf{a}_i \cdot \boldsymbol{s} + e_i)\}_{i \in [\ell]}$ with a non-negligible probability.

Definition 3 (Average-Case Decision Ring-LWE $\mathsf{dRLWE}_{n,q,\chi}$) *Given m independent samples $(a_i, u_i) \in \mathcal{R}_q \times \mathcal{R}_q$, where every sample is distributed according to either:(i) $A_{s,\chi}$ for a uniformly random $\boldsymbol{s} \in \mathcal{R}_q$ (fixed for all samples), or (ii) the uniformly random, distinguish which is the case (with non-negligible advantage).*

Theorem 1 ([16], Theorem 2.22) *Let $\alpha = \alpha(n) > 0$, and let $q = q(n) \geq 2$, $q = 1 \mod 2n$ be a poly(n)-bounded prime such that $\alpha q \geq \omega(\sqrt{\log n})$. Then there is a polynomial-time quantum reduction from $\tilde{O}(\sqrt{n/\alpha})$-approximate SIVP (or SVP) on ideal lattices to the problem of solving $\mathsf{dRLWE}_{n,q,\xi}$ given only ℓ samples, where $\xi = \alpha q \cdot (n\ell / \log(n\ell))^{\frac{1}{4}}$.*

Theorem 2 ([14], Theorem 1) *Let $\alpha = \alpha(n) > 0$, and let $q = q(n) \geq 2$, $q \cong 3 \mod 8$ be a poly(n)-bounded prime such that there is another prime $p \cong 1 \mod 2n$. Let $\alpha q \geq n^{\frac{3}{2}} \ell^{\frac{1}{4}} \omega(\log^{\frac{9}{4}} n)$. Then there is a polynomial-time quantum reduction from $\tilde{O}(\sqrt{n/\alpha})$-approximate SIVP (or SVP) on ideal lattices to the problem of solving $\mathsf{dRLWE}_{n,q,\xi}$ given only ℓ samples, where $\xi = \alpha q$.*

Lemma 4 (Noise Rerandomization [14]). *Let q, ℓ, m be positive integers and r be a positive real satisfying $r > \max\{\omega(\sqrt{\log m}), \omega(\sqrt{\log \ell})\}$. Let $\boldsymbol{b} \in \mathbb{Z}_q^m$ be arbitrary and $\boldsymbol{x}$ chosen from $\mathcal{D}_{\mathbb{Z}^m, r}$. Then for any $\mathbf{V} \in \mathbb{Z}^{m \times \ell}$ and positive real $\sigma > s_1(\mathbf{V})$, there exists a PPT algorithm $\mathsf{ReRand}(\mathbf{V}, \boldsymbol{b} + \boldsymbol{x}, r, \sigma)$ that outputs $\boldsymbol{b}' = \boldsymbol{b}\mathbf{V} + \boldsymbol{x}'$ where $\boldsymbol{x}'$ is distributed statistically close to $\mathcal{D}_{\mathbb{Z}^\ell, 2r\sigma}$.*

2.2 NTRU Lattices

Definition 4 (NTRU lattices) *Let $\mathrm{f}, \mathrm{g} \in \mathcal{R}_q$ and $\mathrm{h} = \mathrm{g} \cdot \mathrm{f}^{-1} \mod q$, the NTRU lattice associated to h and q is*

$$\Lambda_{\mathrm{h},q} = \{(\mathrm{u}, \mathrm{v}) \in \mathcal{R}_q^2 | \mathrm{u} + \mathrm{v} \cdot \mathrm{h} = 0 \mod q\} \tag{1}$$

$\Lambda_{h,q}$ is a full-rank lattice in $\mathbb{Z}^{2n}$ generated by the rows of $\begin{bmatrix} q\mathbf{I}_n & 0 \\ -\mathsf{rot}_n(h) & \mathbf{I}_n \end{bmatrix}$,

where $\mathsf{rot}_n(\mathrm{h})$ is an anticirculant matrix defined as follow:

Definition 5 *An N-dimensional anticirculant matrix of f is the following Toeplitz matrix:*

$$\mathsf{rot}_n(\mathrm{f}) = \begin{bmatrix} f_0 & f_1 & f_2 & \cdots & f_{n-1} \\ -f_{n-1} & f_0 & f_1 & \cdots & f_{n-2} \\ -f_{n-2} & -f_{n-1} & f_0 & \cdots & f_{n-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -f_1 & -f_2 & -f_3 & \cdots & f_0 \end{bmatrix} = \begin{bmatrix} \phi(\mathrm{f}) \\ \phi(x \cdot \mathrm{f}) \\ \phi(x^2 \cdot \mathrm{f}) \\ \vdots \\ \phi(x^{n-1} \cdot \mathrm{f}) \end{bmatrix},$$

where f_i represents the coefficient of term x^i in f.

When it is clear from the context, we will drop the subscript n, and just write $\mathsf{rot}(\mathrm{f})$. For vector of ring elements $\mathbf{f} \in \mathcal{R}_q^\ell$, $\mathsf{rot}(\mathbf{f}) \in \mathbb{Z}_q^{n \times n\ell}$.

Lemma 5. *Let $\mathrm{f}, \mathrm{g} \in \mathcal{R}$, we have $\mathsf{rot}(\mathrm{f}+\mathrm{g}) = \mathsf{rot}(\mathrm{f}) + \mathsf{rot}(\mathrm{g})$, $\mathsf{rot}(\mathrm{f} \cdot \mathrm{g}) = \mathsf{rot}(\mathrm{f}) \cdot \mathsf{rot}(\mathrm{g})$, $\phi(\mathrm{f} \cdot \mathrm{g}) = \phi(\mathrm{f}) \cdot \mathsf{rot}(\mathrm{g})$, and $[\phi(\mathrm{f} \cdot \mathrm{g})]_j = \phi(\mathrm{f}) \cdot [\mathsf{rot}(\mathrm{g})]_j$.*

Definition 6 (NTRU Assumption) *Let $\mathrm{h} = \mathrm{g} \cdot \mathrm{f}^{-1}$ over $\mathcal{R}_q$ where $\mathrm{g}, \mathrm{f} \leftarrow \chi$ are short. The pair (f, g) is called a trapdoor of the $\mathsf{NTRU}_{n,q,\chi}$ instance h. It holds that $h \overset{c}{\approx} u$ where $u \overset{\$}{\leftarrow} \mathcal{R}_q$.*

Definition 7 (Decisional NTRU Problem ($\mathsf{dNTRU}_{n,q,\chi}$)[20]) *Given a sample $\mathrm{a}_i \in \mathcal{R}_q$, where it is distributed according to either:(i) $\mathsf{NTRU}_{n,q,\chi}$, or (ii) the uniformly random, distinguish which is the case (with over negligible advantage).*

2.3 Trapdoors and Sampling Algorithm

We first recall the notion of gadget vector $\mathbf{g}_b$ which was defined in [18].

Theorem 3 ([18], Theorem 4.1 and [14], Lemma 5) *For any integers $q \geq 2$, $b \geq 2$, $n \geq 1$, $d = \lceil \log_b q \rceil$, there exists a primitive vector $\mathbf{g}_b \in \mathcal{R}_q^d$ such that*

- *the lattice $\Lambda^{\perp}(\mathbf{g}_b)$ has a known basis $\mathbf{T}_g \in \mathcal{R}^{d \times d}$ with $\|\tilde{\mathbf{T}}_g\| \leq \sqrt{b^2+1}$ and $\|\mathbf{T}_g\| \leq \max\{\sqrt{b^2+1}, (b-1)\sqrt{k}\}$;*
- *there exists a deterministic polynomial time algorithm $\mathbf{g}_b^{-1}$ which takes a vector $\mathbf{u} \in \mathcal{R}_q^d$ as input and outputs $\mathbf{R} \in [-b, b]_{\mathcal{R}}^{d \times d}$ and $\mathbf{g}_b \cdot \mathbf{R} = \mathbf{u} \mod q$.*

Lemma 6 (NTRU $\mathsf{GenBasis}$[9]). *There exists an efficient algorithm $\mathsf{GenBasis}$ which on input the security parameter 1^{λ}, outputs an NTRU lattice $\Lambda_{h,q}$ along with a short basis $\mathbf{B}$. The Gram-Schmidt norm of the basis $\mathbf{B}$ is $\|\tilde{\mathbf{B}}\| \approx \sqrt{\frac{q \cdot e}{2}}$.*

Theorem 4 ([11]) *For any lattice basis $\mathbf{B} \in \mathbb{Z}^{n \times n}$, any target vector $\mathbf{c} \in \mathbb{Z}^n$ and $\sigma \geq \|\tilde{\mathbf{B}}\| \omega(\sqrt{\log n})$, there exists an algorithm $\mathsf{SampleD}(\mathbf{B}, \sigma, \mathbf{c}) \to \mathbf{t} \in \mathbb{Z}_q$ where $\mathbf{t}$ is within negligible statistical distance of $\mathcal{D}_{\Lambda(\mathbf{B}), \sigma, \mathbf{c}}$.*

Then, for an NTRU lattice $\Lambda_{\mathrm{h},q} = \{(\mathrm{u}, \mathrm{v}) \in \mathcal{R}_q^2 | \mathrm{u} + \mathrm{v} \cdot \mathrm{h} = 0 \mod q\}$ equipped with a short basis $\mathbf{B}$, given $\mathrm{u} \in \mathcal{R}_q$ we can sample $\mathrm{s}_1, \mathrm{s}_2 \leftarrow \mathcal{D}_{\Lambda_{\mathrm{u}}^{\perp}(1,\mathrm{h}), \sigma}$ by computing $(\mathrm{s}_1, \mathrm{s}_2) \leftarrow (\mathrm{u}, 0) - \mathsf{SampleD}(\mathbf{B}, \sigma, (\mathrm{u}, 0))$.

2.4 Algorithm SampleLeft and SampleRight

Now we introduce algorithms $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ which will be leveraged in our construction and security proof.

$\mathsf{SampleLeft}$ algorithm: Let n be a power of 2, $q \geq 2$ be a prime. We define the sample left algorithm as follows:
$\mathsf{SampleLeft}(\mathbf{h}, \mathbf{g}, \mathbf{B}_{\mathrm{h}}, \mathrm{u}, \sigma) \to \mathbf{s}$: On input $\mathbf{h} = [1, \mathrm{h}] \in \mathcal{R}_q^2$, $\mathbf{g} \in \mathcal{R}_q^d$, $\mathbf{B}_{\mathrm{h}}$ a basis for $\mathbf{h}$, $\mathrm{u} \in \mathcal{R}_q$ and $\sigma \geq \|\tilde{\mathbf{B}}_{\mathrm{h}}\| \omega(\sqrt{\log n})$, this algorithm samples $\mathbf{s}' \leftarrow \mathcal{D}_{\mathcal{R}^d, \sigma}$ and computes $\mathrm{u}' = \mathrm{u} - \mathbf{g} \cdot \mathbf{s}'^{\intercal}$. Then, it generates $(\mathrm{s}_1, \mathrm{s}_2) \leftarrow (\mathrm{u}', 0) - \mathsf{SampleD}(\mathbf{B}_{\mathrm{h}}, \sigma, (\mathrm{u}', 0))$ and outputs $\mathbf{s} = [\mathrm{s}_1, \mathrm{s}_2, \mathbf{s}']$. The distribution of $\mathbf{s}$ is close to $\mathcal{D}_{\Lambda_{[\mathbf{h}, \mathbf{g}]}^{\perp}, \sigma}$.

$\mathsf{SampleRight}$ algorithm Let n be a power of 2, $q \geq 2$ be a prime. Let $\mathbf{p}$ be a vector in $\mathcal{R}_q^d$, $\mathbf{B}_p$ be a basis of $\Lambda_q^{\perp}(\mathbf{p})$, $\mathbf{h} = [1, \mathrm{h}]$, $\mathbf{r}, \mathbf{e} \in \mathcal{R}_q^d$. This algorithm aims at output a vector $\mathbf{s} \in \mathcal{R}_q^{2+d}$ satisfies the distribution $\mathcal{D}_{\mathcal{R}_q^{2+d}, \sigma}$ such that

$$[1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r} + \mathbf{e} + \mathbf{p}] \cdot \mathbf{s}^{\intercal} = \mathrm{u} \mod q \tag{2}$$

We define the sample right algorithm for NTRU lattice as follows:
$\mathsf{SampleRight}(\mathbf{h}, \mathbf{p}, \mathbf{r}, \mathbf{e}, \mathbf{B}_{\mathrm{p}}, \mathrm{u}, \sigma) \to \mathbf{s}$: This algorithm will first generate a basis $\mathbf{T} \in \mathbb{Z}_q^{(2+d)n \times (2+d)n}$ for $\Lambda_{[\mathbf{h}, \mathbf{p}']}^{\perp}$ where $\mathbf{p}' = \mathrm{h} \cdot \mathbf{r} + \mathbf{e} + \mathbf{p}$. Then $\mathbf{T}$ can be used to sample $\mathbf{s}$ that satisfies eq. (2).

For $i = 1, ..., dn$, let $\boldsymbol{b}_i$ be the ith row of $\mathbf{B}_{\mathrm{p}} \in \mathbb{Z}_q^{dn \times dn}$, set

$$\boldsymbol{t}_i = \left[-\boldsymbol{b}_i \cdot \begin{bmatrix} \mathsf{rot}(\mathbf{e}) \\ \mathsf{rot}(\mathbf{r}) \end{bmatrix}^{\intercal}, \boldsymbol{b}_i \right] \in \mathbb{Z}_q^{(2+d)n}.$$

It is easy to see that $\mathsf{rot}([1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r} + \mathbf{e} + \mathbf{p}]) \cdot \boldsymbol{t}_i^{\intercal} = 0 \mod q$.

Let $\mathbf{U} \in \mathcal{R}_q^{2 \times d}$ be an arbitrary matrix such that $[1, \mathrm{h}, \mathbf{p}'] \cdot [\mathbf{I}_2, \mathbf{U}]^{\intercal} = 0 \mod q$ where $\mathbf{I}_2$ is the identity matrix. Then construct a matrix

$$\mathbf{T}_{\mathsf{half}} = [\mathsf{rot}(\mathbf{I}_2) - \mathsf{rot}(\mathbf{U}^{\intercal})^{\intercal} \cdot \begin{bmatrix} \mathsf{rot}(\mathbf{e}) \\ \mathsf{rot}(\mathbf{r}) \end{bmatrix}^{\intercal}, \mathsf{rot}(\mathbf{U}^{\intercal})^{\intercal}] \in \mathbb{Z}_q^{2n \times (2+d)n}.$$

Then $\mathsf{rot}([1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r} + \mathbf{e} + \mathbf{p}]) \cdot (\mathbf{T}_{\mathsf{half}})^{\intercal} = 0 \mod q$. For $i = dn + 1, \cdots, (2+d)n$, let $\boldsymbol{t}_i$ be the ith row of $\mathbf{T}_{\mathsf{half}}$.

Lemma 7 ([1],Lemma 18.). *The vectors* $\mathbf{T} = \{\boldsymbol{t}_1; \cdots; \boldsymbol{t}_{(2+d)n}\}$ *are linearly independent in* $\mathbb{Z}_q^{(2+d)n}$ *and satisfies*

$$\|\tilde{\mathbf{T}}\| \leq \|\tilde{\mathbf{B}}_{\mathrm{p}}\| \cdot \left(s_1 \left(\begin{bmatrix} \mathsf{rot}(\mathbf{e}) \\ \mathsf{rot}(\mathbf{r}) \end{bmatrix}^{\intercal} \right) + 1 \right)$$

Thus, with $\mathbf{T}$ as a basis for $\Lambda^{\perp}_{[\mathbf{h}, \mathbf{p}']}$, for $\sigma \geq \|\tilde{\mathbf{T}}\| \cdot \omega(\sqrt{\log n})$, this algorithm can use $\mathbf{T}$ to sample $\mathbf{s}$ such that $[1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r} + \mathbf{e} + \mathbf{p}] \cdot \mathbf{s}^{\intercal} = \mathrm{u} \mod q$. The distribution of $\mathbf{s}$ is close to $\mathcal{D}_{\Lambda^{\perp}_{[\mathbf{h}, \mathbf{p}']}, \sigma}$.

2.5 Identity-Based Encryption

Syntax An identity-based encryption (IBE) consists of four algorithms, namely, $\mathsf{KeySetup}$, $\mathsf{Extraction}$, $\mathsf{Encryption}$, $\mathsf{Decryption}$ defined as follows:

- $\mathsf{KeySetup}(1^\lambda) \to (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^λ, $\mathsf{KeySetup}$ algorithm generates master public key and master secret key pair $(\mathsf{mpk}, \mathsf{msk})$.
- $\mathsf{Extraction}(\mathsf{msk}, \mathsf{id}, \mathsf{mpk}) \to \mathsf{sk}_{\mathsf{id}}$: The extraction algorithm takes as input the master public key mpk, master secret key msk and an identity $\mathsf{id} \in \mathcal{ID}$, and outputs a secret key $\mathsf{sk}_{\mathsf{id}}$.
- $\mathsf{Encryption}(\mathsf{mpk}, \mathsf{id}, m) \to c$: On input a master public key mpk, an identity $\mathsf{id} \in \mathcal{ID}$, and a message m, $\mathsf{Encryption}$ algorithm outputs a ciphertext c.
- $\mathsf{Decryption}(\mathsf{mpk}, \mathsf{id}, \mathsf{sk}_{\mathsf{id}}, c) \to m$:The decryption algorithm takes as input the master public key mpk, a private key $\mathsf{sk}_{\mathsf{id}}$, and a ciphertext c. It outputs a message m.

Correctness. The correctness of an IBE is defined as: for all security parameter 1^λ, all $\mathsf{id} \in \mathcal{ID}$, it holds that

$$\Pr[\mathsf{Decryption}(\mathsf{mpk}, \mathsf{sk}_{\mathsf{id}}, \mathsf{Encryption}(\mathsf{mpk}, \mathsf{id}, m)) = m] = 1 - \mathsf{negl}(\lambda),$$

where $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{KeySetup}(1^\lambda)$, $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{Extraction}(\mathsf{msk}, \mathsf{id}, \mathsf{mpk})$.

IND-CPA security The adaptive IND-CPA security for an IBE scheme is defined by the following game between adversary $\mathcal{A}$ and challenger $\mathcal{C}$.

- *Setup.* Challenger $\mathcal{C}$ runs $\mathsf{KeySetup}(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$ and sends mpk to adversary $\mathcal{A}$.
- *Key Extraction Query* Adversary $\mathcal{A}$ sends an user ID id to challenge $\mathcal{C}$, $\mathcal{C}$ returns $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathsf{Extraction}(\mathsf{msk}, \mathsf{id}, \mathsf{mpk})$.
- *Challenge.* Adversary $\mathcal{A}$ chooses an identity id^*,two messages m_0 and m_1 on which it wishes to be challenged. If id^* has been queried in key extraction query, $\mathcal{C}$ abort and outputs $\perp$. Otherwise, $\mathcal{C}$ tosses a coin $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and sends $C^* \leftarrow \mathsf{Encryption}(\mathsf{mpk}, \mathsf{id}^*, m_{\mathsf{coin}})$ to $\mathcal{A}$.
- *Key Extraction Query.* Adversary $\mathcal{A}$ can continue to make key extraction query on identity $\mathsf{id} \neq \mathsf{id}^*$.
- *Output.* $\mathcal{A}$ output a coin coin^*.

$\mathcal{A}$ wins the game if $\mathsf{coin}^* = \mathsf{coin}$. The advantage of $\mathcal{A}$ is:

$$Adv_{\mathcal{A}} = \Pr[\mathcal{A} \text{ wins IND-CPA game}] - \frac{1}{2}.$$

Definition 8 (IND-CPA) *An IBE scheme is adaptive IND-CPA secure if for any polynomial-time adversary $\mathcal{A}$, $Adv_{\mathcal{A}}^{\mathsf{IBE}}$ is negligible.*

3 Relaxed Programmable Hash Function rPHF

In this section, we begin by introducing the definition of our rPHF. Then, we present two distinct instantiations: Type-I rPHF and Type-II rPHF. These instantiations are designed to cater to different requirements and offer flexibility in the construction of other cryptography primitives.

A $(u, v, \beta, \gamma, \delta)$-$r$PHF function $\mathcal{H}$ consists of four algorithms, namely, $\mathcal{H}.\mathsf{Gen}$, $\mathcal{H}.\mathsf{Eval}$, $\mathcal{H}.\mathsf{TrapGen}$, $\mathcal{H}.\mathsf{TrapEval}$, described below.

- $\mathcal{H}.\mathsf{Gen}(1^\lambda)$: this algorithm takes security parameter 1^λ as input and, outputs a normal mode hash key K.
- $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathrm{X}) \rightarrow \mathbf{Z}$: On input a key K and $\mathrm{X} \in \mathcal{X}$, this algorithm outputs a hash output $\mathbf{Z} \in \mathbb{Z}_q^{n \times \bar{m}}$.
- $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathbf{A}, \mathbf{B}) \rightarrow (\mathsf{K}, \mathsf{td})$: on input user specified uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, and a (public) trapdoor matrix $\mathbf{B} \in \mathbb{Z}_q^{n \times \bar{m}}$, this PPT algorithm outputs a key K together with a trapdoor td.
- $\mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}, \mathrm{X}) \rightarrow (\mathbf{R}_{\mathrm{x}}, \mathbf{E}_{\mathrm{x}}, \mathbf{S}_{\mathrm{x}})$: On input a key K along with its trapdoor td, and $\mathrm{X} \in \mathcal{X}$, this deterministic algorithm returns $\mathbf{R}_{\mathrm{x}} \in \mathbb{Z}_q^{m \times \bar{m}}$, $\mathbf{E}_{\mathrm{x}} \in \mathbb{Z}_q^{n \times \bar{m}}$, and $\mathbf{S}_{\mathrm{x}} \in \mathbb{Z}_q^{\bar{m} \times \bar{m}}$ such that $s_1(\mathbf{R}_{\mathrm{x}}; \mathbf{E}_{\mathrm{x}}) \leq \beta$ hold with overwhelming probability.

$\mathcal{H}.\mathsf{TrapGen}$ and $\mathcal{H}.\mathsf{TrapEval}$ are sometimes referred to as the trapdoor key generation and trapdoor evaluation algorithm.

Correctness. For all $(\mathsf{K}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathbf{A}, \mathbf{B})$, all $\mathrm{X} \in X$ and $(\mathbf{R}_\mathrm{x}, \mathbf{E}_\mathrm{x}, \mathbf{S}_\mathrm{x}) \leftarrow \mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}, \mathrm{X})$, we have $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathrm{X}) = \mathbf{A}\mathbf{R}_\mathrm{x} + \mathbf{E}_\mathrm{x} + \mathbf{S}_\mathrm{x}\mathbf{B}$.

Computational Close Keys. For all $(\mathsf{K}', \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathbf{A}, \mathbf{B})$ and $\mathsf{K} \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda, \mathbf{A}, \mathbf{B})$, given any PPT adversary $\mathcal{A}$, we have

$$|\Pr[\mathcal{A}(\mathbf{A}, \mathsf{K}') = 0] - \Pr[\mathcal{A}(\mathbf{A}, \mathsf{K}) = 0]| \leq \gamma.$$

Well-distributed hidden matrices. For all $(\mathsf{K}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathbf{A}, \mathbf{B})$, any input $\mathrm{X}_1, \cdots, \mathrm{X}_u, \mathrm{Y}_1, \cdots, \mathrm{Y}_v \in \mathcal{X}$ such that $\mathrm{X}_i \neq \mathrm{Y}_j$ for any i, j, we have

$$\Pr[\mathbf{S}_{\mathrm{x}_1} = \cdots = \mathbf{S}_{\mathrm{x}_u} = 0 \wedge \mathbf{S}_{\mathrm{Y}_1}, \cdots, \mathbf{S}_{\mathrm{Y}_v} \in \mathcal{I}] \geq \delta,$$

$$(\mathbf{R}_{\mathrm{x}_i}, \mathbf{E}_{\mathrm{x}_i}, \mathbf{S}_{\mathrm{x}_i}) \leftarrow \mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}, \mathrm{X}_i),$$

$$(\mathbf{R}_{\mathrm{y}_i}, \mathbf{E}_{\mathrm{y}_i}, \mathbf{S}_{\mathrm{y}_i}) \leftarrow \mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}, \mathrm{Y}_i),$$

where $\mathcal{I}$ is the set of all the invertible matrices. If γ is negligible and δ is noticeable, we say $\mathcal{H}$ is a (u, v, β)-$r\mathsf{PHF}$.

3.1 Type-I rPHF from RLWE assumption

Let the hash input space be $\{0,1\}^\ell$ for some $\ell \in \mathbb{N}$. We first present Type-I $r\mathsf{PHF}$, which features $O(\ell)$ size hash key and constant size hash output. The output size can be as small as 2 ring elements. This instantiation has a bound on the v that $v < \frac{q}{2}$.

Definition 9 *The $r\mathsf{PHF}$ $\mathcal{H}$ =($\mathcal{H}.\mathsf{Gen}$, $\mathcal{H}.\mathsf{Eval}$, $\mathcal{H}.\mathsf{TrapGen}$, $\mathcal{H}.\mathsf{TrapEval}$) with key space $\mathcal{K} \in (\mathcal{R}_q^d)^{\ell+1}$ is defined as follows:*

- $\mathcal{H}.\mathsf{Gen}(1^\lambda) \rightarrow \mathsf{K}$: on input the security parameter 1^λ, uniformly random sample $d(\ell+1)$ ring elements

$$\{\mathbf{h}_i = (\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)})\}_{i \in [\ell+1]} \xleftarrow{\$} \mathcal{K}.$$

  Return $\mathsf{K} = \{\boldsymbol{h}_i\}_{i \in [\ell+1]}$.

- $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathrm{X}) \rightarrow \mathbf{z}$: on input $\mathrm{X} = \{x_1, \cdots, x_\ell\} \in \{0,1\}^\ell$ and the key $\mathsf{K} = \{\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)}\}_{i \in [\ell+1]}$, compute

$$\boldsymbol{z} = (\mathrm{h}_0^{(0)}, \cdots, \mathrm{h}_0^{(d-1)}) + \sum_{i=1}^{\ell} (-1)^{x_i} \cdot (\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)})$$

- $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g}) \rightarrow (\mathsf{K}', \mathsf{td})$: given uniform random ring element $\mathrm{a} \xleftarrow{\$} \mathcal{R}_q$ and trapdoor vector $\mathbf{g} = (\mathrm{g}^{(0)}, \cdots, \mathrm{g}^{(d-1)})$, for each $\mathbf{h}_i$, this algorithm chooses $\mathbf{r}_i \leftarrow \mathcal{D}_{\mathcal{R}^d, \beta}$, $\mathbf{e}_i \leftarrow \mathcal{D}_{\mathcal{R}^d, \beta}$, $s_i \leftarrow \mathbb{Z}_q$, and then computes

$$\mathbf{h}_i = (\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)}) = \mathrm{a} \cdot \mathbf{r}_i + \mathbf{e}_i + s_i \mathbf{g}.$$

  Set $\mathsf{K}' = \{\mathbf{h}_i\}_{i \in [\ell+1]}$, $\mathsf{td} = (\{\mathbf{r}_i, \mathbf{e}_i\}_{i=0}^{\ell}, \mathbf{s} = \{s_i\}_{i=0}^{\ell})$.

– $\mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}', \mathrm{X}) \to (\mathbf{r}_{\mathrm{x}}, \mathbf{e}_{\mathrm{x}}, s_{\mathrm{x}})$: given trapdoor $\mathsf{td} = (\{\mathbf{r}_i, \mathbf{e}_i\}_{i=0}^{\ell}, \mathbf{s} = \{s_i\}_{i=0}^{\ell})$, hash key $\mathsf{K}' = \{\mathbf{h}_i\}_{i\in[\ell+1]}$, and $\mathrm{X} = \{x_1, \cdots, x_\ell\} \in \{0,1\}^\ell$, this algorithm computes

$$\mathbf{r}_{\mathrm{x}} = \mathbf{r}_0 + \sum_{i=1}^{\ell}(-1)^{x_i} \cdot \mathbf{r}_i, \mathbf{e}_{\mathrm{x}} = \mathbf{e}_0 + \sum_{i=1}^{\ell}(-1)^{x_i} \cdot \mathbf{e}_i, s_{\mathrm{x}} = s_0 + \sum_{i=1}^{\ell}(-1)^{x_i} \cdot s_i.$$

Looking ahead, we are going to use this Type-I $r\mathsf{PHF}$ to construct lattice-based IBE schemes with short secret keys and ciphertexts. In the constructions and security proofs, we will use $d = 2$ and $\mathbf{g} = [1, \mathrm{g}]$ where g represents an NTRU lattice with a short basis.

Computational Close keys. The distribution of hash keys $\mathsf{K}' = \{(\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)})\}_{i=0}^{\ell}$ generated from $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g})$ is computationally indistinguishable from $\mathsf{K} \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda)$ according to Lemma 8.

Well-distributed hidden matrices. For all $(\mathsf{K}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g})$, any input $\mathrm{X}_1, \mathrm{Y}_1, \cdots, \mathrm{Y}_v \in \{0,1\}^\ell$ such that $\mathrm{X}_1 \neq \mathrm{Y}_i$ for any i, we have

$$\Pr[s_{\mathrm{X}_1} = 0 \wedge s_{\mathrm{Y}_1}, \cdots, s_{\mathrm{Y}_v} \in \mathcal{I}] \geq \gamma$$

where $\gamma \geq \frac{1}{q}(1 - \frac{v}{q})$. We require $v < \frac{q}{2}$ in the security proof. We omit the detailed proof here due to page limitation. The proof is similar to the security proof for adaptively secure IBE in [1].

Lemma 8. *For n be a power of 2, q be a prime such that $q \equiv 1 \mod 2n$, $\beta = n^{\frac{1}{4}} \log n$, $\mathrm{r}_i^{(j)}, \mathrm{e}_i^{(j)} \leftarrow \mathcal{D}_{\mathcal{R},\beta}$ and $\mathrm{a} \xleftarrow{\$} \mathcal{R}_q$ we have distributions*

$$\left(\mathrm{a}, \{\mathrm{a}\cdot \mathrm{r}_i^{(0)} + \mathrm{e}_i^{(0)} + s_i \cdot \mathrm{g}^{(0)}, \cdots, \mathrm{a}\cdot \mathrm{r}_i^{(d-1)} + \mathrm{e}_i^{(d-1)} + s_i \cdot \mathrm{g}^{(d-1)}\}_{i=0}^{\ell}\right),$$

$$\left(\mathrm{a}, \{\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)}\}_{i=0}^{\ell}\right)$$

computationally close.

Proof. We prove the lemma through a sequence of games between a challenger $\mathcal{C}$ and adversary $\mathcal{A}$

– Game_0: Challenger $\mathcal{C}$ first flips a $\mathsf{coin} \xleftarrow{\$} \{0,1\}$. If $\mathsf{coin} = 0$, $\mathcal{C}$ sends $(\mathrm{a}, \{(\tilde{\mathrm{h}}_i^{(0)}, \cdots, \tilde{\mathrm{h}}_i^{(d-1)})\}_{i=0}^{\ell})$ to $\mathcal{A}$ where $\tilde{\mathrm{h}}_i^{(j)} = \mathrm{h}\cdot \mathrm{r}_i^{(j)} + \mathrm{e}_i^{(j)} + s_i \cdot \mathrm{g}_i^{(j)}$; Otherwise, $\mathcal{C}$ sends $\left(\mathrm{a}, \{(\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)})\}_{i=0}^{k}\right)$ to $\mathcal{A}$. $\mathcal{A}$ outputs a bit $\mathsf{coin}^* \in \{0,1\}$.
We use W_i to describe the event that $\mathsf{coin}^* = \mathsf{coin}$ happens in Game_i. The advantage of $\mathcal{A}$ in Game_0 is $|\Pr[W_0] - \frac{1}{2}| = \nu_0$.

– Game_1: $\mathcal{C}$ performs almost the same as in Game_1, except that $\tilde{\mathrm{h}}_0^{(0)}$ now is sampled uniformly random from $\mathcal{R}_q$. We now argue that the $|\Pr[W_1] - \Pr[W_2]| = \mathsf{negl}(n)$.

On receiving a problem instance of dRLWE $\{\mathrm{a}', \mathrm{u}'\}$, $\mathcal{C}$ sets $\mathrm{a} = \mathrm{a}'$, $\tilde{\mathrm{h}}_0^{(0)} = \mathrm{u}' + s_0 \cdot \mathrm{g}^{(0)}$ while other steps remain the same as in Game_1. If $\{\mathrm{a}', \mathrm{u}'\}$ is a valid RLWE sample, then this game is identical to Game_0. If $\mathrm{u}' \xleftarrow{\$} \mathcal{R}_q$, this game is identical to Game_1. Therefore, we have $\mathsf{adv}^{\mathsf{dRLWE}_{n,q,\beta}} = |\Pr[W_0] - \Pr[W_1]|$. pkc

- Game_2: $\mathcal{C}$ performs almost the same as in Game_2, except that $\tilde{\mathrm{h}}_0^{(1)}$ is now sampled uniformly random from $\mathcal{R}_q$. We have $|\Pr[W_1] - \Pr[W_2]| = \mathsf{adv}^{\mathsf{dRLWE}_{n,q,\beta}}$ due to the hardness of $\mathsf{dRLWE}_{n,q,\beta}$. The proof is identical to the proof in Game_1
- $\mathsf{Game}_{d(\ell+1)}$: So on so forth, in $\mathsf{Game}_{d(\ell+1)}$, $\mathcal{C}$ samples $\{\tilde{\mathrm{h}}_i^{(0)}, \cdots, \tilde{\mathrm{h}}_i^{(d-1)}\}_{i=0}^{\ell}$ all uniformly random from $\mathcal{R}_q$. Distributions $(\mathrm{h}, \{(\tilde{\mathrm{h}}_i^{(0)}, \cdots, \tilde{\mathrm{h}}_i^{(d-1)})\}_{i=0}^{\ell})$ and $(\mathrm{h}, \{(\mathrm{h}_i^{(0)}, \cdots, \mathrm{h}_i^{(d-1)})\}_{i=0}^{\ell})$ are now identical and we have $|\Pr[W_{d(\ell+1)}] - \Pr[W_{d(\ell+1)-1}]| = \mathsf{adv}^{\mathsf{dRLWE}_{n,q,\beta}}$.

From the above, we have

$$\left|\Pr[W_{d(\ell+1)} - \frac{1}{2}]\right| = \left|\Pr[W_0] - \frac{1}{2} + \sum_{1}^{d(\ell+1)} (\Pr[W_i] - \Pr[W_{i-1}])\right|$$
$$= d(\ell+1) \cdot \mathsf{adv}^{\mathsf{dRLWE}_{n,q,\beta}} + \nu_0 = \mathsf{negl}(n).$$

3.2 Type-II rPHF from RLWE assumption

Based on [14], we present our Type-II $r\mathsf{PHF}$ with hash key length $O(\sqrt{\ell})$.

Definition 10 *The Type-*II $r\mathsf{PHF}$ $\mathcal{H} = (\mathcal{H}.\mathsf{Gen}, \mathcal{H}.\mathsf{Eval}, \mathcal{H}.\mathsf{TrapGen}, \mathcal{H}.\mathsf{TrapEval})$ *with key space* $\mathcal{K} \in (\mathcal{R}_q^d)^{2\times k}$*, where* $k = \sqrt{\ell}$*, is defined as follows:*

The hash input space is $\{0,1\}^\ell$ for some $\ell \in \mathbb{N}$. S is an injective map that maps $X \in \{0,1\}^\ell$ to a subset $S(X)$ of $[1,k]^2$, where $k = \lceil \ell^{\frac{1}{2}} \rceil$. This injective map can be done by regarding X as the indicator vector of a subset of $[1,k]^2$.

- $\mathcal{H}.\mathsf{Gen}(1^\lambda) \to \mathsf{K}$: on input the security parameter 1^λ, uniformly random sample $d(1 + 2 \times k)$ ring elements, where $k = \sqrt{\ell}$.

$$\Big(\mathbf{h}_0 = (\mathrm{h}_0^{(0)}, \cdots, \mathrm{h}_0^{(d-1)}), \{\mathbf{h}_{i,j} = (\mathrm{h}_{i,j}^{(0)}, \cdots, \mathrm{h}_{i,j}^{(d-1)})\}_{(i,j)\in[2]\times[k]}\Big) \xleftarrow{\$} \mathcal{K}.$$

  Return $\mathsf{K} = \big(\mathbf{h}_0, \{\mathbf{h}_{i,j}\}_{(i,j)\in[2]\times[k]}\big)$.
- $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathrm{X}) \to \mathbf{z}$: on input $\mathrm{X} = \{x_1, \cdots, x_\ell\} \in \{0,1\}^\ell$ and the key $\mathsf{K} = \big(\mathbf{h}_0, \{\mathbf{h}_{i,j}\}_{(i,j)\in[2]\times[k]}\big)$, compute

$$\mathbf{z} = \mathbf{h}_0 + \sum_{(j_1,j_2)\in S(X)} \mathbf{h}_{1,j_1} \cdot \mathbf{g}_b^{-1}(\mathbf{h}_{2,j_2}) \in \mathcal{R}_q^d.$$

Notation $\mathbf{g}_b^{-1}$ is defined as for any $\mathbf{u} \in \mathcal{R}_q^d$, $\mathbf{g}_b^{-1}(\mathbf{u})$ deterministically outputs a short matrix $\mathbf{R} \in \mathcal{R}_q^{d\times d}$ such that $\mathbf{g}_b \cdot \mathbf{R} = \mathbf{u}$ where $\mathbf{R} \in [-b, b]_{\mathcal{R}}^{d\times d}$.

- $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g}_b) \to (\mathsf{K}', \mathsf{td})$: given a uniform random ring element $\mathrm{a} \xleftarrow{\$} \mathcal{R}_q$ and the gadget vector $\mathbf{g}_b = (\mathrm{g}^{(0)}, \cdots, \mathrm{g}^{(d-1)}) \in \mathcal{R}_q^d$, for $\mathbf{h}_0$ and $\mathbf{h}_{i,j}$, this algorithm chooses $\mathbf{r}_0, \mathbf{r}_{i,j} \leftarrow \mathcal{D}_{\mathcal{R}^d,\beta}$, $\mathbf{e}_0, \mathbf{e}_{i,j} \leftarrow \mathcal{D}_{\mathcal{R}^d,\beta}$, $\mathrm{s}_0 \xleftarrow{\$} [-\ell(\kappa n)^2, -1]_{\mathcal{R},2(\kappa-1)+1}$, $\mathrm{s}_{i,j} \xleftarrow{\$} [1, n]_{\mathcal{R},\kappa}$, and then computes

$$\mathbf{h}_0 = \mathrm{a}\cdot \mathbf{r}_0 + \mathbf{e}_0 + \mathrm{s}_0 \cdot \mathbf{g}_b, \mathbf{h}_{i,j} = \mathrm{a} \cdot \mathbf{r}_{i,j} + \mathbf{e}_{i,j} + \mathrm{s}_{i,j} \cdot \mathbf{g}_b, (i,j) \in [2] \times [k].$$

  Set $\mathsf{K}' = \big(\mathbf{h}_0, \{\mathbf{h}_{i,j}\}_{(i,j)\in[2]\times[k]}\big)$, $\mathsf{td} = (\mathbf{r}_0, \mathbf{e}_0, \{\mathbf{r}_{i,j}, \mathbf{e}_{i,j}\}_{(i,j)\in[2]\times[k]}, \mathbf{s} = (\mathrm{s}_0, \{\mathrm{s}_{i,j}\}_{(i,j)\in[2]\times[k]}))$.
- $\mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}', \mathrm{X}) \to (\mathbf{r}_\mathrm{x}, \mathbf{e}_\mathrm{x}, \mathrm{s}_\mathrm{x})$: given trapdoor td, hash key $\mathsf{K}' = \big(\mathbf{h}_0, \{\mathbf{h}_{i,j}\}_{(i,j)\in[2]\times[k]}\big)$, and $\mathrm{X} = \{x_1, \cdots, x_\ell\} \in \{0,1\}^\ell$, this algorithm computes

$$\mathbf{r}_\mathrm{x} = \mathbf{r}_0 + \sum_{(j_1,j_2)\in S(X)} \mathbf{r}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{r}_{2,j_2},$$

$$\mathbf{e}_\mathrm{x} = \mathbf{e}_0 + \sum_{(j_1,j_2)\in S(X)} \mathbf{e}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{e}_{2,j_2},$$

$$\text{and, } \mathrm{s}_\mathrm{x} = s_0 + \sum_{(j_1,j_2)\in S(X)} \mathrm{s}_{1,j_1}\cdot \mathrm{s}_{2,j_2}.$$

  where $\mathbf{R}_{j_2} = \mathbf{g}_b^{-1}(\mathbf{h}_{2,j_2}) \in [-b, b]_{\mathcal{R}}^{d\times d}$.

The above $\mathbf{r}_\mathrm{x}$, $\mathbf{e}_\mathrm{x}$, and s_x are computed from:

$$\begin{aligned}
&\mathbf{h}_0 + \sum_{(j_1,j_2)\in S(X)} \mathbf{h}_{1,j_1} \cdot \mathbf{g}_b^{-1}(\mathbf{h}_{2,j_2}) \\
=&\mathbf{h}_0 + \sum_{(j_1,j_2)\in S(X)} (\mathrm{a}\cdot \mathbf{r}_{1,j_1} + \mathbf{e}_{1,j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{g}_b)\cdot \mathbf{g}_b^{-1}\left(\mathrm{a}\cdot \mathbf{r}_{2,j_1} + \mathbf{e}_{2,j_2} + \mathrm{s}_{2,j_2}\cdot \mathbf{g}_b\right) \\
=&\mathbf{h}_0 + \sum_{(j_1,j_2)\in S(X)} (\mathrm{a}\cdot \mathbf{r}_{1,j_1} + \mathbf{e}_{1,j_1}) \cdot \mathbf{g}_b^{-1}\left(\mathbf{h}_{2,j_2}\right) + \mathrm{s}_{1,j_1}\cdot (\mathrm{a}\cdot \mathbf{r}_{2,j_2} + \mathbf{e}_{2,j_2} + \mathrm{s}_{2,j_2}\mathbf{g}_b) \\
=&\mathbf{h}_0 + \sum_{(j_1,j_2)\in S(X)} \mathrm{a}(\mathbf{r}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{r}_{2,j_2}) + (\mathbf{e}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{e}_{2,j_2}) + \mathrm{s}_{1,j_1}\cdot \mathrm{s}_{2,j_2}\cdot \mathbf{g}_b.
\end{aligned}$$

For $\|\mathrm{s}_{1,j_1}\|_1 \le \kappa$, $\mathbf{r}' = \mathbf{r}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{r}_{2,j_2}$, $\mathbf{e}' = \mathbf{e}_{1,j_1}\cdot \mathbf{R}_{j_2} + \mathrm{s}_{1,j_1}\cdot \mathbf{e}_{2,j_2}$, furthermore, we have

$$s_1([\mathbf{r}';\mathbf{e}']) \le s_1([\mathbf{r}_{1,j_1};\mathbf{e}_{1,j_1}])\cdot s_1(\mathbf{R}_{j_2}) + s_1(\mathrm{s}_{1,j_1})\cdot s_1([\mathbf{r}_{2,j_2};\mathbf{e}_{2,j_2}]) = \vartheta\cdot bnd + \vartheta\cdot \kappa$$

where ϑ is the s_1 norm of $[\mathbf{r}_{1,j_1};\mathbf{e}_{1,j_1}]$ and $[\mathbf{r}_{2,j_2};\mathbf{e}_{2,j_2}]$.

Computational Close keys. The distribution of hash keys generated from $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g}_b)$ is computationally indistinguishable from $\mathsf{K} \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda)$. We omit the proof here and the proof is similar to Lemma 8.

Well-distributed hidden matrices. For all $(\mathsf{K}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{a}, \mathbf{g})$, any input $\mathrm{X}_1, \mathrm{Y}_1, \cdots, \mathrm{Y}_v \in \{0,1\}^\ell$ such that $\mathrm{X}_1 \neq \mathrm{Y}_i$ for any i, we have

$$\Pr[\mathrm{s}_{\mathrm{X}_1} = 0 \wedge \mathrm{s}_{\mathrm{Y}_1}, \cdots, \mathrm{s}_{\mathrm{Y}_v} \in \mathcal{I}] \geq \delta$$

where δ is noticeable.

Lemma 9 ([14], Lemma 3). *Let q be a prime such that $q \equiv 3 \mod 8$ and n be a power of 2. Then, $x^n + 1$ splits as $x^n + 1 \equiv t_1(x) \cdot t_2(x) \mod q$ for two irreducible polynomials $t_1(x) = x^{\frac{n}{2}} + ux^{\frac{n}{4}} - 1$ and $t_2(x) = x^{\frac{n}{2}} - ux^{\frac{n}{4}} - 1$ in $\mathbb{Z}_q[x]$ where $u^2 = -2 \mod q$. Furthermore, all $y \in \mathcal{R}_q$ satisfying $\|\phi(y)\|_2 < \sqrt{q}$ are invertible.*

With the above lemma and Lemma 9 from [14], which argues that the probability $\mathrm{s}_{\mathrm{Y}_i}$ corresponding to the hash query Y_i being invertible and the s_{X} for the hash input X being zero is non-negligible, we can bound the probability δ when $q \equiv 3 \mod 8$ a prime and n be a power of 2 as

$$\frac{1}{(\ell\kappa^2 n^2)^{2\kappa+1}} \cdot (1 - \frac{4v}{n^\kappa}) \leq \delta \leq \frac{1}{(\ell\kappa^2 n^2)^{2\kappa+1}}.$$

For δ being noticeable, we require that $v > \frac{n^\kappa}{4}$. Thus, the above $r\mathsf{PHF}$ possess the *well distributed hidden matrices* property.

4 NTRU IBE from rPHF

In this section, we present the construction of our lattice-based IBE scheme based on $r\mathsf{PHF}$ along with the security proof. The construction can be instantiated by either our Type-I or Type-II $r\mathsf{PHF}$. For each instantiation, we provide the parameter set necessary to achieve the desired security properties.

$\mathsf{KeySetup}(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$: On input the security parameter 1^λ, this algorithm runs as follow:

- Generate $(\mathrm{h}, \mathbf{B}_\mathrm{h}) \leftarrow \mathsf{GenBasis}(1^\lambda)$.
- Generate hash key $\mathsf{K} \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda)$
- Sample a ring element $\mathrm{u} \xleftarrow{\$} \mathcal{R}_q$.
- Output the master public key mpk and master secret key msk: $\mathsf{mpk} = (\mathrm{h}, \mathsf{K}), \mathsf{msk} = \mathbf{B}_\mathrm{h}$.

$\mathsf{Extraction}(\mathsf{msk}, \mathsf{id}, \mathsf{mpk}) \rightarrow \mathsf{sk}_\mathsf{id}$: On input a master public key mpk=(h, K, u), a master secret key $\mathsf{msk} = \mathbf{B}_\mathrm{h} \in \mathbb{Z}^{2n \times 2n}$, and an identity $\mathsf{id} \in \{0,1\}^\ell$, this algorithm runs as following:

- Run $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathsf{id}) \rightarrow \mathbf{h}_\mathsf{id} = [\mathrm{h}_\mathsf{id}^{(0)}, \cdots, \mathrm{h}_\mathsf{id}^{(d-1)}]$.

– Sample $(\mathbf{s} = [\mathrm{s}_0, \cdots, \mathrm{s}_{d+1}]) \leftarrow \mathsf{SampleLeft}([1, \mathrm{h}], \mathbf{h}_{\mathsf{id}}, \mathbf{B}_{\mathrm{h}}, \mathrm{u}, \sigma)$ such that

$$[1, \mathrm{h}, \mathbf{h}_{\mathsf{id}}] \cdot \mathbf{s}^{\intercal} = u$$

– Output $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathbf{s} \in \mathcal{R}_q^{d+2}$.

$\mathsf{Encryption}(\mathsf{mpk}, \mathsf{id}, \mathrm{m}) \rightarrow C$: this algorithm takes a message $\mathrm{m} \in \{0,1\}^n$, a master public key $\mathsf{mpk} = (\mathrm{h}, \mathsf{K}, \mathrm{u})$, and an identity $\mathsf{id} \in \{0,1\}^\ell$ as input and does:

– Run $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, \mathsf{id}) \rightarrow \mathbf{h}_{\mathsf{id}} = [\mathrm{h}_{\mathsf{id}}^{(0)}, \cdots, \mathrm{h}_{\mathsf{id}}^{(d-1)}]$;

– Choose a uniformly random ring element $\mathrm{t} \xleftarrow{\$} \mathcal{R}_q$;

– Choose a uniformly random ring element $\mathrm{v} \xleftarrow{\$} \mathcal{R}_q^{\times}$;

– Choose noises $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}_q^{d+2}, \alpha}$, and $\mathrm{e}_1 \leftarrow \mathcal{D}_{\mathcal{R}_q, \alpha'}$;

– Let $\mathbf{f}_{\mathsf{id}} = [1, \mathrm{h}, \mathbf{h}_{\mathsf{id}}]$. Output $C \leftarrow \{\mathbf{c}, \mathrm{c}_m\}$ where

$$\mathbf{c} \leftarrow \mathrm{v} \cdot \mathbf{f}_{\mathsf{id}} \cdot \mathrm{t} + \mathbf{e}, \mathrm{c}_m \leftarrow (\mathrm{v} \cdot \mathrm{u}) \cdot \mathrm{t} + \mathrm{e}_1 + \mathrm{m} \lfloor \frac{q}{2} \rfloor$$

$\mathsf{Decryption}(C, \mathsf{sk}_{\mathsf{id}}) \rightarrow$ m: On input user secret key $\mathsf{sk}_{\mathsf{id}} = \mathbf{s} \in \mathcal{R}_q^{d+2}$, and ciphertext $C = \{\mathbf{c} \in \mathcal{R}_q^{d+2}, \mathrm{c}_m\}$, do

– Compute $\mathrm{w} \leftarrow \mathrm{c}_m - \mathbf{c} \cdot \mathbf{s}^{\intercal}$;

– Let $\mathrm{m} = \lfloor \frac{\mathrm{w}}{q/2} \rfloor$;

– Output m.

4.1 Correctness and Parameter Selection

In this section, we provide two sets of parameters: one for the IBE scheme based on our Type-I $r\mathsf{PHF}$, and another for the IBE scheme based on Type-II $r\mathsf{PHF}$.

From our Type-I $r\mathsf{PHF}$, the resulting IBE has mpk size of ℓ vectors, ciphertext size of 5 ring ellements, user secret key size of 4 ring elements. The IBE scheme from Type-II $r\mathsf{PHF}$ strikes a balance between the ciphertext length and mpk length, with a ciphertext length of 17 ring elements, user secret key size of 16 ring elements, and mpk length of $O(\sqrt{\ell})$.

IBE from Type-I $r\mathsf{PHF}$: We pick the following parameters:

$$q = \ell^4 \cdot n^4 \omega(\log^2 n), \quad \sigma = \ell^3 \cdot n^{\frac{11}{4}} \omega(\log^{\frac{3}{4}} n),$$

$$\alpha' = n^{\frac{1}{4}} \omega(\log^{\frac{1}{4}} n), \quad \alpha = \ell \cdot n \omega(\sqrt{\log n})$$

Also for Type-I $r\mathsf{PHF}$, we require that in the security proof the total number of identity extraction query Q issued by adversary should satisfy that $Q < q$.

– n, q: n is a power of 2 and q is a prime and satisfies $q \equiv 1 \mod 2n$.

– α': According to Definition 1, set $\alpha' = n^{\frac{1}{4}}\omega(\log^{\frac{1}{4}} n)$ where three $\mathsf{dRLWE}_{n,q,\alpha'}$ samples is given in the security proof.

– β: According to Theorem 13, we set $\beta = n^{\frac{1}{4}}\omega(\log^{\frac{1}{4}} n)$.

– α: We set $\alpha = 2s_1([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}])\alpha'$ with

$$s_1([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}]) \leq (\beta\cdot\ell+\delta)\sqrt{n}\cdot(2\sqrt{2}+\omega(\sqrt{\log n}))$$

for some constant $\delta \approx \frac{1}{\sqrt{2\pi}}$ according to Lemma 1.

– σ: σ should be sufficiently large so $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ work. Thus, we set $\sigma \geq s_1([\mathbf{R}_{\mathsf{id}^*}])\|\tilde{\mathbf{B}}\|\omega(\sqrt{\log n})$ where

$$s_1([\mathbf{R}_{\mathsf{id}^*}]) \leq \beta\cdot\ell\sqrt{n}\cdot(2\sqrt{2}+\omega(\sqrt{\log n})), \|\tilde{\mathbf{B}}\| \approx \sqrt{\frac{q\cdot e}{2}}.$$

– *Error term*: By Lemma 10, the error term $\mathrm{e}' = \mathrm{e}_5 - \mathbf{e}\cdot\mathbf{s}^\intercal$ in $\mathsf{Decryption}$ algorithm has the bound

$$|[\phi(\mathrm{e}_5 - \mathbf{e}\cdot\mathbf{s}^\intercal)]_j| \leq \ell^2 n^{\frac{3}{2}}\cdot\omega(\log n)\sqrt{q} + 1.5n^{\frac{1}{4}}\omega(\log^{\frac{1}{4}} n).$$

Lemma 10. $|\phi(\mathbf{e})\cdot[\mathsf{rot}(\mathbf{s}^\intercal)]_j| \leq \ell^2 n^{\frac{3}{2}}\cdot\omega(\log^{\frac{3}{2}} n)\sqrt{q}$.

Proof. By Lemma 3, we have

$$\phi(\mathbf{e})\cdot[\mathsf{rot}(\mathbf{s}^\intercal)]_j \leq \|[\mathsf{rot}(\mathbf{s}^\intercal)]_j\|\alpha\omega(\sqrt{\log n})$$

And by Lemma 2 and 6, we have

$$\|[\mathsf{rot}(\mathbf{s}^\intercal)]_j\| \leq \beta\cdot\ell n\cdot(2\sqrt{2}+\omega(\sqrt{\log n}))\sqrt{\frac{q\cdot e}{2}}.$$

Thus we have $|\phi(\mathbf{e})\cdot[\mathsf{rot}(\mathbf{s}^\intercal)]_j| \leq \ell^2 n^2\cdot\omega(\log n)\sqrt{q}$.

Lemma 11 (Correctness). *Assume that error term is less than $q/5$ holds with overwhelming probability. Then the above scheme has negligible decryption error.*

Thus, to guarantee the correctness, we have

$$\ell^2 n^2\cdot\omega(\log n)\sqrt{q} + 1.5n^{\frac{1}{4}}\omega(\log^{\frac{1}{4}} n) < \frac{q}{5}.$$

IBE from Type-II rPHF: We pick the following parameters:

$$q = \ell^2 n^{12}\cdot\omega(\log^2 n), \quad \sigma = \ell n^6\omega(\log n),$$

$$\alpha' = n^{\frac{3}{2}}\omega(\log^{\frac{9}{4}} n), \quad \alpha = d^{\frac{3}{2}}\ell n^{\frac{11}{2}}\omega(\log^{\frac{9}{4}} n)$$

For Type-II $r\mathsf{PHF}$, we require that in the security proof the total number of identity extraction query Q issued by adversary should satisfy that $Q < \frac{n^\kappa}{4}$ for a constant κ.

– n, q: n is a power of 2 and q is a prime and satisfies $q \equiv 3 \mod 8$.

– d: $d = \log_b q$ and here we set $b = n$.

– α' *and* β: $\alpha', \beta = n^{\frac{3}{2}}\omega(\log^{\frac{9}{4}} n)$ according to Theorem 2.

– $s_1(\mathbf{R}_{\mathsf{id}})$: According to Lemma 1, $s_1[\mathbf{e}_{i,j}; \mathbf{r}_{i,j}] = n^2 \cdot (\sqrt{2} + \sqrt{d} + \omega(\sqrt{\log n}))$we have $s_1(\mathbf{R}_{\mathsf{id}}) \approx n^2\sqrt{d\ell} \cdot (bnd + \kappa n)$.

– α: We set $\alpha = 2s_1([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}])\alpha'$ with $s_1([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}]) \leq n^2\sqrt{d\ell} \cdot (bnd + \kappa n) + 1$.

– σ: σ should be sufficiently large so $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ work. Thus, we set $\sigma \geq s_1([\mathbf{R}_{\mathsf{id}}])\|\tilde{\mathbf{T}}_g\|\omega(\sqrt{\log n})$ and $\sigma \geq \|\tilde{\mathbf{B}}\|\omega(\sqrt{\log n})$, thus we must have

$$\sigma \geq \sqrt{\frac{q \cdot e}{2}} \cdot \omega(\sqrt{\log n}) \text{ and } \sigma \geq n^2\sqrt{d\ell}\sqrt{b^2+1}(bnd + \kappa n) \cdot \omega(\sqrt{\log n}).$$

For our parameter set, we have $\sigma \geq \sqrt{\frac{q \cdot e}{2}} \cdot \omega(\sqrt{\log n})$ satisfies both $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ requirement.

– *Error term*: The error term $\mathrm{e}' = \mathrm{e}_1 - \mathbf{e} \cdot \mathbf{s}^\intercal$ in $\mathsf{Decryption}$ algorithm has the bound

$$|\phi(\mathbf{e}) \cdot [\mathsf{rot}(\mathbf{s}^\intercal)]_j| \leq \|[\mathsf{rot}(\mathbf{s}^\intercal)]_j\|\alpha\omega(\sqrt{\log n}) \approx d^2\ell n^6\sqrt{q} \cdot \omega(\log n)$$

$$|[\phi(\mathrm{e}_5 - \mathbf{e} \cdot \mathbf{s}^\intercal)]_j| \leq d^2\ell n^6\sqrt{q} \cdot \omega(\log n) + n^{\frac{3}{2}}\omega(\log^{\frac{9}{4}} n).$$

To guarantee the correctness, we should have

$$d^2\ell n^6\sqrt{q} \cdot \omega(\log n) + n^{\frac{3}{2}}\omega(\log^{\frac{9}{4}} n) < \frac{q}{5}$$

4.2 Security Proof

Theorem 5 *The NTRU IBE scheme presented above is IND-CPA adaptively secure in the standard model assuming the underlying* $r\mathsf{PHF}$ *is* $(1, Q, \beta, \varrho, \delta) - r\mathsf{PHF}$*, NTRU assumption holds, and* $\mathsf{dRLWE}_{n,q,\alpha'}$ *is hard.*

Proof. The proof proceeds in a sequence of games as follows:

Game_0 : This game is identical to the original security game. At the end of the game, $\mathcal{A}$ outputs coin^*. The challenger $\mathcal{C}$ sets $\mathsf{coin}' = \mathsf{coin}^*$. We have

$$\left|\Pr[X_0] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}' = \mathsf{coin}] - \frac{1}{2}\right| = \left|\Pr[\mathsf{coin}^* = \mathsf{coin}] - \frac{1}{2}\right| = \epsilon_0$$

Game_1 : In Game_1, Challenger $\mathcal{C}$ performs exactly the same as in Game_0, except that $\mathcal{C}$ executes an abort event that is independent of the adversaries view. At the beginning of the game, $\mathcal{C}$ runs $\mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{h}, \mathbf{g}_b) \rightarrow (\mathsf{K}', \mathsf{td})$. Due to NTRU assumption, h is indistinguishable with an uniformly random ring elements. $\mathcal{C}$ will keep td to itself. For identity id, $\mathcal{C}$ sets a function $F_y(\mathsf{id}) = \mathrm{y}_x$ where $(\mathbf{r}_x, \mathbf{e}_x, \mathrm{y}_x) \leftarrow \mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}', \mathsf{id})$. For a sequence of identities $\mathsf{ID} = (\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$ where id^* represents the challenge identity and $\mathsf{id}_1, \cdots .\mathsf{id}_Q$ are the identities queried by $\mathcal{A}$ key extraction queries, we define

$$\gamma(\mathsf{ID}) = \Pr_{\mathbf{y}}[F_y(\mathsf{id}^*) = 0 \wedge F_y(\mathsf{id}_1) \in \mathcal{I} \wedge \cdots \wedge F_y(\mathsf{id}_Q) \in \mathcal{I}],$$

where the probability is taken over $\mathbf{y}$ sampled in $\mathcal{H}.\mathsf{TrapGen}$. For $\mathsf{ID} = (\mathsf{id}^*, \mathsf{id}_1, \cdots, \mathsf{id}_Q)$ such that $\mathsf{id}^* \neq \mathsf{id}_i$ for $i = 1, \cdots, Q$, we define $\gamma_{\min}$ and $\gamma_{\max}$ as the lower and upper bound of $\gamma(\mathsf{ID})$ taken over all such ID and thus we have $\gamma(\mathsf{ID}) \in [\gamma_{\min}, \gamma_{\max}]$.

We have Game_1 running as follows:

- *Setup.* Challenger $\mathcal{C}$ runs $\mathsf{KeySetup}(1^\lambda) \rightarrow (\mathsf{mpk}, \mathsf{msk})$ and sends mpk to adversary $\mathcal{A}$.
- *Key Extraction Query.* and *Challenge.* phases are exactly identical as in Game_0.
- *Output.* In the final guess phase, $\mathcal{A}$ output a coin $\mathsf{coin}^* \in \{0, 1\}$. The challenger $\mathcal{C}$ now does the following:
  Abort Check. $\mathcal{C}$ checks that whether the following condition holds:

$$\tau(\boldsymbol{y}, \mathsf{ID}) = (F_y(\mathsf{id}^*) = 0 \wedge F_y(\mathsf{id}_1) \in \mathcal{I} \wedge \cdots \wedge F_y(\mathsf{id}_Q) \in \mathcal{I}) \tag{3}$$

  Where id^* is the challenge identity and id_i is the identity for the ith key extraction query. If condition (3) does not hold, the $\mathcal{C}$ sets $\mathsf{coin}' \xleftarrow{\$} \{0, 1\}$. Otherwise, $\mathcal{C}$ performs identically as in Game_0 and sets $\mathsf{coin}' = \mathsf{coin}^*$.
  Artificial Abort. The artificial abort proceeds the same as in [24]. The simulator samples enough times the probability $\gamma(\mathsf{ID})$ by choosing random $\boldsymbol{y}$ and evaluating $\tau(\boldsymbol{y}, \mathsf{ID}')$ to compute an estimate $\gamma'(\mathsf{ID})$. Then if $\gamma'(\mathsf{ID}) \geq \gamma_{\min}$ the simulator will abort with probability $\frac{\gamma'(\mathsf{ID}) - \gamma_{\min}}{\gamma'(\mathsf{ID})}$ and take a random guess $\mathsf{coin}' \xleftarrow{\$} \{0, 1\}$. Otherwise, the simulator will not abort.

We use W_i to describe the event that $\mathsf{coin}' = \mathsf{coin}$ in Game_i. According to Lemma 12, we have

$$\left| \Pr[X_1] - \frac{1}{2} \right| \geq \frac{3}{4} \gamma_{\min} \cdot \epsilon_0.$$

Since we are using $(1, Q, \beta, \varrho, \delta)r\mathsf{PHF}$, we have $\gamma_{\min} = \delta$.

Game_2 : In Game_2, we modify the way we generating the hash key K in mpk. During the setup phase Game_2, $\mathcal{C}$ returns $\mathsf{mpk} = (\mathsf{K}', \mathrm{h})$ where K' is the output of $\mathcal{H}.\mathsf{TrapGen}$ algorithm. The rest of Game_2 is identical to Game_1.

From the *computational close key* property, we have the distributions of K' and K $\mathsf{negl}(n)$-close. Therefore, we have $|\Pr[X_1] - \Pr[X_2]| = \mathsf{negl}(n)$.

Game_3*:* In this game, we change how h are chosen in Game_2. In Game_3, $\mathcal{C}$ generates $\mathrm{h} \xleftarrow{\$} \mathcal{R}_q$. NTRU assumption guarantees that the distribution of h in Game_2 and Game_3 is indistinguishable. Same as in Game 1 and 2, $\mathcal{C}$ has the trapdoor td for hash key K'. We use $\mathbf{T}_g$ to represent the basis of gadget vector $\mathbf{g}_b$.

Instead of using the trapdoor for $[1, \mathrm{h}]$, $\mathcal{C}$ now uses $\mathbf{T}_g$ to answer the key extraction queries. To respond a key extraction query for $\mathsf{id} = (\mathsf{b}_1, \cdots, \mathsf{b}_k) \in \{0,1\}^\ell$, $\mathcal{C}$ first constructs:

$$\mathbf{f}_{\mathsf{id}} = [1, \mathrm{h}, \mathbf{h}_{\mathsf{id}}] \in \mathcal{R}_q^{2+d}$$

where $\mathbf{h}_{\mathsf{id}} = \mathcal{H}.\mathsf{Eval}(\mathsf{K}', \mathsf{id}) = \mathrm{h} \cdot \mathbf{r}_{\mathsf{id}} + \mathbf{e}_{\mathsf{id}} + F_y(\mathsf{id}) \cdot \mathbf{g}_b$.

Then, $\mathcal{C}$ needs to find short $\mathbf{s} \in \mathcal{R}_q^{2+d}$ such that $\mathbf{f}_{\mathsf{id}} \cdot \mathbf{s}^\intercal = \mathrm{u}$.

For $\mathbf{r}_{\mathsf{id}} = [\mathrm{r}_{\mathsf{id}}^{(0)}, \cdots, \mathrm{r}_{\mathsf{id}}^{(d-1)}]$, $\mathbf{e}_{\mathsf{id}} = [\mathrm{e}_{\mathsf{id}}^{(0)}, \cdots, \mathrm{e}_{\mathsf{id}}^{(d-1)}]$, $\mathbf{R}_{\mathsf{id}} = \begin{bmatrix} \mathrm{e}_{\mathsf{id}}^{(0)} & \cdots & \mathrm{e}_{\mathsf{id}}^{(d-1)} \\ \mathrm{r}_{\mathsf{id}}^{(0)} & \cdots & \mathrm{r}_{\mathsf{id}}^{(d-1)} \end{bmatrix} \in \mathcal{R}_q^{2\times d}$, $\mathbf{g}_b \in \mathcal{R}_q^d$, we have $\mathbf{h}_{\mathsf{id}} = [1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r}_{\mathsf{id}} + \mathbf{e}_{\mathsf{id}} + F_y(\mathsf{id}) \cdot \mathbf{g}_b]$.

The challenger $\mathcal{C}$ now does the following:

1. If $F_y(\mathsf{id}) \notin \mathcal{I}$, abort the game and set $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$ as in Game_2;
2. Set $\mathbf{s} \leftarrow \mathsf{SampleRight}(\mathbf{h}, F_y(\mathsf{id}) \cdot \mathbf{g}_b, \mathbf{r}_{\mathsf{id}}, \mathbf{e}_{\mathsf{id}}, \mathbf{T}_g, \mathrm{u}, \sigma) \in \mathcal{R}_q^{2+d}$;
3. Send $\mathsf{sk}_{\mathsf{id}} \leftarrow \mathbf{s}$ to $\mathcal{A}$.

Since $\mathbf{T}_g$ is a basis for $\mathbf{g}_b$. According the property of algorithm $\mathsf{SampleRight}$ with $\sigma \geq s_1(\mathbf{R}_{\mathsf{id}}) \|\tilde{\mathbf{T}}_g\| \omega(\sqrt{\log n})$, the generated $\mathbf{s}$ has distribution $\mathsf{negl}(n)$-close to $\mathcal{D}_{\Lambda_{\mathrm{u}}^\perp(\mathbf{f}_{\mathsf{id}}), \sigma}$, which is the same as in Game_2. Thus, we have $\Pr[X_2] = \Pr[X_3]$.

Game_4*:* In this game, $\mathcal{C}$ changes the way to generate challenge ciphertext. In the challenge phase of Game_4, for identity id^* and a message m, $\mathcal{C}$ samples $\mathrm{v} \xleftarrow{\$} \mathcal{R}_q$ with an inverse, $\mathrm{t} \xleftarrow{\$} \mathcal{R}_q$, and $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}^2, \alpha'}$, $\mathbf{e}_m \leftarrow \mathcal{D}_{\mathcal{R}, \alpha'}$. $\mathcal{C}$ then computes

$$\mathbf{c} = [\mathrm{v}, \mathrm{v} \cdot \mathrm{h}] \cdot \mathrm{t} + \mathbf{e} \in \mathcal{R}_q^2, \ \mathbf{R}_{\mathsf{id}} = \begin{bmatrix} \mathrm{e}_{\mathsf{id}}^{(0)} & \cdots & \mathrm{e}_{\mathsf{id}}^{(d-1)} \\ \mathrm{r}_{\mathsf{id}}^{(0)} & \cdots & \mathrm{r}_{\mathsf{id}}^{(d-1)} \end{bmatrix} \in \mathcal{R}_q^{2\times d}$$

$$\mathsf{ReRand}\left(\mathsf{rot}_n([\mathbf{I}_2 | \mathbf{R}_{\mathsf{id}^*}]), \phi(\mathbf{c}), \alpha', \frac{\alpha}{2\alpha'}\right) \to \boldsymbol{c}'$$

and $\boldsymbol{c}' = \phi(\mathbf{c}) \in \mathbb{Z}_q^{(2+d)n}$. $\mathcal{C}$ sets the challenge ciphertext c^* as

$$c^* = \left(\mathbf{c}', \mathrm{c}_m = (\mathrm{v} \cdot \mathrm{u}) \cdot \mathrm{t} + \mathrm{e}_m + \lfloor \frac{q}{2} \rfloor \mathrm{m}_{\mathsf{coin}}\right).$$

We claim that this change alters the view of $\mathcal{A}$ only negligibly. Observe that

$$\phi(\mathbf{c}) = \phi([\mathrm{v}, \mathrm{v} \cdot \mathrm{h}] \cdot \mathrm{t}) + \phi(\mathbf{e})$$

where $\phi(\mathbf{e})$ is distributed according to $\mathcal{D}_{\mathbb{Z}^{2n}, \alpha'}$. After applying the ReRand algorithm, according to Lemma 4, the output $\boldsymbol{c} \in \mathbb{Z}_q^{(2+d)n}$ is

$$\begin{aligned} \boldsymbol{c} =& \phi([\mathrm{v}, \mathrm{v} \cdot \mathrm{h}] \cdot \mathrm{t}) \cdot \mathsf{rot}_n([\mathbf{I}_2 | \mathbf{R}_{\mathsf{id}^*}]) + \phi(\mathbf{e}') \\ =& \phi([\mathrm{v}, \mathrm{v} \cdot \mathrm{h}, \mathrm{v} \cdot \mathrm{h}_{\mathsf{id}^*}^{(0)}, \mathrm{v} \cdot \mathrm{h}_{\mathsf{id}^*}^{(1)}] \cdot \mathrm{t}) + \phi(\mathbf{e}') \end{aligned}$$

where $\mathbf{e}' \in \mathcal{R}_q^{2+d}$ is distributed according to $\mathcal{D}_\alpha^{(2+d)n}$. Since for the challenge identity, id^*, we have $F_y(\mathsf{id}^*) = 0$ and $\mathbf{h}_{\mathsf{id}^*} = \mathbf{e}_{\mathsf{id}^*} + \mathrm{h}\cdot \mathbf{r}_{\mathsf{id}^*}$ for $(\mathbf{r}_{\mathsf{id}^*}, \mathbf{e}_{\mathsf{id}^*}, y_{\mathsf{id}^*}) \leftarrow \mathcal{H}.\mathsf{TrapEval}(\mathsf{td}, \mathsf{K}', \mathsf{id}^*)$. Thus we have the distribution of $\mathbf{c}^*$ is statistically close to that in Game_3 and $|\Pr[X_3] - \Pr[X_4]| = \mathsf{negl}(n)$.

Game_5*:* In this game, challenger $\mathcal{C}$ changes the way to generate challenge ciphertext. In the challenge phase of Game_5, $\mathcal{C}$ picks $\mathrm{c}_m \xleftarrow{\$} \mathcal{R}_q$, $\mathbf{c}' \xleftarrow{\$} \mathcal{R}_q^2$, $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}^2,\alpha'}$ and runs $\mathsf{ReRand}\left(\mathsf{rot}_n([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}]), \phi(\mathbf{c}), \alpha', \frac{\alpha}{2\alpha'}\right) \rightarrow \phi(\mathbf{c}^*)$, where $\mathbf{c} = \mathbf{c}' + \mathbf{e}$, $\mathbf{R}_{\mathsf{id}} = [\mathbf{e}_{\mathsf{id}^*}; \mathbf{r}_{\mathsf{id}^*}]$. $\mathcal{C}$ sets the challenge ciphertext c^* as $c^* = (\mathbf{c}^*, \mathrm{c}_m)$. As we show in Lemma 13, if $\mathsf{dRLWE}_{n,q,\alpha'}$ is hard, we have $|\Pr[X_4] - \Pr[X_5]| = \mathsf{negl}(n)$.

Analysis. Combining Game_1 to Game_5, we obtain that

$$\begin{aligned}\left|\Pr[X_5] - \frac{1}{2}\right| &= \left|\Pr[X_1] - \frac{1}{2} + \sum_{i=1}^{4}(\Pr[X_{i+1}] - \Pr[X_i])\right| \\ &\geq \left|\Pr[X_1] - \frac{1}{2}\right| - \sum_{i=1}^{4}|\Pr[X_{i+1}] - \Pr[X_i]| \geq \frac{1}{2}\delta\cdot\epsilon_0 - \mathsf{negl}(n)\end{aligned}$$

Since in Game_5, the challenge ciphertext c^* is independent from the value of coin. We have $\left|\Pr[X_5] - \frac{1}{2}\right| = \mathsf{negl}(n)$. Therefore, we have $\epsilon_0 = \mathsf{negl}(n)$.

Lemma 12. *For any PPT adversary $\mathcal{A}$, we have*

$$\left|\Pr[X_1] - \frac{1}{2}\right| \geq \frac{3}{4}\gamma_{\min}\left|\Pr[X_0] - \frac{1}{2}\right| = \frac{3}{4}\gamma_{\min}\cdot\epsilon_0.$$

Proof. The probability of event X_1 ($\mathsf{coin}' = \mathsf{coin}$) happens in Game_1 is analyze as following

$$\begin{aligned}&\Pr[X_1] \\ =&\Pr[\mathsf{abort} \wedge \mathsf{coin}' = \mathsf{coin}] + \Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}' = \mathsf{coin}] \\ =&\Pr[\mathsf{coin}' = \mathsf{coin}|\mathsf{abort}]\Pr[\mathsf{abort}] + \Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}' = \mathsf{coin}] \\ =&\frac{1}{2}\Pr[\mathsf{abort}] + \Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}^* = \mathsf{coin}] \\ =&\left(\frac{1}{2} - \frac{1}{2}\Pr[\overline{\mathsf{abort}}]\right) + \Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}^* = \mathsf{coin}] \\ =&\frac{1}{2} + \frac{1}{2}\Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}^* = \mathsf{coin}] - \frac{1}{2}\Pr[\overline{\mathsf{abort}} \wedge \mathsf{coin}^* \neq \mathsf{coin}] \\ =&\frac{1}{2} + \frac{1}{2}\Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* = \mathsf{coin}]\Pr[\mathsf{coin}^* = \mathsf{coin}] - \frac{1}{2}\cdot \\ &\Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* \neq \mathsf{coin}]\Pr[\mathsf{coin}^* \neq \mathsf{coin}] \\ =&\frac{1}{2} + \frac{1}{2}\Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* = \mathsf{coin}](\frac{1}{2} + \epsilon_0) - \\ &\frac{1}{2}\Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* \neq \mathsf{coin}](\frac{1}{2} - \epsilon_0) \qquad (4)\end{aligned}$$

The analysis of the artificial abort is the same as in [24] and from it we have

$$\Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* = \mathsf{coin}](\frac{1}{2} + \epsilon_0) - \Pr[\overline{\mathsf{abort}}|\mathsf{coin}^* \neq \mathsf{coin}](\frac{1}{2} - \epsilon_0) \geq \frac{3}{2}\gamma_{\min} \cdot \epsilon_0 \quad (5)$$

Thus, combining (4) and (5) we have $\left|\Pr[X_1] - \frac{1}{2}\right| \geq \frac{3}{4}\gamma_{\min} \cdot \epsilon_0$.

Lemma 13. *With the* $\mathsf{RLWE}_{n,q,\alpha'}$ *assumption, we have* $|\Pr[X_4] - \Pr[X_5]| = \mathsf{negl}(n)$.

Proof. Assume there is an adversary $\mathcal{A}$ who has non-negligible advantage in distinguishing Game_4 and Game_5. $\mathcal{C}$ then can use $\mathcal{A}$ solve $\mathsf{dRLWE}_{n,q,\alpha'}$.

$\mathcal{C}$ receives the dRLWE problem instances $\{(\mathrm{a}_i, \mathrm{u}_i)\}_{i=1}^3 \in (\mathcal{R}_q \times \mathcal{R}_q)^3$. If a_1 is not invertible, $\mathcal{C}$ reject. This is reasonable since when $q = \Omega(n)$, the probability for a uniform element of $\mathcal{R}_q$ being invertible is non-negligible [23].

Without loss of generality, we assume that $\mathrm{u}_i = \mathrm{u}'_i + \mathrm{e}'_i$ for $\mathrm{e}'_i \leftarrow \mathcal{D}_{\mathcal{R},\alpha'}$. $\mathcal{C}$ then runs the game as follow.

- *Setup.* Challenger $\mathcal{C}$ sets

  $$\mathrm{u} := \mathrm{a}_1^{-1} \cdot \mathrm{a}_3, \mathrm{h} := \mathrm{a}_1^{-1} \cdot \mathrm{a}_2, \hat{\mathbf{h}} := [1, \mathrm{h}]$$

  where a_1^{-1} is the inverse of a_1. It also generates $\mathsf{K}' = (\mathbf{h}_0, \{\mathbf{h}_{i,j}\}_{(i,j)\in[2]\times[t]})$ as described in Game_3. $\mathcal{C}$ returns $\mathsf{mpk} = (\mathrm{h}, \mathsf{K}', \mathrm{u})$.
- *Key Extraction Query.* this phases are exactly identical as in Game_4 including aborting the game.
- *Challenge.* Adversary $\mathcal{A}$ chooses an identity id^* on which it wishes to be challenged. $\mathcal{C}$ samples $\mathsf{coin} \xleftarrow{\$} \{0,1\}$ and computes $F_y(\mathsf{id}^*)$. If $F_y(\mathsf{id}^*) \neq 0$, $\mathcal{C}$ aborts and sets $\mathsf{coin}' \xleftarrow{\$} \{0,1\}$. Otherwise, $\mathcal{C}$ sets the challenge ciphertext C^* as follow

  $$\mathsf{ReRand}\left(\mathsf{rot}_n([\mathbf{I}_2|\mathbf{R}_{\mathsf{id}^*}]), \phi([\mathrm{u}_1, \mathrm{u}_2]), \alpha', \frac{\alpha}{2\alpha'}\right)$$

  $$\downarrow$$

  $$\phi(\mathbf{c}^*) \in \mathbb{Z}_q^{(2+d)n}$$

  $$C^* = \left(\mathbf{c}^*, \mathrm{c}_m = \mathrm{u}_3 + \lfloor\frac{q}{2}\rfloor \mathrm{m}_{\mathsf{coin}}\right).$$

  Here $\boldsymbol{v}$ will be implicitly setted as a_1. $\mathcal{C}$ returns C^* to $\mathcal{A}$.
- *Output.* In the final guess phase, $\mathcal{A}$ output a coin $\mathsf{coin}^* \in \{0,1\}$. The challenger $\mathcal{C}$ sets $\mathsf{coin}' = \mathsf{coin}$. $\mathcal{C}$ outputs 1 if $\mathsf{coin}' = \mathsf{coin}$. Otherwise, output 0.

If $\{\mathrm{a}_i, \mathrm{u}_i\}_{i\in[3]}$ are valid RLWE samples such that $\mathrm{u}_i = \mathrm{a}_i \cdot \mathrm{s} + \mathrm{e}_i$ for some $\mathrm{s} \in \mathcal{R}_q$, $\mathcal{C}$ perfect simulates the view of $\mathcal{A}$ in Game_4. If u_i are sampled uniformly random from $\mathcal{R}_q$, $\mathcal{C}$ perfect simulates the view of $\mathcal{A}$ in Game_5. Thus we have

$$|\Pr[X_4] - \Pr[X_5]| = \mathsf{adv}_{\mathcal{C}}^{\mathsf{dRLWE}_{n,q,\alpha'}}.$$

5 Short Signature from rPHF

In this section, we are going to present the construction of our signature scheme based on our rPHF $\mathcal{H}$ =($\mathcal{H}$.Gen, $\mathcal{H}$.Eval, $\mathcal{H}$.TrapGen, $\mathcal{H}$.TrapEval) proposed in Section 3.2.
$\mathsf{Keygen}(1^\ell) \to (\mathsf{vk}, \mathsf{sk})$: On input the security parameter 1^λ, this algorithm runs as follow:

- Generate $(\mathrm{h}, \mathbf{B}_\mathrm{h}) \leftarrow \mathsf{GenBasis}(1^\lambda)$;
- Generate $\mathsf{K} \leftarrow \mathcal{H}.\mathsf{Gen}(1^\lambda)$
- Choose $\mathrm{u} \xleftarrow{\$} \mathcal{R}_q$;
- Output the verification key vk and signing key msk,

$$\mathsf{vk} = (\mathrm{h}, \mathsf{K}, \mathrm{u}) \ , \ \mathsf{sk} = \mathbf{B}_\mathrm{h}.$$

In the $\mathsf{KeySetup}$ algorithm, $\mathbf{B}_\mathrm{h}$ is a short basis of $\Lambda_{\mathrm{h},q}$. It allows us to sample $(\mathrm{s}_1, \mathrm{s}_2)$, where $\mathrm{s}_1 + \mathrm{h} \cdot \mathrm{s}_2 = \mathrm{t} \mod q$ for a given t.

$\mathsf{Sign}(M, \mathsf{vk}, \mathsf{sk}) \to \sigma$: Given message $M \in \{0,1\}^\ell$, verification key $\mathsf{vk} = (\mathrm{h}, \mathsf{K}, \mathrm{u})$, and signing key $\mathsf{sk} = \mathbf{B}_\mathrm{h}$, this algorithm runs as follow

- Run $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, M) \to \mathbf{z} \in \mathcal{R}_q^d$;
- Let $\mathbf{h} = [1, \mathrm{h}]$, sample a short vector $\mathbf{s} \leftarrow \mathsf{SampleLeft}(\mathbf{h}, \mathbf{z}, \mathbf{B}_\mathrm{h}, \mathrm{u}, \sigma)$;
- Return signature $\mathbf{s}$.

$\mathsf{Verify}(\mathbf{s}, M, \mathsf{vk}) \to 0/1$: Given signature $\mathbf{s} \in \mathcal{R}_q^{2+d}$, message M, and verification key $\mathsf{vk} = (\mathrm{h}, \mathsf{K}, \mathrm{u})$, this algorithm then checks for $\mathbf{h} = [1, \mathrm{h}]$, $\mathcal{H}.\mathsf{Eval}(\mathsf{K}, M) \to \boldsymbol{z} \in \mathcal{R}_q^d$, whether $[\mathbf{h}, \mathbf{z}] \cdot \mathbf{s}^\intercal = \mathrm{u}$ and $\|\mathbf{s}\| \leq \varphi$ holds or not. If it holds, output 1; otherwise, output 0.

Correctness and Parameter Selection for Type-I rPHF

- n, q: n is a power of 2, q is a prime and satisfies $q \equiv 1 \mod 2n$.
- β: According to Theorem 1, we set $\beta = n^{\frac{1}{4}} \omega(\log^{\frac{1}{4}} n)$.
- $\varphi = \sigma\sqrt{4n}$.
- σ: should be sufficiently large to allow $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ work. Thus, According to Lemma 1, 7, we set $\sigma \geq s_1(\mathbf{R}) \cdot \|\tilde{\mathbf{B}}\| \cdot \omega(\sqrt{\log n})$ for $\mathbf{B}$ the NTRU basis, $s_1(\mathbf{R}) \leq \ell \cdot \beta\sqrt{n} \cdot (2\sqrt{2} + \omega(\sqrt{\log n}))$, where $\mathbf{R} = \begin{bmatrix} \mathrm{e}_m^{(0)} & \mathrm{e}_m^{(1)} \\ \mathrm{r}_m^{(0)} & \mathrm{r}_m^{(1)} \end{bmatrix}$.

Correctness and Parameter Selection for Type-II $r\mathsf{PHF}$

- n, q: n is a power of 2, q is a prime and satisfies $q \cong 3 \mod 8$.
- β: According to Theorem 2, we set $\beta = n^{\frac{3}{2}}\omega(\log^{\frac{9}{4}} n)$.
- b: We set $b = n$ and thus we have $d = \log_b q$
- $\varphi = \sigma\sqrt{(d+2)n}$.
- σ: should be sufficiently large to allow $\mathsf{SampleLeft}$ and $\mathsf{SampleRight}$ work. Thus, According to Lemma 1,6,7, we set $\sigma \geq \sqrt{\frac{q \cdot e}{2}} \cdot \omega(\sqrt{\log n})$, and $\sigma \geq s_1(\mathbf{R}) \cdot \|\tilde{\mathbf{T}}_g\| \cdot \omega(\sqrt{\log n})$ for $s_1(\mathbf{R}) \approx n^2\sqrt{d}\ell \cdot (bnd + \kappa n)$, where $\mathbf{R} = \begin{bmatrix} \mathbf{e}_m \\ \mathbf{r}_m \end{bmatrix}$.

We set $s_1(\mathbf{R})\|\tilde{\mathbf{T}}_g\| \cdot \omega(\sqrt{\log n}) \approx n^2\sqrt{d}\ell \cdot (bnd + \kappa n) \cdot \sqrt{b^2+1} \approx \sqrt{\frac{q \cdot e}{2}}$ and get $q \approx \ell^2 n^{10}$. For $\ell = n$, we have $d = 12$.

5.1 Security Proof

Theorem 6 *The signature scheme presented above is existentially unforgeable in the standard model assuming the underlying $r\mathsf{PHF}$ is $(1, Q, \alpha, \varrho, \delta) - r\mathsf{PHF}$, NTRU assumption holds, and $\mathsf{dRLWE}_{n,q,\beta}$ problem are hard.*

Proof. If there exists an adversary $\mathcal{A}$ who can break the existentially unforgeability of our signature scheme, we then can construct a simulator $\mathcal{S}$ who breaks the NTRU problem.
Key Generation. Given instance $(\mathrm{h}, \mathrm{u}) \in \mathcal{R}_q \times \mathcal{R}_q$, simulator sets $\mathbf{h} = [1, \mathrm{h}]$. $\mathcal{S}$ then generates $(\mathsf{K}, \mathsf{td}) \leftarrow \mathcal{H}.\mathsf{TrapGen}(1^\lambda, \mathrm{h}, \mathbf{g}_b)$. According the property of $r\mathsf{PHF}$, trapdoor mode key K will be computationally indistinguishable with normal mode hash key.
Signing Oracle. On given a message M, simulator $\mathcal{S}$ computes $(\mathbf{r}_m, \mathbf{e}_m, y_m) \leftarrow \mathcal{H}.\mathsf{TrapGen}(\mathsf{td}, \mathsf{K}, m)$. If $y_m = 0$, abort. Otherwise, run $\mathsf{SampleRight}(\mathbf{h}, y_m \cdot \mathbf{g}_b, \mathbf{r}_m, \mathbf{e}_m, \mathbf{T}_g, \mathrm{u}, \sigma) \rightarrow \mathbf{s} \in \mathcal{R}_q^{2+d}$.
Forge. Finally, adversary $\mathcal{A}$ will output a message signature pair $(M_f, \mathbf{s}_f)$ as forgery where M_f has not been queried to signing oracle. Simulator $\mathcal{S}$ runs $\mathcal{H}.\mathsf{TrapGen}(\mathsf{td}, \mathsf{K}, M_f) \rightarrow (\mathbf{r}_{m_f}, \mathbf{e}_{m_f}, y_{m_f})$. If $y_{m_f} \neq 0$, $\mathcal{S}$ aborts. Otherwise, $\mathcal{S}$ obtains $\mathbf{s}_{m_f} \in \mathcal{R}_q^{2+d}$ which satisfies

$$[1, \mathrm{h}, \mathrm{h} \cdot \mathbf{r}_{m_f} + \mathbf{e}_{m_f}] \cdot \mathbf{s}_{m_f}^{\intercal} = \mathrm{u}. \tag{6}$$

For $\mathbf{s}_{m_f} = [\mathrm{s}_1, \mathrm{s}_2, \mathbf{s}']$, $\mathcal{S}$ then has

$$(\mathrm{s}_1 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}) + \mathrm{h} \cdot (\mathrm{s}_2 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}) = \mathrm{u}.$$

$\mathcal{S}$ obtains $\mathbf{t} = [\mathrm{s}_1 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}, \mathrm{s}_2 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}]$ as the solution to the NTRU problem (h, u) with probability at least δ. According to the property of our $r\mathsf{PHF}$, we have $\|\mathrm{r}_{m_f}^{(j)}\|$, $\|\mathrm{e}_{m_f}^{(j)}\| \leq \alpha\sqrt{n}$. Thus, we have $\|\mathrm{s}_1 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}\|, \|\mathrm{s}_2 + \mathbf{r}_{m_f} \cdot \mathbf{s}'^{\intercal}\| \leq \sigma\sqrt{n}(1 + dn\alpha)$.

References

1. S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 553–572. Springer, 2010.
2. J. Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions. In J. Katz, editor, *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography, Gaithersburg, MD, USA, March 30 - April 1, 2015, Proceedings*, volume 9020 of *Lecture Notes in Computer Science*, pages 236–255. Springer, 2015.
3. D. Boneh and X. Boyen. Efficient selective-id secure identity-based encryption without random oracles. In C. Cachin and J. Camenisch, editors, *Advances in Cryptology - EUROCRYPT 2004, International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004, Proceedings*, volume 3027 of *Lecture Notes in Computer Science*, pages 223–238. Springer, 2004.
4. D. Boneh and X. Boyen. Secure identity based encryption without random oracles. In M. K. Franklin, editor, *Advances in Cryptology - CRYPTO 2004, 24th Annual International CryptologyConference, Santa Barbara, California, USA, August 15-19, 2004, Proceedings*, volume 3152 of *Lecture Notes in Computer Science*, pages 443–459. Springer, 2004.
5. D. Boneh and M. K. Franklin. Identity-based encryption from the weil pairing. In J. Kilian, editor, *Advances in Cryptology - CRYPTO 2001, 21st Annual International Cryptology Conference, Santa Barbara, California, USA, August 19-23, 2001, Proceedings*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229. Springer, 2001.
6. X. Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In P. Q. Nguyen and D. Pointcheval, editors, *Public Key Cryptography - PKC 2010, 13th International Conference on Practice and Theory in Public Key Cryptography, Paris, France, May 26-28, 2010. Proceedings*, volume 6056 of *Lecture Notes in Computer Science*, pages 499–517. Springer, 2010.
7. R. Canetti, S. Halevi, and J. Katz. A forward-secure public-key encryption scheme. In E. Biham, editor, *Advances in Cryptology - EUROCRYPT 2003, International Conference on the Theory and Applications of Cryptographic Techniques, Warsaw, Poland, May 4-8, 2003, Proceedings*, volume 2656 of *Lecture Notes in Computer Science*, pages 255–271. Springer, 2003.
8. D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 523–552. Springer, 2010.
9. L. Ducas, V. Lyubashevsky, and T. Prest. Efficient identity-based encryption over ntru lattices. In P. Sarkar and T. Iwata, editors, *Advances in Cryptology – ASIACRYPT 2014*, pages 22–41, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.
10. L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. In J. A. Garay and R. Gennaro, editors, *Advances in Cryptology - CRYPTO*

2014 - 34th Annual Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2014, Proceedings, Part I, volume 8616 of *Lecture Notes in Computer Science*, pages 335–352. Springer, 2014.

11. C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, page 197–206, New York, NY, USA, 2008. Association for Computing Machinery.
12. D. Hofheinz and E. Kiltz. Programmable hash functions and their applications. In D. A. Wagner, editor, *Advances in Cryptology - CRYPTO 2008, 28th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 17-21, 2008. Proceedings*, volume 5157 of *Lecture Notes in Computer Science*, pages 21–38. Springer, 2008.
13. T. Jager, R. Kurek, and D. Niehues. Efficient adaptively-secure ib-kems and vrfs via near-collision resistance. In J. A. Garay, editor, *Public-Key Cryptography - PKC 2021 - 24th IACR International Conference on Practice and Theory of Public Key Cryptography, Virtual Event, May 10-13, 2021, Proceedings, Part I*, volume 12710 of *Lecture Notes in Computer Science*, pages 596–626. Springer, 2021.
14. S. Katsumata and S. Yamada. Partitioning via non-linear polynomial functions: More compact ibes from ideal lattices and bilinear maps. In J. H. Cheon and T. Takagi, editors, *Advances in Cryptology - ASIACRYPT 2016 - 22nd International Conference on the Theory and Application of Cryptology and Information Security, Hanoi, Vietnam, December 4-8, 2016, Proceedings, Part II*, volume 10032 of *Lecture Notes in Computer Science*, pages 682–712, 2016.
15. V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. In H. Gilbert, editor, *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.
16. V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-lwe cryptography. In T. Johansson and P. Q. Nguyen, editors, *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, volume 7881 of *Lecture Notes in Computer Science*, pages 35–54. Springer, 2013.
17. S. McCarthy, N. Smyth, and E. O'Sullivan. A practical implementation of identity-based encryption over NTRU lattices. In M. O'Neill, editor, *Cryptography and Coding - 16th IMA International Conference, IMACC 2017, Oxford, UK, December 12-14, 2017, Proceedings*, volume 10655 of *Lecture Notes in Computer Science*, pages 227–246. Springer, 2017.
18. D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In D. Pointcheval and T. Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, pages 700–718, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.
19. D. Micciancio and O. Regev. Worst-case to average-case reductions based on gaussian measures. In *45th Symposium on Foundations of Computer Science (FOCS 2004), 17-19 October 2004, Rome, Italy, Proceedings*, pages 372–381. IEEE Computer Society, 2004.
20. A. Pellet-Mary and D. Stehlé. On the hardness of the NTRU problem. In M. Tibouchi and H. Wang, editors, *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and In-*

formation Security, Singapore, December 6-10, 2021, Proceedings, Part I, volume 13090 of *Lecture Notes in Computer Science*, pages 3–35. Springer, 2021.
21. O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In H. N. Gabow and R. Fagin, editors, *Proceedings of the 37th Annual ACM Symposium on Theory of Computing, Baltimore, MD, USA, May 22-24, 2005*, pages 84–93. ACM, 2005.
22. A. Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and D. Chaum, editors, *Advances in Cryptology, Proceedings of CRYPTO '84, Santa Barbara, California, USA, August 19-22, 1984, Proceedings*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53. Springer, 1984.
23. D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In K. G. Paterson, editor, *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, volume 6632 of *Lecture Notes in Computer Science*, pages 27–47. Springer, 2011.
24. B. Waters. Efficient identity-based encryption without random oracles. In R. Cramer, editor, *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, volume 3494 of *Lecture Notes in Computer Science*, pages 114–127. Springer, 2005.
25. B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In S. Halevi, editor, *Advances in Cryptology - CRYPTO 2009, 29th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2009. Proceedings*, volume 5677 of *Lecture Notes in Computer Science*, pages 619–636. Springer, 2009.
26. S. Yamada. Adaptively secure identity-based encryption from lattices with asymptotically shorter public parameters. In M. Fischlin and J. Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part II*, volume 9666 of *Lecture Notes in Computer Science*, pages 32–62. Springer, 2016.
27. S. Yamada. Asymptotically compact adaptively secure lattice ibes and verifiable random functions via generalized partitioning techniques. In J. Katz and H. Shacham, editors, *Advances in Cryptology - CRYPTO 2017 - 37th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 20-24, 2017, Proceedings, Part III*, volume 10403 of *Lecture Notes in Computer Science*, pages 161–193. Springer, 2017.
28. J. Zhang, Y. Chen, and Z. Zhang. Programmable hash functions from lattices: Short signatures and ibes with small key sizes. In M. Robshaw and J. Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part III*, volume 9816 of *Lecture Notes in Computer Science*, pages 303–332. Springer, 2016.