Gadget-Based iNTRU Lattice Trapdoors

Nicholas Genise[1] and Baiyu Li[2]

[1] SRI International[†],
nicholas.genise@sri.com
[2] University of California, San Diego,
baiyu@cs.ucsd.edu

Abstract. We present two new related families of lattice trapdoors based on the inhomogeneous NTRU problem (iNTRU) defined in Genise et. al [16] (ASIACRYPT 2019). Our constructions are "gadget-based" and offer compact secret keys and preimages and compatibility with existing, efficient preimage sampling algorithms. Our trapdoors can be used as a fundamental building block in lattice-based schemes relying lattice trapdoors. In addition, we implemented our trapdoors using the PALISADE library.

1 Introduction

Lattice-based cryptography provides powerful building blocks used to create a plethora of advanced cryptographic capabilities. For example, lattices yield fully homomorphic encryption [19], fully-homomorphic signatures [21], and attribute-based encryption (ABE) [6]. Further appealing aspects of lattice-based cryptography are its potential post-quantum hardness[‡] and its inherent parallelism.

Many of the listed schemes rely on short integer solution (SIS) lattice-trapdoors and their most efficient implementations are in the ring setting (RSIS). Let R_q (R) be a polynomial ring with coefficients in $\mathbb{Z}_q$ ($\mathbb{Z}$) and $X \subset R^m$ be a set of small-norm polynomials. Then, RSIS trapdoors are summarized as follows: the user generates a (pseudo)random $\mathbf{a} \in R_q^m$ together with a *trapdoor* which allows one to invert the function $f_{\mathbf{a}} : X \to R_q$, defined as $f_{\mathbf{a}}(\mathbf{x}) := \mathbf{a}^t\mathbf{x} \pmod{q}$. A powerful method used to invert this function is discrete Gaussian sampling [20, 28] where the trapdoor is used to sample a discrete, Gaussian-like distribution over all RSIS preimages of an output of the RSIS function, $u \in R_q$.

There are two ways to instantiate the GPV paradigm [20]. The first is to generate an RSIS instance which is statistically close to uniformly random together with a trapdoor. The second, more efficient, way to instantiate the GPV paradigm is to generate a *pseudorandom* RSIS instance whose trapdoor is also the secret used in the pseudorandom generator. Lattice cryptography's prominent sources of pseudorandomness are the (ring-)learning with errors problem

[†] This work was done at Rutgers supported by National Science Foundation grant SaTC-1815562.

[‡] https://csrc.nist.gov/Projects/post-quantum-cryptography/round-2-submissions

(RLWE) [26, 35] and the NTRU learning problem [22, 32]. RSIS trapdoors whose pseudorandomness is based on RLWE were introduced in [28] (MP12). Here the trapdoor is used as a linear transformation and the trapdoor is hidden via Peikert's discrete Gaussian perturbation method [31]. We call these trapdoors "gadget-based" since the trapdoor reduces inverting RSIS on a (pseudo)random matrix to a "gadget" matrix on which RSIS is easy to solve. The study of GPV trapdoors whose pseudorandomness is based on NTRU was initiated in [13]. Here the seed to the NTRU PRG is used to construct a short basis and trapdoor inversion is done via a structured version, [15], of the GPV discrete Gaussian sampler [20].

For the remainder, let *approximate trapdoor sampling* denote a trapdoor scheme where the trapdoor is used to sample a short vector whose RSIS image is close to the input's RSIS image: given $u \in R_q$, sample a short $\mathbf{x} \in R^m$ such that $f_{\mathbf{a}}(\mathbf{x}) \approx u \in R_q$. Approximate trapdoors are used in [13] via the HNF optimization and [7]'s adaptation of MP12 to save a constant fraction of preimage and public key memory.

Contribution. We introduce two similar families of gadget-based RSIS (approximate) trapdoors whose pseudorandomness is based on the inhomogeneous NTRU problem (iNTRU), introduced in [16]. Of the two instantiations presented in this paper, the first trapdoor is a noisy version of an RSIS trapdoor, but with the advantage that it can be almost entirely *parallelized*, down to the ring level. The second is an exact analog to MP12, but with a secret key nearly half the size of MP12 when we fix the parameters n, q, m, and R: ours has just one vector of small polynomials serving as a trapdoor instead of two vectors in MP12. Comparing the two iNTRU trapdoors gives an interesting trade-off: the second can have a smaller modulus making it suitable for use over cyclotomic rings where we can take advantage of the ring structure to sample perturbations as in [15, 17], while the first scheme offers more parallelism at the cost of a noisy verification and larger modulus. The larger modulus implies a ring other than a cyclotomic should be used in order to avoid subfield lattice attacks [2, 23].

From a theoretical point of view, our results show MP12's efficient gadget-based trapdoor framework exists on a family of NTRU lattices. This provides another connection between NTRU and ring-LWE, which have similar cryptographic functionality despite apparent structural differences.

Lastly, we implemented our trapdoors in the PALISADE[§] lattice cryptography library to demonstrate efficiency. The trapdoors presented in this paper offer interesting trade-offs with applicability to a large selection of lattice-based schemes.

Technical Details. Similar to MP12, our RSIS inversions can be broken into two steps: offline (precomputation) sampling of perturbation vectors dependent only on the trapdoor/secret key and online sampling which depends on the RSIS image we are inverting. If we precompute the perturbation vectors, as is

[§] `https://palisade-crypto.org/`

standard, then we can reduce preimage sampling in the first, parallel trapdoor scheme to the following computations (with parallelism listed in parentheses):

- one polynomial addition,
- one G-lattice sample (parallelism n, the ring's dimension),
- one ring polynomial multiplication (parallelism $m = O(\log q)$),
- and one polynomial addition (parallelism $m = O(\log q)$).

Now we briefly discuss the trapdoor mechanism. A matrix $\mathbf{R}$ is a $\mathbf{g}$-trapdoor for the RSIS instance $\mathbf{a}^t$ if $\mathbf{a}^t\mathbf{R} = \mathbf{g}^t$ and $\mathbf{R}$ has small entries [28, Definition 5.2]. The public vector $\mathbf{g}^t$ is called a "gadget" since it is easy to solve RSIS on $\mathbf{g}^t$. The iNTRU problem is to distinguish $\mathbf{a}^t := r^{-1}(\mathbf{g}^t + \mathbf{e}^t)$ from uniformly random where $(r, \mathbf{e}) \leftarrow \chi^{k+1}$ is drawn from a distribution with small entries and $\mathbf{g}^t$ is a gadget matrix, usually taken to be $\mathbf{g}^t := [1, b, b^2, \dots, b^{k-1}] \in R^k$ and $k = \log_b q$. In the first, parallelized version of the trapdoor, the polynomial r is *almost* a $\mathbf{g}$-trapdoor for $\mathbf{a}^t := r^{-1}(\mathbf{g}^t + \mathbf{e}^t)$:

$$r \cdot \mathbf{a}^t = \mathbf{g}^t + \mathbf{e}^t \approx \mathbf{g}^t.$$

Let us proceed naively in an attempt to construct a preimage: given $u \in R_q$, sample a discrete Gaussian $\mathbf{x}$ satisfying $\mathbf{g}^t\mathbf{x} = u$ and return $\mathbf{y} := r\mathbf{x}$. Clearly, $\mathbf{y}$ leaks information about r, but the verification equation $u - \mathbf{a}^t\mathbf{y} = \mathbf{e}^t\mathbf{x}$ leaks information about $\mathbf{e}$ in the *range* of the RSIS function. This implies we need to add a perturbation in the domain *and* the range of the RSIS function, unlike previous convolution based trapdoors which only perturb in the RSIS domain. The second trapdoor scheme is given by correcting the error term $\mathbf{e}^t$. This is realized by using $\left[\begin{smallmatrix} -\mathbf{e}^t \\ r\mathbf{I} \end{smallmatrix}\right]$ as a $\mathbf{g}$-trapdoor for $[1, \mathbf{a}] = [1, r^{-1}(\mathbf{g}^t + \mathbf{e}^t)]$.

In terms of applications, we describe the simple hash-and-sign digital signature scheme provided by the former trapdoor scheme and prove its security in the random oracle model (ROM) in the appendix of the full (eprint) version[¶]. In the ROM, we have that the message is hashed to an element in R_q and this hash is modeled as uniformly random, $u \leftarrow H(m) = \mathcal{U}(R_q)$. Other applications, like identity-based encryption, attribute-based encryption, and others follow from the fact that the trapdoor is statistically hidden.

Both trapdoor schemes can be instantiated with the approximate gadget/trapdoor introduced in [7]. The approximate trapdoor increases the verification error while lowering the memory of the scheme and keeping the same concrete security of the underlying SIS problem. This is in contrast to the large base gadget $\mathbf{g} = [1, \sqrt{q}]$ which decreases the concrete hardness of the trapdoor's underlying SIS problem since the shortness needed to break the SIS problem scales with the gadget base in MP12 trapdoors. Further, [7, Section 1.3] give parameter sets incompatible with $\mathbf{g} = [1, \sqrt{q}]$.

Comparison to other SIS signatures and trapdoors. The most natural comparison is Falcon [13][‖], since it is an efficient NTRU-based, discrete Gaussian, hash-and-sign digital signature scheme. Falcon reports a signing time of .16 milliseconds

[¶] `https://eprint.iacr.org/`

[‖] `https://falcon-sign.info/`

for dimension $n = 512$ and .36 milliseconds when $n = 1024$ on an Intel i5 CPU. In contrast, for $n = 512$ we match Falcon's online sampling *with preprocessing* and less classical concrete security (roughly 100 bits versus 130 bits) and we have slower online signing times by an order of magnitude *with preprocessing* for $n = 1024$. In addition, our computations were done on a faster CPU (Intel i7). We expect our online sampling to be an order of magnitude slower in dimension $n = 512$ due to this difference in CPU (four cores versus eight cores). Our experiments are reported in Table 1. We emphasize that the larger modulus in our trapdoors is significant since it lowers the concrete security *and* increases the storage costs. On the other hand, Falcon's signing algorithm uses the power of two cyclotomic subfield structure whereas the trapdoor schemes presented here only use the subfield structure during their offline phase. Our iNTRU trapdoors have an efficient online sampling in any ring with efficient polynomial multiplication, like the NTRUPrime ring [5], $R = \mathbb{Z}[x]/(x^p - x - 1)$ whose lack of subfields resists the overstretched subfield attack [2] (yet still susceptible to [23] for a large modulus). In addition, our scheme is compatible with advanced protocols like ABE [6] and fully-homomorphic signatures [21] since the SIS matrix has the same length as some underlying gadget matrix whereas Falcon's is not. Falcon has much smaller public keys (two ring elements) and signature sizes (one ring element with small coefficients). It should be noted that the recent ModFalcon [10, 9] scheme is compatible with these advanced schemes due to its longer length and retains its efficient sampling (12ms in [9]).

Another comparison is MP12 [28] and its approximate trapdoor version [7]. These schemes' online sampling phase involves a matrix-vector multiplication whereas our first trapdoor's online sampling can be parallelized to one ring multiplication and gadget-lattice sample. Our first trapdoor's offline sampling involves a diagonal covariance over R, so it can be parallelized to the ring-level as well. The second trapdoor is a direct NTRU analog of MP12 but with nearly half the memory for the secret key. Conversely, the use of NTRU lattices invites the dense sublattice attack of [23] and has a lower concrete security in comparison with RLWE-based trapdoors as the modulus q increases for a fixed ring dimension n.

We performed experiments to compare the performance of our trapdoor schemes with [7]. The offline and online running times as well as the key sizes are collected in our experiments, summarized in Table 1. The parameter set in the first column of Table 1 is a toy example for our schemes as by our estimation it provides only 40 bits of security, while the rest of the parameters provide at least 100 bits of security. These experimental results show our schemes have similar online and offline running times comparing to [7], and our secret signing keys are smaller. In Section 4 we explain concrete security estimations and we also provide more detailed discussion about performance comparisons with [7].

2 Preliminaries

Let $\mathbb{N}, \mathbb{Z}, \mathbb{R}, \mathbb{C}$ denote the natural, integer, real, and complex numbers respectively. We denote numbers as lowercase plain (possibly Greek) letters, vectors

Params	Exact	Exact	Exact	Approx	Exact	Approx	Exact	Approx	Exact
n	512	512	1024	1024	1024	1024	1024	1024	8192
$\lceil \log_2 q \rceil$	13	20	20	20	27	27	30	30	30
b	4	4	4	4	4	4	8	8	32
j	0	0	0	3	0	3	0	4	0
σ	10	32	32	32	107	107	181	181	181
Alg 2, offline (ms)	-	-	-	-	19	16	16	12	105
Alg 2, online (ms)	-	-	-	-	2	2	2	2	12
Alg 4, offline (ms)	6	6	20	13	21	18	18	13	113
Alg 4, online (ms)	0.12	0.16	2	2	2	2	2	2	14
[7], offline (ms)	6	6	16	14	19	15	15	12	113
[7], online (ms)	1	1	2	2	3	3	2	2	12
Alg 2 & 4, |sk| (KB)	7	7	32	24	53	42	47	31	248
Alg 2 & 4, |vk| (KB)	7	7	29	21	49	39	43	28	217
[7], |sk| (KB)	13	13	58	42	98	77	86	55	434
[7], |vk| (KB)	8	8	35	27	56	46	51	35	279

Table 1. Summary of online preimage sampling times and key sizes of the hash-and-sign signature schemes based our trapdoors versus the RLWE based trapdoor of [7]. For our schemes we set $\sigma = q^{1/4}$, while for [7] we set $\sigma = 8$. The running times shown are the mode of the data collected from 100 runs for each parameter set. For some parameters, the verification error in Algorithm 2 exceeds $q/2$, and hence we discard its statistics and denote using "-".

as lowercase boldface letters (e.g. $\mathbf{u}, \mathbf{v} \in \mathbb{R}^n$), and matrices as boldface capital letters (e.g. $\mathbf{T}, \mathbf{M} \in \mathbb{Z}^{n \times n}$). Complex conjugation is denoted as $\overline{z}$ for $z \in \mathbb{C}$ and matrix transpose is denoted as $\mathbf{M}^t$. The symbol Σ will be reserved for positive-(semi)definite matrices. We will often use the integers modulo q for some integer $q \geq 2$, $\mathbb{Z}_q := \mathbb{Z}/q\mathbb{Z}$. Denote the logarithm base $b \geq 2$ as $\log_b$, and denote log as logarithm base two when b is omitted. We say a function $f : \mathbb{N} \to [0, 1]$ is negligible if $f(\lambda) = \lambda^{-\omega(1)}$. The letter λ is reserved for the security parameter of a cryptographic scheme.

For a probability distribution $\mathcal{X}$ with support S, we denote $x \leftarrow \mathcal{X}$ to denote an independent sample from $\mathcal{X}$. Let $\mathcal{X}(s)$ be shorthand for $\Pr[x = s \mid x \leftarrow \mathcal{X}]$ for $s \in S$. We denote the uniform distribution over a finite set, Y, as $\mathcal{U}(Y)$. We will use the max-log metric on discrete probability distributions over the same support, $\Delta_{\text{ML}}(\mathcal{X}, \mathcal{Y}) := \max_{z \in S} |\log(\mathcal{X}(z)) - \log(\mathcal{Y}(z))|$ [30, 34] and we abbreviate two probability distributions being close under this metric as $\mathcal{X} \approx_s \mathcal{Y}$ (meaning they are statistically close).

Linear algebra. We view vectors as column vectors, and matrices as collections of column vectors $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k] \in \mathbb{R}^{n \times k}$. For any matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$, its *singular values* are the square roots of the first $d := \min(m, n)$ eigenvalues of its *Gram matrix* $\mathbf{A}^t\mathbf{A}$, listed in non-increasing order: $s_1(\mathbf{A}) \geq s_2(\mathbf{A}) \geq \cdots \geq s_d(\mathbf{A}) \geq 0$.

A symmetric matrix $\Sigma \in \mathbb{R}^{n \times n}$ is *positive-(semi)definite* if for all non-zero $\mathbf{x} \in \mathbb{R}^n$, $\mathbf{x}^t \Sigma \mathbf{x} > 0$ $(\geq)$. We denote a matrix as being positive-(semi)definite as

$\Sigma \succ 0$ ($\succeq$), and denote $\Sigma_1 - \Sigma_2 \succ 0$ ($\succeq$) as $\Sigma_1 \succ \Sigma_2$ ($\succeq$). This relation forms a partial ordering on the set of positive-(semi)definite matrices. We abbreviate scalar matrices, $\sigma \mathbf{I}_n$, by their scalar σ and denote $\sigma \succ \Sigma$ for $\sigma \mathbf{I}_n - \Sigma \succ 0$. For any two matrices $\mathbf{R}, \mathbf{S} \in \mathbb{R}^{n \times k}$, we denote $\mathbf{R} \geq \mathbf{S}$ if $\mathbf{R}\mathbf{R}^t \succeq \mathbf{S}\mathbf{S}^t$. We call $\mathbf{R}$ a square-root of Σ if $\mathbf{R}\mathbf{R}^t = \Sigma$. We denote square-roots of Σ as $\sqrt{\Sigma}$. If $\mathbf{R}$ is a square-root of Σ, then so is $\mathbf{RQ}$ for any orthogonal matrix $\mathbf{Q}$ ($\mathbf{QQ}^t = \mathbf{I}$).

View $\Sigma \succ 0$ through its block form, $\Sigma = \left[\begin{smallmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^t & \mathbf{D} \end{smallmatrix}\right]$. The Schur complement of $\mathbf{D}$ is $\Sigma/\mathbf{D} := \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^t$, and the Schur complement of $\mathbf{A}$ is $\Sigma/\mathbf{A} := \mathbf{D} - \mathbf{B}^t\mathbf{A}^{-1}\mathbf{B}$. A symmetric matrix Σ is positive definite if and only if $\Sigma/\mathbf{A}$ and $\mathbf{A}$ are positive definite—or equivalently $\Sigma/\mathbf{D}$ and $\mathbf{D}$. Further, Σ with $\mathbf{D} \succ 0$ is positive semidefinite if and only if $\Sigma/\mathbf{D} \succeq 0$.

2.1 Lattices and Discrete Gaussians

A *lattice* is a discrete subgroup in euclidean space, $\Lambda \subset \mathbb{R}^n$, and can be expressed as the set of all integer combinations of a basis $\Lambda = \mathbf{B}\mathbb{Z}^k = \{\sum_{i=1}^k z_i \mathbf{b}_i : z_i \in \mathbb{Z}\}$ for $\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_k] \in \mathbb{R}^{n \times k}$. The rank of a lattice $\Lambda = \mathbf{B}\mathbb{Z}^k$ is k and it is *full-rank* if $k = n$. Each lattice of rank $k \geq 2$ has infinitely many bases: any $\mathbf{B}' = \mathbf{BU}$ for $\mathbf{U} \in \mathbb{Z}^{k \times k}$ with $\det(\mathbf{U}) = \pm 1$ is a basis of the same lattice.

A *lattice coset* is a set of the form $\Lambda + \mathbf{a} := \{\mathbf{v} + \mathbf{a} \colon \mathbf{v} \in \Lambda\}$. Further, we call a subspace V of $\mathbb{R}^n$ a *Λ-subspace* if Λ intersect V spans V, $\operatorname{span}(\Lambda \cap V) = V$. The Gram-Schmidt orthogonalization of a basis $\mathbf{B}$ is $\widetilde{\mathbf{B}} = [\widetilde{\mathbf{b}_1}, \ldots, \widetilde{\mathbf{b}_k}]$ where $\widetilde{\mathbf{b}}_i := \mathbf{b}_i \perp \operatorname{span}(\mathbf{b}_1, \cdots, \mathbf{b}_{i-1})$. We define the *minimum GSO length* of a lattice as $\widetilde{bl}(\Lambda) := \min_{\mathbf{B}} \max_i \|\widetilde{\mathbf{b}}_i\|_2$, where the minimum is taken over all bases $\mathbf{B}$ of Λ.

The dual of a lattice is the set of vectors mapping the lattice to the integers via the inner product, $\Lambda^* = \{\mathbf{x} \in \operatorname{span}(\Lambda) : \langle \mathbf{x}, \mathbf{v} \rangle \in \mathbb{Z}, \forall \mathbf{v} \in \Lambda\}$. Let $\mathbf{T}$ be an invertible linear transformation. Then we have $(\mathbf{T}\Lambda)^* = \mathbf{T}^{-t}\Lambda^*$.

Discrete Gaussians. We define the Gaussian function centered at $\mathbf{t} \in \mathbb{R}^n$ as $\rho_{\mathbf{t}}(\mathbf{x}) := e^{-\pi\|\mathbf{x}-\mathbf{t}\|^2}$, and the warped Gaussian as

$$\rho_{\mathbf{t},\sqrt{\Sigma}} := e^{-\pi(\mathbf{x}-\mathbf{t})^t \Sigma^{-1}(\mathbf{x}-\mathbf{t})}$$

for some $\Sigma \succ 0$. We denote the Gaussian function as $\rho_{\mathbf{t},s}(\cdot)$ whenever Σ is a scalar, $\Sigma = s^2\mathbf{I}$. A *discrete Gaussian* distribution over a lattice Λ, or a lattice coset $\Lambda + \mathbf{a}$, is the probability distribution proportional to $\rho_{\sqrt{\Sigma}}(\mathbf{x})$ for $\mathbf{x} \in \Lambda + \mathbf{a}$. We denote this distribution as $\mathcal{D}_{\Lambda+\mathbf{a},\mathbf{t},\sqrt{\Sigma}}$ and $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda+\mathbf{a},\mathbf{t},\sqrt{\Sigma}}$ to mean $\mathbf{x}$ is efficiently sampled from $\mathcal{D}_{\Lambda+\mathbf{a},\mathbf{t},\sqrt{\Sigma}}$. In practice, we often sample from a distribution statistically close to $\mathcal{D}_{\Lambda+\mathbf{a},\mathbf{t},\sqrt{\Sigma}}$.

Fix $\epsilon > 0$, then the *smoothing parameter* [29] of Λ is

$$\eta_\epsilon(\Lambda) := \min\{s > 0 : \rho(s\Lambda^*) \leq 1 + \epsilon\}.$$

Informally, this is the smallest width of a discrete Gaussian over Λ which acts like a continuous Gaussian distribution. Notice that $s \geq \eta_\epsilon(\Lambda)$ if and only if $1 \geq \eta_\epsilon((1/s)\Lambda)$. We extend the definition to invertible matrices and say $\mathbf{T} =$

$\sqrt{\Sigma} \geq \eta_\epsilon(\Lambda)$ if and only if $1 \geq \eta_\epsilon(\mathbf{T}^{-1}\Lambda)$. We will use the following two lemmas in our analysis.

Lemma 1 ([29, 20]). *Let $\Lambda' \subseteq \Lambda$ be full-rank lattices in $\mathbb{R}^n$, $\epsilon \in (0,1)$, and $s \geq \eta_\epsilon(\Lambda')$. Let X be the probability distribution given by sampling $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda,s}$ and returning $\mathbf{x} \bmod \Lambda'$. Then,*

$$\Delta_{\text{ML}}(X, \mathcal{U}(\Lambda/\Lambda')) \leq \log \frac{1+\epsilon}{1-\epsilon} \leq \frac{2\epsilon}{1-\epsilon}.$$

Lemma 2 ([20, Lemma 3.1]). *For any rank-n lattice Λ and $\epsilon > 0$, we have $\eta_\epsilon(\Lambda) \leq \tilde{bl}(\Lambda) \cdot \sqrt{\log(2n(1+1/\epsilon))/\pi}$.*

The next lemma is helpful in analyzing smoothness criteria.

Lemma 3. *For any $\epsilon > 0$ and invertible matrix $\mathbf{S}, \mathbf{T} \in \mathbb{R}^{n\times n}$, $\eta_\epsilon(\Lambda) \leq \mathbf{S}$ if and only if $\eta_\epsilon(\mathbf{T}\Lambda) \leq \mathbf{TS}$.*

Proof. $\eta_\epsilon(\Lambda) \leq \mathbf{S}$ is equivalent to $\eta_\epsilon(\mathbf{S}^{-1}\Lambda) = \eta_\epsilon(\mathbf{S}^{-1}(\mathbf{T}^{-1}\mathbf{T})\Lambda) \leq 1$. □

The following theorem (first proved in [12] for the integer lattice $\mathbb{Z}^n$) regarding linear transformations and discrete Gaussians is most useful for this work.

Theorem 1 ([18, Theorem 3.2]). *For any matrix $\mathbf{S}$, lattice coset $A = \Lambda + \mathbf{a} \subset \text{span}(\mathbf{S})$, and linear transformation $\mathbf{T}$, if $\ker(\mathbf{T})$ is a Λ-subspace, and $\eta_\epsilon(\ker(\mathbf{T}) \cap \Lambda) \leq \mathbf{S}$, then*

$$\Delta_{\text{ML}}(\mathbf{T}(\mathcal{D}_{A,\mathbf{S}}), \mathcal{D}_{\mathbf{T}A,\mathbf{TS}}) \leq \log \frac{1+\epsilon}{1-\epsilon} \leq \frac{2\epsilon}{1-\epsilon}.$$

The following concentration bound is useful since we will be working with approximate trapdoors.

Lemma 4 (Lemma 4.4, [29]). *Let Λ be a lattice of rank n, $\mathbf{c} \in span(\Lambda)$, and $s \geq \eta_\epsilon(\Lambda)$. Then for $\mathbf{x} \leftarrow \mathcal{D}_{\Lambda,\mathbf{c},s}$,*

$$\Pr[\|\mathbf{x} - \mathbf{c}\| \geq s\sqrt{n}] \leq \frac{1+\epsilon}{1-\epsilon} 2^{-n}.$$

2.2 Polynomial Rings

We use R to denote some underlying polynomial ring, $R := \mathbb{Z}[x]/f(x)$ where $f(x)$ is a monic, irreducible polynomial. Further, we use $R_q := R/(qR)$ to denote its coefficients modulo q. The letter n is reserved for the ring's dimension, or $f(x)$'s degree. Common choices for R are cyclotomic rings and the NTRUPrime ring [5] $\mathbb{Z}[x]/(x^p - x - 1)$ for p prime. We emphasize that the online sampling phase of our trapdoors works for any polynomial ring, but its offline perturbation sampling phase is most efficient in cyclotomic rings.

Here we discuss power of two cyclotomic rings since our implementation is done in these rings. Let n be a power of two and $\zeta_{2n} \in \mathbb{C}$ be a $2n$-th primitive root of unity. The $2n$-th rational cyclotomic field's ring of integers is $R := \mathbb{Z}[\zeta_{2n}] \cong \mathbb{Z}[x]/(x^{2n}+1)$. Given an element in R, $f = \sum f_i \zeta_{2n}^i$, define its *coefficient embedding* as its vector of coefficients $(f_0, \dots, f_{n-1})$. All norms will be the l_2 (euclidean) norm in the coefficient embedding. Further, the coefficient embedding allows us to treat R^m as the lattice $\mathbb{Z}^{nm}$.

We will represent multiplication by a fixed ring element (say $p \in R$) as an "anticylic" matrix and call this ring embedding $\varphi : R \to \mathbb{Z}^{n \times n}$,

$$\varphi(p) = \begin{bmatrix} p_0 & -p_{n-1} & \cdots & -p_1 \\ p_1 & p_0 & \cdots & -p_2 \\ \vdots & \vdots & \ddots & \vdots \\ p_{n-1} & p_{n-2} & \cdots & p_0 \end{bmatrix}.$$

Further, we apply φ entry-wise to vectors and matrices over the ring R. An important fact is that matrix transpose of a ring element's matrix representation corresponds to the complex conjugation automorphism of a ring element, $\varphi(f)^t = \varphi(\overline{f})$. This implies $\varphi(f\overline{f}) \succ 0$ and $\varphi(\mathbf{e}^t\mathbf{e}) \succ 0$ for any non-zero $f \in R$ and $\mathbf{e} \in R^m$ since ζ_{2n} is primitive. For a matrix with independent discrete Gaussian entries over $\mathbf{R} \leftarrow \mathcal{D}_{R,s}^{m \times k}$, we have $s_1(\varphi(\mathbf{R})) \leq s\sqrt{n}O(\sqrt{m}+\sqrt{k}+\omega(\sqrt{\log n}))$ with overwhelming probability in n, [14, Appendix A]. Further, the empirically observed constant is $\approx 1/\sqrt{2\pi}$ [28, Section 2].

2.3 G-Lattices

Fix a modulus $q > 1$, an integer base $2 \leq b \leq \sqrt{q}$. Let $\mathbf{g}^t := [1, b, \dots, b^{k-1}] \in R_q^k$ with $k = \lceil \log_b q \rceil$. The matrix $\mathbf{g}^t$ is commonly referred to the gadget matrix since the Ring-SIS (RSIS) problem is easily solved on $\mathbf{g}^t$ [28]. The trapdoor schemes presented in Section 3 will sample a discrete Gaussian over the lattice coset

$$\Lambda_u^{\perp}(\mathbf{g}^t) := \{\mathbf{x} \in R^k : \mathbf{g}^t\mathbf{x} = u \in R_q\}.$$

We denote the process of sampling from the discrete Gaussian $\mathcal{D}_{\Lambda_u^{\perp}(\mathbf{g}^t),\sigma_g}$ as $\mathbf{x} \leftarrow \textsc{SampleG}(u, \sigma_g)$. Efficient algorithms for $\textsc{SampleG}(\cdot,\cdot)$ can be found in [28, 17].

2.4 RSIS and iNTRU

Here we discuss the hardness assumptions used throughout the paper. Our main result's hardness relies on the RSIS problem [1, 27, 25, 33] and the inhomogeneous NTRU (iNTRU) problem [16]. The RSIS problem with parameters $q, m \in \mathbb{N}, \beta \in \mathbb{R}^+$, denoted as $\mathsf{RSIS}_{q,m,\beta}$, over a ring R is: given a random vector $\mathbf{a} \leftarrow \mathcal{U}(R_q^m)$, find a short vector $\mathbf{x} \neq \mathbf{0}$ with $\|\mathbf{x}\| \leq \beta$ such that $\mathbf{a}^t\mathbf{x} = 0 \in R_q$. The HNF optimization of the RSIS problem is the RSIS where we are given $(1, \mathbf{a}') \in R_q^m$, with $\mathbf{a}'$ uniformly random in place of $\mathbf{a}$.

Let q be an integer modulus, χ be a distribution over R whose coefficient embedding is $\ll q$ with all but negligible probability, and $\mathbf{g}^t := [1, b, b^2, \dots, b^{k-1}] \in$

Notations	Description
n	Polynomial ring dimension
η_ϵ	Smoothing parameter
q	Modulus
b, $k = \log_b q$	Base and number of entries of the gadget vector $\mathbf{g}$
j	# components dropped in the approximate gadget $\mathbf{f}$
$m = k - j$	The dimension of the approximate gadget $\mathbf{f}$
σ_g	Gaussian width for G-lattice sampling, see Tables 3 and 4
Σ_p,s	Covariance and parameter of the perturbation $\mathbf{p}$, see Theorems 2 and 3
Σ_e,s_{err}	Covariance and parameter of the perturbation p_e, see Theorem 2
σ_e	Gaussian width of the verification error, see Tables 3 and 4
α,β	Approximate RSIS bounds, see Table 4

Table 2. A summary of parameters used to define our trapdoors. The pair Σ_p,s represents a discrete Gaussian convolution. For example, the perturbation has shape Σ_p and the output of the online sampling phase is statistically close to a spherical discrete Gaussian of width s. The same for the pair Σ_e,s_{err}. See Algorithms 2 and 4.

R_q^k be the gadget matrix (defined below). The $\mathsf{iNTRU}_{q,\chi}$ distribution is $r^{-1}(\mathbf{g}^t + \mathbf{e}^t)$ where $(r, \mathbf{e}) \leftarrow \chi^{k+1}$ and the iNTRU assumption is that $(r, \mathbf{e}) \mapsto r^{-1}(\mathbf{g}^t + \mathbf{e}^t)$ is a PRG. There is a reduction [16, Proposition 4.3] from iNTRU to a strengthened version of the RLWE problem [26].

We call $(\mathbf{a}, \mathbf{R})$ an (α, β)-approximate trapdoor if $\mathbf{a} \in R_q^m$ is (pseudo)random, and $\mathbf{R}$ is any matrix having the following property. Given $\mathbf{u} \in R_q$ as input, $\mathbf{R}$ allows one to efficiently find an $\mathbf{x} \in R^m$ such that $\|\mathbf{x}\| \leq \beta$ and $\|\mathbf{a}^t\mathbf{x} - u\| \leq \alpha$. It is shown in [7, Lemma 5.2] that (α, β)-approximate trapdoors is related to the RSIS problem via near collisions.

3 iNTRU Trapdoors

Fix a finite dimensional polynomial ring $R := \mathbb{Z}[x]/f(x)$ of degree n and its quotient ring modulo q for some modulus $q \in \mathbb{N}$, $R_q := R/qR$. Let R_q^* denote the polynomials of R_q which have multiplicative inverses. Fix an integer base $b \geq 2$, a gadget dimension $k = \lceil \log_b q \rceil$, and a gadget-approximate gadget pair $\mathbf{g}^t := [1, b, b^2, \ldots, b^{k-1}] = [\mathbf{m}^t|\mathbf{f}^t]$ for $\mathbf{f}^t := [b^j, \ldots, b^{k-1}]$, where $j \geq 0$ is the dimension of $\mathbf{m}$. Let $m := k - j$ be the approximate gadget's dimension. Further, let χ be a distribution over R such that $\|\chi\| \ll q$ with high probability. We list the parameters used to define our trapdoors in Table 2 for reference.

In this section, we present two efficient, gadget-based (approximate) RSIS trapdoor constructions. Both are pseudorandom whose pseudorandomness relies on the iNTRU problem, introduced in [16]. The first is a parallel approximate trapdoor where the online sampling portion of inversion can be done, after some preprocessing which corresponds to perturbation generation, by:

- one G-lattice sample,

- one polynomial multiplication (in-parallel),
- then one vector addition.

In addition, the first trapdoor's offline perturbation sampling is parallelized to sampling one non-spherical ring element. This is because the perturbation's shape is diagonal over the ring, $\Sigma_p = \gamma \mathbf{I}_m$ for γ a polynomial with positive-definite coefficient embedding.

The second trapdoor is an even more direct iNTRU-analog to the MP12 RLWE trapdoor than the first. Its ability to vary verification error allows for a smaller modulus, q. This is because the verification error only depends on if it uses the approximate gadget $\mathbf{f}$ or the exact gadget $\mathbf{g}$. Also, its secret signing key is $m+1$ polynomials in R with small entries compared to MP12's $2m$ polynomials, and the offline perturbation sampling reduces to sampling two positive-definite ring elements (with parallelism). Lastly, this second trapdoor scheme is compatible with applications requiring tag matrices. These are the trapdoor schemes where $\mathbf{a}^t \mathbf{R} = h \mathbf{g}^t$ for some $h \in R_q^*$ ($\mathbf{R}$ is the trapdoor). Both of the RSIS trapdoor schemes we present rely on Peikert's perturbation technique [31]. This technique is used to statistically hide the trapdoor. We analyze both trapdoors using the approximate gadget $\mathbf{f}^t$ in place of the gadget $\mathbf{g}^t$ for flexibility in parameter choices and because the analysis is nearly the same. Importantly, the second trapdoor scheme becomes an approximate trapdoor when using $\mathbf{f}^t$ in place of $\mathbf{g}^t$.

Trapdoor descriptions. Recall, [28, Definition 1] defines a $\mathbf{g}$*-trapdoor for* $\mathbf{a}^t \in R_q^m$ as any matrix $\mathbf{R}$ with small entries such that $\mathbf{a}^t \mathbf{R} = \mathbf{g}^t \in R_q^m$. One can easily invert the function $f_{\mathbf{a}}(\cdot)$ on any $u \in R_q$ when armed with a $\mathbf{g}$-trapdoor: sample $\mathbf{x} \leftarrow \text{SampleG}(u, \sigma_g)$, for $\sigma_g \geq \eta_\epsilon(\Lambda_q^\perp(\mathbf{g}^t))$[**], then return $\mathbf{Rx}$. Notice that $\mathbf{a}^t \mathbf{Rx} = u \in R_q$ and $\mathbf{Rx}$ has small entries since $\mathbf{R}$ has small entries. However, this clearly leaks information about the trapdoor. So, [28] uses Peikert's perturbation method [31] to hide the trapdoor.

Now recall the iNTRU distribution: $\mathbf{a} := r^{-1}(\mathbf{g}^t + \mathbf{e}^t) \in R_q^m$ where $(r, \mathbf{e}) \leftarrow \chi^{m+1}$ has small norm and $\mathbf{g}^t = [1, b, b^2, \ldots, b^{k-1}]$ is the gadget matrix. The polynomial r is *almost* a $\mathbf{g}$-trapdoor for $\mathbf{a}^t$ since $r\mathbf{a}^t = \mathbf{g}^t + \mathbf{e}^t \approx \mathbf{g}^t$. As a result, we can simply sample $\mathbf{x} \leftarrow \text{SampleG}(u, \sigma_g)$ and return $\mathbf{y} := r\mathbf{x}$. Then, we have

$$\mathbf{a}^t \mathbf{y} = \mathbf{g}^t \mathbf{x} + \mathbf{e}^t \mathbf{x} \approx \mathbf{g}^t \mathbf{x} = u \in R_q.$$

The relation above shows that we have *two* areas where we can leak information about the trapdoor, $(r, \mathbf{e})$. The first is in the domain of $f_{\mathbf{a}}(\cdot)$ given by $r\mathbf{x}$. The second is in the range of $f_{\mathbf{a}}(\cdot)$, given by the verification error $u - \mathbf{a}^t \mathbf{y} = \mathbf{e}^t \mathbf{x}$. This implies we need to use Peikert's perturbation method *twice*: once in $f_{\mathbf{a}}(\cdot)$'s domain and again in its range.

All of the above goes through when we replace the gadget $\mathbf{g}^t$ with the approximate gadget $\mathbf{f}^t$, similar to [7]. So, let $\mathbf{a}^t := r^{-1}(\mathbf{f}^t + \mathbf{e}^t)$ for this construction. Now, we have an algorithmic outline for approximate trapdoor inversion. Given a trapdoor $(r, \mathbf{e})$ and a target, we perform the following:

[**] The smoothing parameter of $\eta_\epsilon(\Lambda_q^\perp(\mathbf{g}^t))$ can be lower-bounded by $\approx b \cdot \eta_\epsilon(\mathbb{Z}^{nk})$ [28, Section 4]. And, $\eta_\epsilon(\mathbb{Z}^{nk})$ is no more than $\sqrt{\log(2nk(1+1/\epsilon))/\pi}$ by Lemma 2.

1. Sample a perturbation for the domain

$$\mathbf{p} \leftarrow \mathcal{D}_{R^m, \sqrt{\Sigma_p}},$$

where $\Sigma_p := (s^2 - \sigma_g^2 r\bar{r})\mathbf{I}_m \succ 0$, and a perturbation for the range

$$p_e \leftarrow \mathcal{D}_{R, \sqrt{\Sigma_e}},$$

where $\Sigma_e := s_{\mathrm{err}}^2 - \sigma_g^2 \mathbf{e}^t \mathbf{e} \succ 0$[††].
2. Set the shifted coset and sample from the G-lattice: $v := u - \mathbf{a}^t\mathbf{p} + p_e$ and $\mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$.
3. Add the perturbation in the domain of $f_{\mathbf{a}}(\cdot)$:

$$\mathbf{y} := r\mathbf{x} + \mathbf{p}.$$

Notice that the perturbations in the first step are statistically independent of the coset $u \in R_q$. Therefore, we can precompute them offline and store them, as is done in [28]. Further, we can store their hashes $\mathbf{a}^t\mathbf{p} \in R_q$. These perturbations can be computed in time $O(n \log n)$ using the algorithms of [15, 17] and $\mathbf{p}$ can be computed in parallel since its Gaussian shape is a diagonal over R. Then, online inversion is just a few efficient, parallel operations:

1. Shifting the coset ($v := u - \mathbf{a}^t\mathbf{p} + p_e$);
2. G-lattice sampling ($\mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$); and
3. Polynomial multiplication and addition ($\mathbf{y} := r\mathbf{x} + \mathbf{p}$).

The detailed trapdoor generation in this construction is shown in Algorithm 1 and the preimage sampling is shown in Algorithm 2.

Now we describe our second trapdoor scheme. Let $\mathbf{a}' := r^{-1}(\mathbf{g} + \mathbf{e})$, $\mathbf{a} := (1, \mathbf{a}')$ be the function description and $\mathbf{R} := \begin{bmatrix} -\mathbf{e}^t \\ r\mathbf{I}_m \end{bmatrix}$ the trapdoor. Then, $\mathbf{a}^t\mathbf{R} = \mathbf{g}^t$ and we can carry on as in [28]: To sample a preimage of $u \in R_q$,

1. first sample a perturbation $\mathbf{p} \leftarrow \mathcal{D}_{R^{m+1}, \sqrt{\Sigma_p}}$ for $\Sigma_p := s^2\mathbf{I}_{m+1} - \sigma_g^2 \mathbf{R}\mathbf{R}^t$,
2. set the shifted coset as $v := u = \mathbf{a}^t\mathbf{p}$ and sample from the G-lattice $\mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$,
3. then return $\mathbf{y} := \mathbf{R}\mathbf{x} + \mathbf{p} \in R^{m+1}$.

Note, here the online signing does not have a natural parallelization since $\mathbf{R}\mathbf{x}$ involves a polynomial multiplication by r and an inner product with $-\mathbf{e}^t$. We can also turn this into an approximate trapdoor scheme by replacing $\mathbf{g}^t$ with $\mathbf{f}^t$ for smaller preimages and public keys at the cost of a verification error. This verification error is smaller than the verification error in the first, parallel trapdoor.

[††] Notice how these are the best Gaussian shapes, in terms of spectral width, that one can hope for. The Gaussian shape of $r\mathbf{x}$ is $\sigma_g^2 r\bar{r}\mathbf{I}_m$ and the Gaussian shape of $\mathbf{e}^t\mathbf{x}$ is $\sigma_g^2\mathbf{e}^t\mathbf{e}$.

```
Algorithm 1: Approx.TrapGen($1^\lambda$)
  Input: The security parameter $1^\lambda$.
  Output: A function-approximate trapdoor pair.
    Sample $r \leftarrow \chi, \mathbf{e} \leftarrow \chi^m$.
    Set $\mathbf{a} := r^{-1}(\mathbf{f} + \mathbf{e}) \in R_q^m$.
    return $(\mathrm{vk}, \mathrm{sk}) := (\mathbf{a}, (r, \mathbf{e})) \in R_q^m \times R^{m+1}$.
```

```
Algorithm 2: Approx.SamplePre($\mathbf{a}, u, \mathrm{sk}, s, s_{\mathrm{err}}$)
  Input: A coset $u \in R_q$, a trapdoor $(r, \mathbf{e})$.
  Output: An approximate preimage of $u$.
    Sample (offline) perturbation $\mathbf{p} \leftarrow \mathcal{D}_{R^m,\sqrt{\Sigma_p}}$ for $\Sigma_p := s^2\mathbf{I}_m - \sigma_g^2 r\bar{r}\mathbf{I}_m$.
    Sample (offline) perturbation $p_e \leftarrow \mathcal{D}_{R,\sqrt{\Sigma_e}}$ for $\Sigma_e := s_{\mathrm{err}}^2 - \sigma_g^2 \mathbf{e}^t\mathbf{e}$.
    Set coset $v := u - \mathbf{a}^t\mathbf{p} + p_e \in R_q$.
    Sample the G-lattice $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \leftarrow \mathrm{SampleG}(v, \sigma_g)$ for $x_2 \in R^m$.
    Set the approximate preimage $\mathbf{y} := r\mathbf{x}_2 + \mathbf{p} \in R^m$.
    return $\mathbf{y}$.
```

Fig. 1. The trapdoor generation (Algorithm 1) and approximate preimage (Algorithm 2) sampling algorithms of our parallel trapdoor construction. The perturbations in Algorithm 2 are independent of the input image, $u \in R_q$, and can be precomputed and stored before, along with $\mathbf{a}^t\mathbf{p}$. These perturbations can be computed in time $O(n \log n)$ using [15] or [17, Section 4]. The rest of Algorithm 2 is the "online" sampling portion. This consists mostly of G-lattice sampling, which can be done in time $O(\log q)$ for each polynomial coefficient [17].

The rest of this section is organized as follows. Subsection 3.1 proves that the first iNTRU-based approximate RSIS trapdoor construction is secure in the random oracle model, which are summarized in Theorem 2. The associated algorithms are Algorithms 1 and 2. Subsection 3.2 contains the algorithms for the $\mathbf{g}$-trapdoors, Algorithms 3 and 4, as well as the theorem statement for its security in the random oracle model, Theorem 3.

3.1 Trapdoor 1: Approximate Trapdoor with Parallel Online Sampling

Our first trapdoor construction is specified in Algorithms 1 and 2. Here we state and prove that the approximate preimage samples are statistically close to a trapdoor-independent distribution in the random oracle model.

Theorem 2. *Sample $r \leftarrow \chi, \mathbf{e}^t \leftarrow \chi^m$ and set the trapdoor function description as $\mathbf{a}^t := r^{-1}(\mathbf{f}^t + \mathbf{e}^t) \in R_q^m$ as in Algorithm 1. Let $\eta := \eta_\epsilon(\mathbb{Z}^{nm}) = \eta_\epsilon(R^m)$, $\sigma_g = \eta_\epsilon(\Lambda_q^\perp(\mathbf{g}^t)) \geq \sqrt{b^2+1} \cdot \eta$ for some $\epsilon \in (0,1)$, $s \geq \sqrt{\sigma_g^2 r\bar{r} + \eta^2}$, and $s_{\mathrm{err}} \geq \sqrt{\sigma_g^2\mathbf{e}^t\mathbf{e} + \eta^2}$. Then, the following distributions are within a max-log distance of $3\log\frac{1+\epsilon}{1-\epsilon} \leq \frac{6\epsilon}{1-\epsilon}$: The "real" distribution with an honestly sampled approximate*

Preimage Gaussian Width, s	Verify Gaussian Width, σ_e
$\sqrt{(b^2+1)s_1^2(\chi)+1}\cdot\eta$	$\sqrt{(b^{2j}-1)\left(1+\frac{2}{b^2-1}\right)+m(b^2+1)s_1^2(\chi)+1}\cdot\eta$

Table 3. The parameters for the first approximate trapdoor construction from iNTRU. Let $\eta := \eta_\epsilon(R^m) = \eta_\epsilon(\mathbb{Z}^{nm})$ be the smoothing parameter of the underlying integer lattice. Let $s_1(\chi)$ be the expected singular value of $\varphi(\chi)$. We assume $\sigma_g := \sqrt{b^2+1}\eta$. The $\mathsf{RSIS}-\beta$ bound is $s\sqrt{nm}$ and the verification error's Gaussian width is σ_e.

preimage in the random oracle model (as in Figure 1)

$$\{(\mathbf{a},\mathbf{y},u,e) : u \leftarrow \mathcal{U}(R_q),\ \mathbf{y} \leftarrow \textsc{SamplePre}(\mathbf{a},u,\mathrm{sk},s,s_{\mathrm{err}}),\ e = u - \mathbf{a}^t\mathbf{y} \in R_q\},$$

and the "programmed," trapdoor-independent approximate preimage, distribution

$$\{(\mathbf{a},\mathbf{y},u,e) : \mathbf{y} \leftarrow \mathcal{D}_{R^m,s},\ e \leftarrow \mathcal{D}_{R,\sigma_e} \bmod q,\ u = \mathbf{a}^t\mathbf{y} + e \in R_q\},$$

for $\sigma_e := \sqrt{s_{\mathrm{err}}^2 + \sigma_g^2\frac{b^{2j}-1}{b^2-1}}$.

The intuition of the above theorem is clear once we see the samples from the viewpoint of the adversary:

$$\{(\mathbf{a},\mathbf{y},u,e) : u \leftarrow \mathcal{U}(R_q),\ \mathbf{y} \leftarrow \textsc{SamplePre}(\mathbf{a},u,\mathrm{sk},s,s_{\mathrm{err}}),\ e = u - \mathbf{a}^t\mathbf{y} \in R_q\}.$$

Let $\mathbf{y} \leftarrow \mathcal{D}_{R^m,s}$ and $e' \leftarrow \mathcal{D}_{R,s_{\mathrm{err}}}$ (trapdoor-independent samples). The proof is given through five hybrids as expressed through the shifted coset v in Algorithm 2:

$$\begin{aligned} u &= v + \mathbf{a}^t\mathbf{p} - p_e \\ &\approx_s \mathbf{g}^t\mathbf{x} + \mathbf{a}^t\mathbf{p} - p_e \\ &= \mathbf{f}^t\mathbf{x}_2 + \mathbf{a}^t\mathbf{p} - p_e + \mathbf{m}^t\mathbf{x}_1 \\ &= [\mathbf{a}^t, 1]\begin{bmatrix}\mathbf{I}_m & \mathbf{0} & r\mathbf{I}_m \\ \mathbf{0}^t & -1 & -\mathbf{e}^t\end{bmatrix}\begin{pmatrix}\mathbf{p} \\ p_e \\ \mathbf{x}_2\end{pmatrix} + \mathbf{m}^t\mathbf{x}_1 \\ &\approx_s \mathbf{a}^t\mathbf{y} + \mathbf{m}^t\mathbf{x}_1 + e'. \end{aligned}$$

Note how this equation allows us to analyze the perturbations in the function domain and the function range *simultaneously*. Let

$$\mathbf{L} := \begin{bmatrix}\mathbf{I}_m & \mathbf{0} & r\mathbf{I}_m \\ \mathbf{0}^t & -1 & -\mathbf{e}^t\end{bmatrix}.$$

Then, the proof comes down to showing

$$\mathbf{L}\begin{pmatrix}\mathbf{p} \\ p_e \\ \mathbf{x}_2\end{pmatrix} \approx_s \mathcal{D}_{R^{m+1},\sqrt{s^2\mathbf{I}\oplus s_{\mathrm{err}}^2}}.$$

This is done by analyzing linear transformations of discrete Gaussians, Theorem 1[‡‡]. We summarize the parameters in Table 3. Now we give a formal proof of Theorem 2

Proof (Theorem 2). Our proof is done via hybrids on the distribution of $(\mathbf{a}, \mathbf{y}, u, e)$. Notice that $\mathbf{f}^t = [\mathbf{a}^t, 1] \begin{bmatrix} r\mathbf{I}_m \\ -\mathbf{e}^t \end{bmatrix}$. Let $\mathbf{g}^t = [\mathbf{m}^t | \mathbf{f}^t]$. Let $\mathbf{p} \sim \mathcal{D}_{R^m, \sqrt{\Sigma_p}}$ for $\Sigma_p := s^2\mathbf{I} - \sigma_g^2 r\bar{r}\mathbf{I}$ and $p_e \leftarrow \mathcal{D}_{R, \sqrt{\Sigma_e}}$ for $\Sigma_e := s_{\text{err}}^2 - \sigma_g^2 \mathbf{e}^t\mathbf{e}$, as in Algorithm 2.

Hybrid 0. Let this distribution be honestly sampled preimage in the ROM. That is, $u \leftarrow \mathcal{U}(R_q)$ and $\mathbf{y} \leftarrow \textsc{SamplePre}(\mathbf{a}, u, \text{sk}, s, s_{\text{err}})$. Or in $\textsc{SamplePre}(\mathbf{a}, u, \text{sk}, s, s_{\text{err}})$, $\mathbf{p} \leftarrow \mathcal{D}_{R^m, \sqrt{\Sigma_p}}$, $p_e \leftarrow \mathcal{D}_{R, \sqrt{\Sigma_e}}$, $v := u - \mathbf{a}^t\mathbf{p} + p_e \in R_q$, $(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$, set $\mathbf{y} := r\mathbf{x}_2 + \mathbf{p}$, and return $(\mathbf{a}, \mathbf{y}, u, e)$.

Hybrid 1. Swap the order in which u and v are sampled. So, sample $v \leftarrow \mathcal{U}(R_q)$, sample the perturbations as before, and set $u = v + \mathbf{a}^t\mathbf{p} - p_e \in R_q$. Sample the preimage as before: $\mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$ and return $\mathbf{y} := r\mathbf{x}_2 + \mathbf{p}$. Hybrid 0 and Hybrid 1 have the same distribution on $(\mathbf{a}, \mathbf{y}, u, e)$.

Hybrid 2. Here we first sample over the G-lattice cosets $\mathbf{x} \leftarrow \mathcal{D}_{R^k, \sigma_g}$ and replace v with $\mathbf{g}^t\mathbf{x} \in R_q$. Since $\sigma_g \geq \eta_\epsilon(\Lambda_q^\perp(\mathbf{g}))$, Hybrids 1 and 2 are within a max-log distance of $\log \frac{1+\epsilon}{1-\epsilon}$ by Lemma 1. Notice that

$$u = [\mathbf{a}^t, 1]\mathbf{L} \begin{pmatrix} \mathbf{p} \\ p_e \\ \mathbf{x}_2 \end{pmatrix} + \mathbf{m}^t\mathbf{x}_1$$

in this hybrid.

Hybrid 3. Here we sample the trapdoor-independent preimage. Sample v and $\mathbf{x}$ as before and sample $(\mathbf{y}, e') \leftarrow \mathcal{D}_{R^{m+1}, \sqrt{s^2\mathbf{I} \oplus s_{\text{err}}^2}}$. Theorem 1 combined with Lemma 5 (below) tells us the max-log distance between

$$\mathbf{L} \begin{pmatrix} \mathbf{p} \\ p_e \\ \mathbf{x}_2 \end{pmatrix} \text{ and } \mathcal{D}_{R^{m+1}, \sqrt{s^2\mathbf{I} \oplus s_{\text{err}}^2}}$$

is at most $\log \frac{1+\epsilon}{1-\epsilon}$. We set $u := \mathbf{a}^t\mathbf{y} + e' + \mathbf{m}^t\mathbf{x}_1$. Therefore, Hybrids 2 and 3 are within a max-log distance of $\log \frac{1+\epsilon}{1-\epsilon}$.

[‡‡] We remark that the analogous proof in [20] amounts to s being above the smoothing parameter, $s \geq \eta_\epsilon(\Lambda_q^\perp(\mathbf{A}))$, and Lemma 1. In our situation, we must *carefully* show the trapdoor is statistically hidden with the kernel lemma, Lemma 5.

Hybrid 4. Here we combine the error terms $\mathbf{m}^t\mathbf{x}_1$ and e' using Theorem 1. Note that $\mathbf{m}^t\mathbf{x}_1 + e'$ can be expressed as $[1|\mathbf{m}_1^t]\mathcal{D}_{R^{j+1},\sqrt{s_{\text{err}}^2 \oplus \sigma_g^2 \mathbf{I}_j}}$ (over the ring). Now, the kernel of $[1|\mathbf{m}_1^t]$ is spanned by $(1, -1, \mathbf{0})$ and

$$(0, b, -1, \mathbf{0}), (0, 0, b, -1, 0, \dots), \dots, (\mathbf{0}, b, -1)$$

and its intersection with R^{j+1} has smoothing parameter at most $\sqrt{b^2+1}\eta$ by the GPV bound, Lemma 2. The kernel lattice is full-rank in the kernel of $[1|\mathbf{m}_1^t]$ and the discrete Gaussian is smooth over the kernel lattice Theorem 1.

□

Next, we prove the main statistical lemma used in the Theorem 2's proof.

Lemma 5. *Let*

- $\Sigma_p = s^2\mathbf{I}_m - \sigma_g^2 r\bar{r}\mathbf{I}_m$,
- $\Sigma_e = s_{\text{err}}^2 - \sigma_g^2 \mathbf{e}^t\mathbf{e}$,
- $\mathbf{L} = \begin{bmatrix} \mathbf{I}_m & \mathbf{0} & r\mathbf{I}_m \\ \mathbf{0}^t & -1 & -\mathbf{e}^t \end{bmatrix}$, *as in Theorem 2's proof,*
- $\Gamma = \{\mathbf{x} \in R^{2m+1} : \mathbf{L}\mathbf{x} = \mathbf{0} \in R^{m+1}\}$,
- *and* $\sigma_g > \eta_\epsilon(\mathbb{Z}^{nm})$, *for some* $\epsilon \in (0, 1)$.

Then, $\sqrt{\Sigma_p \oplus \Sigma_e \oplus \sigma_g^2\mathbf{I}} \geq \eta_\epsilon(\Gamma)$ *if*

$$s^2\mathbf{I} \succeq \sigma_g^2 r\bar{r} + \eta_\epsilon^2(\mathbb{Z}^{nm})\mathbf{I} \text{ and } s_{\text{err}}^2\mathbf{I} \succeq \sigma_g^2\mathbf{e}^t\mathbf{e} + \eta_\epsilon^2(\mathbb{Z}^{nm})\mathbf{I}.$$

Proof. We write ring elements, and vectors and matrices over the ring, without the coefficient embedding, $\varphi(\cdot)$, throughout the proof with the understanding that we are using φ's euclidean geometry. This significantly improves the proof's readability.
Consider that a basis for Γ is

$$\mathbf{B} = \begin{bmatrix} -r\mathbf{I}_m \\ -\mathbf{e}^t \\ \mathbf{I}_m \end{bmatrix} \in R^{(2m+1)\times m}.$$

One can see this via Γ's definition,

$$\Gamma = \{\mathbf{x} = (\mathbf{x}_1, x_2, \mathbf{x}_3) \in R^{2m+1} : \mathbf{x}_1 + r\mathbf{x}_3 = \mathbf{0} \in R^m, x_2 + \mathbf{e}^t\mathbf{x}_3 = 0 \in R\}.$$

Next, we use Lemma 3. Let

$$\mathbf{T} := \begin{bmatrix} \mathbf{I}_{m+1} & \mathbf{Q} \\ \mathbf{0} & \mathbf{I}_m \end{bmatrix} \text{ where } \mathbf{Q} := \begin{bmatrix} r\mathbf{I}_m \\ \mathbf{e}^t \end{bmatrix}.$$

Notice that

$$\mathbf{T}\mathbf{B} = \begin{bmatrix} \mathbf{0} \\ \mathbf{I}_m \end{bmatrix} \in R^{(2m+1)\times m}.$$

In other words,

$$\mathbf{T} \cdot \Gamma = \{\mathbf{0}\} \oplus R^m.$$

Lemma 3 tells us $\sqrt{\Sigma_p \oplus \Sigma_e \oplus \sigma_g^2 \mathbf{I}} \geq \eta_\epsilon(\Gamma)$ if and only if $\mathbf{T}\sqrt{\Sigma_p \oplus \Sigma_e \oplus \sigma_g^2 \mathbf{I}} \geq \eta_\epsilon(\mathbf{T}\Gamma) = \eta_\epsilon(R^m)$. Let $\Sigma := \Sigma_p \oplus \Sigma_e \oplus \sigma_g^2 \mathbf{I}$. Then a (tedious) computation tells us

$$\Sigma' := \mathbf{T}\Sigma\mathbf{T}^t = \begin{bmatrix} \mathbf{A} & \sigma_g^2 \mathbf{Q} \\ \sigma_g^2 \mathbf{Q}^t & \sigma_g^2 \mathbf{I}_m \end{bmatrix}$$

where

$$\mathbf{A} = \begin{bmatrix} s^2 \mathbf{I}_m & \sigma_g^2 r \mathbf{e} \\ \sigma_g^2 \overline{r} \mathbf{e}^t & s_{\text{err}}^2 \end{bmatrix}.$$

Let $\eta := \eta_\varepsilon(R^m) = \eta_\varepsilon(\mathbb{Z}^{nm})$. Now we use Schur complements in order to show $\Sigma'' := \Sigma' - \eta^2 \mathbf{I} \succeq 0$. Let $\mathbf{D} := (\sigma_g^2 - \eta^2)\mathbf{I}$ and recall $\mathbf{D} \succ 0$ by assumption. Recall, a symmetric matrix $\Sigma'' = \left[\begin{smallmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^t & \mathbf{D} \end{smallmatrix}\right]$ with $\mathbf{D} \succ 0$ is positive-semidefinite if and only if $\Sigma''/\mathbf{D} \succeq 0$.

Notice that

$$\Sigma''/\mathbf{D} = \begin{bmatrix} (s^2 - \eta^2 - \sigma_g^2 r\overline{r})\mathbf{I} & \mathbf{0} \\ \mathbf{0} & s_{\text{err}}^2 - \eta^2 - \sigma_g^2 \mathbf{e}^t \mathbf{e} \end{bmatrix}.$$

Therefore, $\Sigma''/\mathbf{D}$ is positive semidefinite assuming

$$s_{\text{err}}^2 \succeq \sigma_g^2 \mathbf{e}^t \mathbf{e} + \eta^2, \text{ and } s^2 \mathbf{I} \succeq (\sigma_g^2 r\overline{r} + \eta^2)\mathbf{I}.$$

□

3.2 Trapdoor 2: f-Trapdoor

Our second trapdoor construction is an iNTRU-analog of MP12's trapdoor. Let $\mathbf{a}^t = [1, r^{-1}(\mathbf{f}^t + \mathbf{e}^t)]$ be the SIS matrix and let $\mathbf{R} = \left[\begin{smallmatrix} -\mathbf{e}^t \\ r\mathbf{I}_m \end{smallmatrix}\right]$ be the trapdoor. Then, we have the trapdoor relation $\mathbf{a}^t \mathbf{R} = \mathbf{f}^t$. Now we can proceed as in [28, 7] when given an RSIS image $u \in R_q$:

- Sample a perturbation $\mathbf{p}$ independent of the image u. This can be done offline.
- Sample the input coset shifted by the perturbation's SIS image, as in $\mathbf{x} \leftarrow \textsc{SampleG}(v, \sigma_g)$ for $v = u - \mathbf{a}^t \mathbf{p}$. We drop the necessary entries of $\mathbf{x}$ to match $\mathbf{f}$'s dimension.
- Convolve the samples by returning $\mathbf{y} := \mathbf{R}\mathbf{x} + \mathbf{p}$.

The algorithms are given in Algorithms 3 and 4. Further, we list the parameters in Table 4. If R is a cyclotomic rings, we can efficiently sample the perturbation by using the structured samplers of [15, 17]. A detailed example is given in Section 4. Similar to the previous trapdoor, this trapdoor has efficient online sampling over any ring with efficient polynomial multiplication.

The following theorem's statement and proof are nearly identical to [7]'s modification to [28]. The intuition is the same as Theorem 2. We include a detailed proof in the full version of this paper.

```
Algorithm 3: $\mathbf{f}.\textsc{TrapGen}(1^\lambda)$
  Input: The security parameter $1^\lambda$.
  Output: A function-approximate trapdoor pair.
    Sample $r \leftarrow \chi, \mathbf{e} \leftarrow \chi^m$.
    Set $\mathbf{a}' := r^{-1}(\mathbf{f} + \mathbf{e}) \in R_q^m$ and $\mathbf{a} := (1, \mathbf{a}') \in R_q^{m+1}$.
    return $(\mathrm{vk}, T) := (\mathbf{a}, (r, \mathbf{e})) \in R_q^m \times R^{m+1}$.
```

```
Algorithm 4: $\mathbf{f}.\textsc{SamplePre}(\mathbf{a}, u, \mathrm{sk}, s)$
  Input: A coset $u \in R_q$, and a trapdoor $(r, \mathbf{e})$.
  Output: An approximate preimage of $u$.
    Sample perturbation $\mathbf{p} \leftarrow \mathcal{D}_{R^{m+1}, \sqrt{\Sigma_p}}$ for $\Sigma_p := s^2\mathbf{I}_{m+1} - \sigma_g^2 \begin{bmatrix} \mathbf{e}^t\mathbf{e} & -\bar{r}\mathbf{e}^t \\ -r\mathbf{e} & r\bar{r}\mathbf{I}_m \end{bmatrix}$.
    Set coset $v := u - \mathbf{a}^t\mathbf{p} \in R_q$.
    Sample the G-lattice $\mathbf{x} = (\mathbf{x}_1, \mathbf{x}_2) \leftarrow \textsc{SampleG}(v, \sigma_g)$.
    Define the $\mathbf{f}$-trapdoor as $\mathbf{R} := \begin{bmatrix} -\mathbf{e}^t \\ r\mathbf{I}_m \end{bmatrix}$.
    Set the approximate preimage $\mathbf{y} := \mathbf{R}\mathbf{x}_2 + \mathbf{p} \in R^m$.
    return $\mathbf{y}$.
```

Fig. 2. The trapdoor generation and approximate preimage sampling algorithms of our second $\mathbf{f}$-trapdoor construction. The perturbations in Algorithm 4 are independent of the input image u, and can be precomputed and stored before, along with $\mathbf{a}^t\mathbf{p}$. The covariance Σ_p can be decomposed using the Schur decomposition, and the perturbation $\mathbf{p}$ can be computed in parallel time $O(n \log n)$ in cyclotomic rings as in [17, Section 4]. The rest of Algorithm 4 is the "online" sampling phase, which consists mostly of G-lattice sampling that can be done in time $O(\log q)$ for each polynomial coefficient [17].

Theorem 3. *Sample $r \leftarrow \chi, \mathbf{e}^t \leftarrow \chi^m$ and set the trapdoor function description as $\mathbf{a} = (1, \mathbf{a}') := (1, r^{-1}(\mathbf{f} + \mathbf{e})) \in R_q^{m+1}$ as in Algorithm 3. Let $\eta := \eta_\epsilon(\mathbb{Z}^{nm})$ and $\sigma_g = \eta_\epsilon(\Lambda_q^\perp(\mathbf{g}^t)) \geq \sqrt{b^2+1} \cdot \eta$ for some $\epsilon \in (0,1)$, and let $s \succeq \sqrt{\sigma_g^2\mathbf{R}\mathbf{R}^t + \eta^2\mathbf{I}_{m+1}}$. Then, the following distributions are within a max-log distance $3\log\frac{1+\epsilon}{1-\epsilon} \leq \frac{6\epsilon}{1-\epsilon}$: The "real" distribution with an honestly sampled approximate preimage in the random oracle model (as in Figure 2)*

$$\{(\mathbf{a}, \mathbf{y}, u, e) : u \leftarrow \mathcal{U}(R_q),\ \mathbf{y} \leftarrow \textsc{SamplePre}(\mathbf{a}, u, \mathrm{sk}, s),\ e = u - \mathbf{a}^t\mathbf{y} \in R_q\},$$

and the "programmed," trapdoor-independent approximate preimage, distribution

$$\{(\mathbf{a}, \mathbf{y}, u, e) : \mathbf{y} \leftarrow \mathcal{D}_{R^m, s},\ e \leftarrow \mathcal{D}_{R, \sigma_e} \bmod q,\ u = \mathbf{a}^t\mathbf{y} + e \in R_q\},$$

for $\sigma_e := \sigma_g\sqrt{\frac{b^{2j}-1}{b^2-1}}$.

Preimage Gaussian Width, s	Verify Gaussian Width, σ_e
$\sqrt{m(b^2+1)s_1^2(\chi)+1}\cdot\eta$	$\sqrt{(b^{2j}-1)\left(1+\frac{2}{b^2-1}\right)}\cdot\eta$

Table 4. The parameters for the second $\mathbf{f}$-trapdoor construction from iNTRU. Let $\eta := \eta_\epsilon(R^m) = \eta_\epsilon(\mathbb{Z}^{nm})$ be the smoothing parameter of the underlying integer lattice. Let $s_1(\chi)$ be the expected singular value of $\varphi(\chi)$. We assume $\sigma_g := \sqrt{b^2+1}\eta$. Also, the $\mathsf{RSIS}-\beta$ bound is $s\sqrt{nm}$ and the verification width α is $\sigma_e\sqrt{n}$. We remark that the verification width matches [7] and s matches [28].

4 Implementation and Experiment

We implemented both our trapdoor schemes using the PALISADE library. In our proof-of-concept implementations[§§], we use power-of-2 cyclotomic rings, and we set the size of modulus to be no more than 30 bits. These parameters allow us to take advantage of the fast ring operations available in PALISADE and use native integer arithmetic. We conducted some experiments on an Intel i7-4790 CPU with 8 cores to compare the performance of the hash-and-sign signature schemes using our trapdoors versus the RLWE based approximate trapdoor in [7]. We now describe some implementation details and discuss the experiment results.

4.1 Perturbation Generation

The offline part of our preimage sampling algorithms require some perturbation vectors being sampled from discrete Gaussian distributions with certain covariances. Efficient sampling algorithms for power-of-2 cyclotomic rings have been proposed in [17, 11], and they are implemented in PALISADE. Among them, we are particularly interested in the algorithm SampleFz, which on input a ring element f samples a perturbation $\hat{p} \leftarrow \mathcal{D}_{\mathbb{Z}^n,c,\sqrt{\phi(f)}}$.

In Algorithm 2, since Σ_p and Σ_e are both diagonal matrices, we can directly call SampleFz on their diagonal entries to sample $\mathbf{p}$ and p_e. More specifically, we make m parallel and independent calls to SampleFz on the diagonal entries $s^2 - \sigma_g^2 r\overline{r}$ to sample $\mathbf{p}$. Similarly, we compute $s_{err}^2 - \sigma_g^2\mathbf{e}^t\mathbf{e}$ and call SampleFz to sample p_e.

In Algorithm 4, we compute the Schur complement decomposition of the covariance matrix Σ_p as

$$\Sigma_p = \begin{bmatrix}\mathbf{A} & \mathbf{B}\\ \mathbf{B}^t & \mathbf{D}\end{bmatrix} = \begin{bmatrix}\mathbf{I} & \mathbf{B}\mathbf{D}^{-1}\\ \mathbf{0} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\Sigma/\mathbf{D} & \mathbf{0}\\ \mathbf{0} & \mathbf{D}\end{bmatrix}\begin{bmatrix}\mathbf{I} & \mathbf{0}\\ \mathbf{D}^{-1}\mathbf{B}^t & \mathbf{I}\end{bmatrix},$$

where $\mathbf{A} = s^2 - \sigma_g^2\mathbf{e}^t\mathbf{e}$, $\mathbf{B} = \sigma_g^2\overline{r}\mathbf{e}^t$, $\mathbf{D} = (s^2 - \sigma_g^2 r\overline{r})\mathbf{I}_m$, and $\Sigma/\mathbf{D} = \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{B}^t$. Using the convolution theorem in [17, Lemma 4.1], we generate the perturbation $\mathbf{p} = (p_1, \mathbf{p}_2)$ in the following steps:

[§§] Source code can be downloaded at `https://www.dropbox.com/s/uz3g3atpqu7u87n/intrusign.zip?dl=0`

Params	Exact	Exact	Exact	Approx	Exact	Approx	Exact	Approx	Exact
n	512	512	1024	1024	1024	1024	1024	1024	8192
$\lceil \log_2 q \rceil$	13	20	20	20	27	27	30	30	30
b	4	4	4	4	4	4	8	8	32
j	0	0	0	3	0	3	0	4	0
σ	10	32	32	32	107	107	181	181	181
s	2259.23	7229.5	10224.1	10224.1	34186.7	34186.7	113080	113080	1.27009e+06
nm	3072	3072	10240	7168	13312	10240	10240	6144	49152
κ	42	103	243	243	278	278	251	251	2192
κ'	299	299	397	397	220	220	179	179	2819

Table 5. Summary of the concrete bit security estimations for the parameters mentioned in Section 4.2, where $\kappa = 0.292k$ is the classical bit security estimation using the method mentioned in Section 4.2, and κ' is the classical bit security estimation using the overstretched attack of [23].

1. Sample $\mathbf{p}_2 \leftarrow \mathcal{D}_{\mathbb{Z}^{nm},\sqrt{\mathbf{D}}}$. Since $\mathbf{D}$ is diagonal, we simply call SampleFz m times in parallel to sample each entry of $\mathbf{q}$.
2. Compute $c = \mathbf{B}\mathbf{D}^{-1}\mathbf{q}$.
3. Sample $p_1 \leftarrow \mathcal{D}_{\mathbb{Z}^m,c,\sqrt{\mathbf{A}-\mathbf{B}\mathbf{D}^{-1}\mathbf{B}^t}}$.

4.2 Concrete Parameters.

We estimated concrete security by estimating how long it would take the BKZ algorithm to break the scheme. Recall, BKZ [36] is a family of basis reduction algorithms which reduces to an SVP oracle of dimension $k < r$, where r is the input lattice's rank. The BKZ algorithm with block size k is expected to return a basis with shortest vector length $\delta_k^r \det(\Lambda)^{1/r}$ where δ_k is the *root Hermite factor*. The root Hermite factor is, asymptotically, the following function of the block size [8]

$$\delta_k \approx \left(\frac{k}{2\pi e} (\pi k)^{1/k} \right)^{\frac{1}{2(k-1)}}.$$

We used the core SVP hardness model [3] for BKZ. Here, we assume BKZ only needs one SVP oracle call. Further, we use [4] for the best known heuristic SVP time complexities: $2^{.292k(1+o(1))}$ classically and $2^{.265k(1+o(1))}$ for a quantum computer [24].

Given the above, we found the smallest block size k which would break the underlying $\mathsf{RSIS}_{q,m+1,2(\alpha+\beta)}$. We remark that this is a crude concrete security estimate, but we choose this method for its simplicity. This is not a state-of-the-art concrete security estimate.

We selected several sets of concrete parameters, which are summarized in Table 5 along with the estimated bit security levels.

4.3 Performance Comparisons with other schemes

Our first two sets of parameters fall into the regime of Falcon, with $n = 512$, $q = 12289$ and $\log q = 20$. and $\sigma = q^{1/4} = 10$ and 32. For such parameters, the verification error width β exceeds $q/2$, so the signing error for our first trapdoor becomes big enough such that verification would not be able to distinguish correct and incorrect plaintexts. However, our second trapdoor works just fine.

The next several sets of parameters have $n = 1024$ and various sizes of modulus q and gadget base b. When $\log q \geq 27$, the signing error in our first trapdoor becomes sufficiently small to allow valid verification. Notice that for these parameters, the modulus q falls into the range of the overstretched attack, and we set $\sigma = q^{1/4}$ as suggested in [2].

Our last parameter chooses a large ring dimension $n = 8192$ but with relatively small modulus $\log q = 30$. We observed that our preimage sampling algorithms still run very fast with such parameters. In contrast, when Falcon is instantiated using large rings, the online stage may become slow due to the complicated ring decomposition operations.

Experiments and performance results. For each set of parameter, we ran our signature schemes 100 times each, and we recorded the offline and online signing times. To compare with [7], we modified PALISADE's implementation of [28] by adding the approximate trapdoor capability. The RLWE based schemes do not require large σ, so we ran it using the same parameters except with $\sigma = 8$, and we measure the offline and online signing times in the same way. These experiment results show that both the offline sampling times of our signature schemes are very close to the RLWE based signature schemes in [7], with about 5% to 20% slowdowns, while the online signing times are the same or shorter in a few cases. We also measured the sizes of the signing and the verification keys in these schemes, and as expected, the results show that our signing keys are almost half in size comparing to [7], and our verification keys are slightly smaller. The detailed performance numbers are shown in Table 1.

In the offline stage, the slowdowns are mainly due to more complicated covariance matrices for the perturbation vectors, as our trapdoors require non-spherical discrete Gaussian sampling over integers for all $m+1$ perturbation ring elements, where as in [28] only two perturbation ring elements are non-spherical. However, with the help of parallel computation, the slowdowns are not significant.

Moreover, with the help of parallel computation, signing using our second $\mathbf{f}$-trapdoor of Algorithm 4 takes only about 10% more time comparing with our first, noisy trapdoor of Algorithm 2. The extra time is mainly spent on computing the center and covariance for the first part p_1 of the perturbation.

For the online stage, our signing algorithms take the same number of operations comparing to [7], but because we can take advantage of parallel computation, our schemes run slightly faster.

References

1. Miklós Ajtai. Generating hard instances of lattice problems (extended abstract). In *STOC*, pages 99–108. ACM, 1996.
2. Martin R. Albrecht, Shi Bai, and Léo Ducas. A subfield lattice attack on overstretched NTRU assumptions - cryptanalysis of some FHE and graded encoding schemes. In *CRYPTO (1)*, volume 9814 of *Lecture Notes in Computer Science*, pages 153–178. Springer, 2016.
3. Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In *USENIX Security Symposium*, pages 327–343. USENIX Association, 2016.
4. Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In *SODA*, pages 10–24. SIAM, 2016.
5. Daniel J. Bernstein, Chitchanok Chuengsatiansup, Tanja Lange, and Christine van Vredendaal. NTRU prime: Reducing attack surface at low cost. In *SAC*, volume 10719 of *Lecture Notes in Computer Science*, pages 235–260. Springer, 2017.
6. Dan Boneh, Craig Gentry, Sergey Gorbunov, Shai Halevi, Valeria Nikolaenko, Gil Segev, Vinod Vaikuntanathan, and Dhinakaran Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, volume 8441 of *Lecture Notes in Computer Science*, pages 533–556. Springer, 2014.
7. Yilei Chen, Nicholas Genise, and Pratyay Mukherjee. Approximate trapdoors for lattices and smaller hash-and-sign signatures. In *ASIACRYPT (3)*, volume 11923 of *Lecture Notes in Computer Science*, pages 3–32. Springer, 2019.
8. Yuanmi Chen. *Réduction de réseau et sécurité concréte du chiffrement complétement homomorphe*. PhD thesis, Paris 7, 2013.
9. Jung Hee Cheon, Duhyeong Kim, Taechan Kim, and Yongha Son. A new trapdoor over module-ntru lattice and its application to id-based encryption. *IACR Cryptology ePrint Archive*, 2019:1468, 2019.
10. Chitchanok Chuengsatiansup, Thomas Prest, Damien Stehlé, Alexandre Wallet, and Keita Xagawa. Modfalcon: compact signatures based on module NTRU lattices. *IACR Cryptology ePrint Archive*, 2019:1456, 2019.
11. David Bruce Cousins, Giovanni Di Crescenzo, Kamil Doruk Gür, Kevin King, Yuriy Polyakov, Kurt Rohloff, Gerard W. Ryan, and Erkay Savas. Implementing conjunction obfuscation under entropic ring LWE. In *2018 IEEE Symposium on Security and Privacy, SP 2018, Proceedings, 21-23 May 2018, San Francisco, California, USA*, pages 354–371, 2018.
12. Léo Ducas, Steven D. Galbraith, Thomas Prest, and Yang Yu. Integral matrix gram root and lattice gaussian sampling without floats. In *EUROCRYPT (2)*, volume 12106 of *Lecture Notes in Computer Science*, pages 608–637. Springer, 2020.
13. Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In *ASIACRYPT (2)*, volume 8874 of *Lecture Notes in Computer Science*, pages 22–41. Springer, 2014.
14. Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO (1)*, volume 8616 of *Lecture Notes in Computer Science*, pages 335–352. Springer, 2014.
15. Léo Ducas and Thomas Prest. Fast fourier orthogonalization. In *ISSAC*, pages 191–198. ACM, 2016.

16. Nicholas Genise, Craig Gentry, Shai Halevi, Baiyu Li, and Daniele Micciancio. Homomorphic encryption for finite automata. In *ASIACRYPT (2)*, volume 11922 of *Lecture Notes in Computer Science*, pages 473–502. Springer, 2019.
17. Nicholas Genise and Daniele Micciancio. Faster gaussian sampling for trapdoor lattices with arbitrary modulus. In *EUROCRYPT (1)*, volume 10820 of *Lecture Notes in Computer Science*, pages 174–203. Springer, 2018.
18. Nicholas Genise, Daniele Micciancio, Chris Peikert, and Michael Walter. Improved discrete gaussian and subgaussian analysis for lattice cryptography. In *Public Key Cryptography (1)*, volume 12110 of *Lecture Notes in Computer Science*, pages 623–651. Springer, 2020.
19. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178. ACM, 2009.
20. Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. ACM, 2008.
21. Sergey Gorbunov, Vinod Vaikuntanathan, and Daniel Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, pages 469–477. ACM, 2015.
22. Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 267–288. Springer, 1998.
23. Paul Kirchner and Pierre-Alain Fouque. Revisiting lattice attacks on overstretched NTRU parameters. In *EUROCRYPT (1)*, volume 10210 of *Lecture Notes in Computer Science*, pages 3–26, 2017.
24. Thijs Laarhoven. *Search Problems in Cryptography.* PhD thesis, Eindhoven University of Technology, 2015.
25. Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, volume 4052 of *Lecture Notes in Computer Science*, pages 144–155. Springer, 2006.
26. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23. Springer, 2010.
27. Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions from worst-case complexity assumptions. In *FOCS*, pages 356–365. IEEE Computer Society, 2002.
28. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 700–718. Springer, 2012.
29. Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.
30. Daniele Micciancio and Michael Walter. Gaussian sampling over the integers: Efficient, generic, constant-time. In *CRYPTO (2)*, volume 10402 of *Lecture Notes in Computer Science*, pages 455–485. Springer, 2017.
31. Chris Peikert. An efficient and parallel gaussian sampler for lattices. In *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97. Springer, 2010.
32. Chris Peikert. A decade of lattice cryptography. *Foundations and Trends in Theoretical Computer Science*, 10(4):283–424, 2016.
33. Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 145–166. Springer, 2006.

34. Thomas Prest. Sharper bounds in lattice-based cryptography using the rényi divergence. In *ASIACRYPT (1)*, volume 10624 of *Lecture Notes in Computer Science*, pages 347–374. Springer, 2017.
35. Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In *STOC*, pages 84–93. ACM, 2005.
36. Claus-Peter Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math. Program.*, 66:181–199, 1994.