

Improved Rejection Sampling for Compact Lattice Signatures

Joel Gärtner

KTH Royal Institute of Technology
jgartner@kth.se

December 27, 2024

Abstract

One of the primary approaches used to construct lattice-based signature schemes is through the “Fiat-Shamir with aborts” methodology. Such a scheme may abort and restart during signing which corresponds to rejection sampling produced signatures to ensure that they follow a distribution that is independent of the secret key. This rejection sampling is only feasible when the output distribution is sufficiently wide, limiting how compact this type of signature schemes can be.

In this work, we develop a new method to construct signatures influenced by the rejection condition. This allows our rejection sampling to target significantly narrower output distributions than previous approaches, allowing much more compact signatures. The combined size of a signature and a verification key for the resulting scheme is less than half of that for ML-DSA and comparable to that of compact hash-and-sign lattice signature schemes, such as Falcon.

1 Introduction

As a part of their post-quantum standardization process, NIST has already standardized the signature scheme CRYSTALS-Dilithium [25] as ML-DSA and is in the process of standardizing Falcon. Both Dilithium and Falcon are lattice-based signature schemes but they are constructed using two different approaches, corresponding to the two main approaches for constructing lattice-based signature schemes. The first of these is the hash-and-sign approach of Gentry, Peikert and Vaikuntanathan [21] on which Falcon is based. The second of these, on which Dilithium is based, is the one of Lyubashevsky [23, 24], which consists of an aborting variant of the Fiat-Shamir transform.

One of the primary reasons for why NIST decided to standardize Falcon in addition to Dilithium is that Falcon has significantly smaller signatures and verification keys. In fact, to the best of our knowledge, there has not previously been any lattice-based Fiat-Shamir signatures with signature and verification key sizes comparable to those for Falcon.

A limiting factor in constructing a compact Fiat-Shamir lattice-based signature scheme is that it must produce signatures that follow some distribution that is independent of the scheme’s secret. To accomplish this

using the Fiat-Shamir with aborts approach, the scheme initially produce candidate signatures from a distribution that depends on the secret of the scheme. Only some of these signatures are actually emitted, with the scheme otherwise aborting the signing attempt and retrying. This corresponds to rejection sampling from a secret-dependent distribution to produce signatures that follow a secret-independent distribution. However, unless the output distribution is sufficiently wide in relation to the size of the secret, this rejection sampling is too inefficient for it to be useable.

Better rejection sampling can allow for a narrower output distribution, and thus smaller signatures, while keeping the rejection probability the same. Furthermore, smaller signatures are harder to forge, meaning that, for a comparable security level, many other parameters of the scheme may be selected differently. As such, a better rejection sampling method can influence many different aspects of the scheme.

An example of the impact of improved rejection sampling is with BLISS [16], where rejection sampling from a bimodal Gaussian distribution allows for a significantly more compact scheme than Lyubashevsky’s original approach. A more recent example is HAETAE [10, 11] which relies on bimodal rejection sampling from uniform distributions over hyperballs to allow for a more compact signature scheme than Dilithium. It has even been shown in [13] that the approaches of BLISS and HAETAE are essentially optimal for schemes that rely on rejection sampling from bimodal distributions, and that rejection sampling from bimodal distributions is superior to similar sampling from unimodal distributions.

Works have also explored the possibility of removing the rejection condition [7]. A concrete example of this is Raccoon [12], which was a submission to the first round of NIST’s standardization process for additional post-quantum secure signatures. The Raccoon parameters are selected to ensure that rejection conditions can be securely ignored. This does, however, come at the cost of signatures that are about five times larger than those of Dilithium.

An alternative approach for constructing signatures without aborts was developed for the G+G signature scheme [14]. Signatures from this scheme follow a discrete Gaussian distribution which is constructed as a convolution of two different discrete Gaussian distributions, both of which have covariance that depend on the secret key of the scheme. By carefully selecting parameters, the signatures follow a centered spherical discrete Gaussian distribution, and the signatures thus do not leak anything about the secret.

The G+G scheme is thus constructed without relying on aborts, and the resulting scheme is relatively compact. The G+G scheme is, however, still far from as compact as Falcon, and it also seems like implementing G+G securely would be non-trivial.

1.1 Our Contribution

In this work we present an alternative method for constructing lattice-based signatures with the Fiat-Shamir with aborts paradigm. By constructing signatures in a way that is influenced by the rejection condition, our new method has a rejection probability that is significantly smaller than that of previous methods. A lower rejection probability allows for greater freedom when parametrizing the scheme, allowing us to construct a signature scheme that is significantly more compact than previous lattice-

based Fiat-Shamir signature schemes.

We propose concrete parametrization of a signature scheme that makes use of our new method. The resulting scheme has signature size around one third of that for ML-DSA, and verification key size somewhat smaller than that for ML-DSA. Additionally, the combined size of a signature and a verification key is less than half of what it is for ML-DSA and within 10% to that of Falcon.

We also consider alternative parametrization of our scheme where the rejection condition can be safely ignored. In a similar manner as for Raccoon [12], this is accomplished by using a wider output distribution. However, the penalty that we pay, in the form of increased size, is significantly smaller than for Raccoon, and our resulting scheme without rejection sampling is still more compact than previous lattice-based Fiat-Shamir signature schemes. In particular, the combined size of a signature and a verification key for the resulting scheme is less than 60% to that of ML-DSA.

1.2 Technical overview

The idea behind Lyubashevsky’s signature scheme [23, 24] is similar to the idea behind Schnorr signatures [28]. To sign a message with Lyubashevsky’s scheme, a vector \mathbf{y} is sampled from some relatively narrow distribution. Then, a commitment $\mathbf{w} = \mathbf{A}\mathbf{y} \bmod q$ is computed, where \mathbf{A} is a matrix that is part of the verification key and q is a parameter for the scheme.

Using the commitment \mathbf{w} and the message to be signed, a challenge \mathbf{c} is derived, with this challenge guaranteed to be a short vector. A signature is given by $(\mathbf{z}, \mathbf{c}, \mathbf{w})$, where $\mathbf{z} = \mathbf{y} + \mathbf{S}\mathbf{c} \bmod q$ with \mathbf{S} the secret key of the signer. The secret key \mathbf{S} is sampled to have short columns, and thus $\mathbf{S}\mathbf{c}$ is relatively short, ensuring that \mathbf{z} also is a relatively short vector.

In contrast to the analogous case for typical Schnorr signatures, in the lattice setting, the vector \mathbf{y} does not perfectly mask the contribution of $\mathbf{S}\mathbf{c}$ to \mathbf{z} . This is the case as \mathbf{z} , and therefore also \mathbf{y} , must be relatively short vectors in order to ensure that signatures are hard to forge. As such, the distribution of signatures produced in this manner depend on \mathbf{S} and may therefore leak information about the secret key of the scheme.

To ensure that the distribution of signatures do not depend on secret information, Lyubashevsky [23, 24] does not output all constructed signatures. Instead, signatures are only emitted with some probability that depends on \mathbf{z} , and otherwise the signing attempt is aborted and retried with a different \mathbf{y} . By aborting with a suitable probability, it is ensured that the \mathbf{z} vector in emitted signatures follow some secret-independent distribution. As such, emitted signatures do not leak any information about \mathbf{S} .

A variant of Lyubashevsky’s scheme was later considered for the signature scheme BLISS [16]. In BLISS, valid signatures can be constructed with \mathbf{z} either as $\mathbf{y} + \mathbf{S}\mathbf{c}$ or as $\mathbf{y} - \mathbf{S}\mathbf{c}$, with each of these two constructions for \mathbf{z} selected with probability 1/2. As with Lyubashevsky’s original scheme, BLISS occasionally aborts and does not output the constructed \mathbf{z} .

As \mathbf{y} is sampled from a discrete Gaussian distribution in BLISS, without rejection sampling, the distribution of BLISS signatures follows a bimodal discrete Gaussian distribution, with the two centers $\pm\mathbf{S}\mathbf{c}$. This is in contrast to the scheme of Lyubashevsky [24] where the distribution of signatures would be a unimodal Gaussian distribution with center $\mathbf{S}\mathbf{c}$ if

no rejection sampling was used.

In both the unimodal and bimodal cases, a suitable rejection condition allows the output signatures to follow a centered discrete Gaussian distribution. However, the bimodal discrete Gaussian distribution is significantly more similar to the desired output distribution than a unimodal discrete Gaussian distribution centered on \mathbf{Sc} . This gives the bimodal approach of BLISS a significant advantage over the original unimodal approach of Lyubashevsky.

1.2.1 Our Improved Signature Construction

The rejection sampling in BLISS can be described through a rejection function $R(\mathbf{z})$, with each constructed \mathbf{z} accepted with probability $R(\mathbf{z})$ and otherwise rejected. For a sampled vector \mathbf{y} , the signature is given by $\mathbf{z} = \mathbf{y} + \mathbf{Sc}$ with probability $R(\mathbf{y} + \mathbf{Sc})/2 = f_{\mathbf{Sc}}(\mathbf{y})$ and by $\mathbf{z} = \mathbf{y} - \mathbf{Sc}$ with probability $R(\mathbf{y} - \mathbf{Sc})/2 = g_{\mathbf{Sc}}(\mathbf{y})$, while it otherwise rejects the signature and tries again with a different \mathbf{y} .

With the rejection sampling formulated in this way, it is natural to consider which other choices for $f_{\mathbf{Sc}}$ and $g_{\mathbf{Sc}}$ are possible. More general functions $f_{\mathbf{Sc}}$ and $g_{\mathbf{Sc}}$ can determine both the choice of $\mathbf{y} \pm \mathbf{Sc}$ and whether or not to reject. In particular, this allows how frequently \mathbf{z} is constructed to be influenced by the rejection condition for \mathbf{z} . Note however that this requires quite a bit of care, as changing how often \mathbf{z} is constructed must also alter the rejection condition for \mathbf{z} if the output is to follow the desired distribution.

In this work, we consider such more general functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$, with a focus on the case where the input distribution for \mathbf{y} is a discrete Gaussian distribution while ensuring that, conditioned on $\mathbf{z} \neq \perp$ the output \mathbf{z} follow the same distribution. By carefully crafting $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$, this allows for rejection sampling with significantly smaller rejection probability than for BLISS.

The rejection failure probability of both our new approach and the approach of BLISS depends on the norm of vectors \mathbf{v} used to construct $\mathbf{z} = \mathbf{y} \pm \mathbf{v}$. In particular, it depends on the quotient $\alpha = r/\|\mathbf{v}\|$ between the Gaussian parameter r and the norm of the vector \mathbf{v} of maximal norm that the rejection sampling can handle. As the rejection sampling handles vectors $\mathbf{v} = \mathbf{Sc}$, the rejection probability depends on the maximal norm of \mathbf{Sc} . The signature size of the scheme essentially scales linearly with $\log(r)$ and thus also with $\log(\alpha)$. As such, if the rejection sampling allows for a smaller α to be used, it would also allow for a more compact signature scheme.

When targeting the same rejection probability, our new approach can use a smaller α than previous approaches, and thus allows for smaller signatures. One could hope that our new approach allows us to use a significantly smaller α , and therefore enables a scheme with significantly smaller signatures. However, decreasing α quickly results in unacceptably large rejection probabilities for our new approach, even though it still does have an advantage over previous rejection sampling methods.

Although our new method has a slight advantage for small α , its advantage is much more pronounced for larger α . For instance, with α such that the old approach has a rejection probability of a bit more than 10% our new approach has to reject less than one in a million signatures. Although this is a large difference in rejection probability, in practice there is only a small performance difference between rejecting 10% of the time

and rejecting only very rarely. However, by iteratively constructing \mathbf{z} , we can exploit this significant advantage for larger α to allow for much more compact signatures.

1.2.2 Iterative Rejection Sampling

Our new signature scheme is quite similar to BLISS, with an analogous relation between public and private keys, and with both \mathbf{y} and \mathbf{z} following a discrete Gaussian distribution. Furthermore, the space \mathcal{C} of possible challenges \mathbf{c} consist of vectors where all elements are either zero or one and which often are quite sparse. As noted by Ducas [15], and more recently used for the G+G signature scheme [14], this allows signatures from the scheme to be constructed as $\mathbf{y} + \mathbf{S}\mathbf{c}'$ for any $\mathbf{c}' \equiv \mathbf{c} \pmod{2}$.

For our work, we make use of this flexibility in how \mathbf{c}' can be selected to iteratively rejection sample over the coefficients of \mathbf{c} . This consists of selecting the sign of each coefficient of \mathbf{c} independently, corresponding to either producing $\mathbf{y} + \mathbf{v}_i$ or $\mathbf{y} - \mathbf{v}_i$ where \mathbf{v}_i is a column of \mathbf{S} .

To construct \mathbf{z} , the iterative process starts with $\mathbf{z}_0 = \mathbf{y}$, performs rejection sampling with an input \mathbf{v}_1 to get \mathbf{z}_1 such that, conditioned on $\mathbf{z}_1 \neq \perp$, it follows the same distribution as \mathbf{z}_0 . Repeating this leads to a sequence of vectors $\mathbf{y} = \mathbf{z}_0, \dots, \mathbf{z}_\kappa = \mathbf{z}$ such that, for every $i \in \{1, \dots, \kappa\}$ and conditioned on $\mathbf{z}_i \neq \perp$, \mathbf{z}_i follows the same distribution as \mathbf{y} . Thus, the output \mathbf{z} follows the desired distribution and there are $a_i \in \{0, 1\}$ such that

$$\mathbf{z} = \mathbf{y} + \sum_{i=1}^{\kappa} (-1)^{a_i} \mathbf{v}_i = \mathbf{y} + \mathbf{S}\mathbf{c}'$$

for some $\mathbf{c}' \equiv \mathbf{c} \pmod{2}$.

Each vector \mathbf{v}_i added in the iterative steps is a column of \mathbf{S} and is therefore expected to be significantly shorter than $\mathbf{S}\mathbf{c}$. Each rejection sampling step thus use a larger α than would be used for a single-step construction of \mathbf{z} . Although this requires multiple steps succeeding, the larger α from the iterative construction combined with our new rejection sampling allows targeting a significantly narrower output distribution than previous approaches.

1.3 Concrete Signature Scheme

For efficiency and compactness, we base our signature scheme on structured lattice problems. In particular, we work with elements in the ring $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$. We additionally limit ourself to n that are powers of two to enable efficient computations through the Number Theoretic Transform (NTT), with a particular focus on $n = 256$.

For our scheme, we let the challenge space \mathcal{C} be a subset of \mathcal{R} and the secret \mathbf{S} be a vector $\mathbf{s} \in \mathcal{R}^k$, where k is a relatively small integer typically not larger than 10. Similar to BLISS [16] and G+G [14], we generate keys such that $\mathbf{A}\mathbf{s} = \mathbf{q}\mathbf{j} \pmod{2q}$ where $\mathbf{A} \in \mathcal{R}^{m \times k}$ is the verification key with $m \approx k$ and $\mathbf{j} = [1, 0, \dots, 0]^T$ the first unit vector.

A signature from the scheme consist of (\mathbf{z}, c) for \mathbf{z} a relatively short vector such that

$$\mathcal{H}(\mathbf{A}\mathbf{z} - \mathbf{q}\mathbf{c}\mathbf{j} \pmod{2q}, \mu) = c$$

where μ is the signed message and \mathcal{H} is a cryptographic hash function. To construct such a signature, the signer samples \mathbf{y} from some relatively narrow distribution. From this \mathbf{y} , the signer computes $\mathbf{w} = \mathbf{A}\mathbf{y} \pmod{2q}$ and derives c via $\mathcal{H}(\mathbf{w}, \mu)$.

Since $\mathbf{A}\mathbf{s} = q\mathbf{j} \bmod 2q$, constructing \mathbf{z} as $\mathbf{y} + \mathbf{s}c$ leads to a valid signature, as $\mathbf{A}\mathbf{z} \equiv \mathbf{w} + qc\mathbf{j} \bmod 2q$. Furthermore, it follows that for any $\mathbf{c}' \equiv \mathbf{c} \bmod 2$, then $\mathbf{z}' = \mathbf{y} + \mathbf{s}c'$ also gives a valid signature as long as \mathbf{z}' is short enough.

The challenge space \mathcal{C} is selected to only contain ring elements with at most κ non-zero coefficients, with all non-zero elements equal to 1. Thus, $\mathbf{s}c$ is equal to the sum of at most κ terms of the form $x^j \mathbf{s}$, and where we are allowed to select the sign for each of these terms when constructing our signature. Each of these terms $x^j \mathbf{s}$ have the same length, namely $\|\mathbf{s}\|$, and our rejection probability depends on this norm.

For simplicity we ensure that all secret keys of our scheme have the same rejection probability, which is determined by the maximal value of $\|\mathbf{s}\|$ over all possible secrets. The secret keys for our scheme are therefore sampled with a length bound B , guaranteeing that $\|\mathbf{s}\| \leq B$. In practice, this is accomplished by sampling \mathbf{s} from some other distribution and retrying if $\|\mathbf{s}\| > B$.

The bound B on the length of the secret key indirectly impacts the size of signatures from the scheme. With the distributions that \mathbf{y} and \mathbf{z} follow left unchanged, increasing the bound B results in a larger rejection probability. To increase B without altering the rejection probability, \mathbf{y} and \mathbf{z} must follow wider distributions, leading to larger signatures.

1.3.1 Security Against Forgeries

To argue for our scheme's security against forgeries, we assume that the \mathbf{A} matrix is constructed from elements that are computationally hard to distinguish from uniformly random. In the random oracle model, it can then be proven that an efficient algorithm for forging signatures implies that the Module Short Integer Solutions (MSIS) problem is easy. Thus, our scheme's security against forgeries relies upon the assumed hardness of the MSIS problem.

For the MSIS problem, a matrix \mathbf{A} is sampled uniformly at random modulo q and the task is to recover a non-zero vector \mathbf{x} such that $\mathbf{A}\mathbf{x} \equiv \mathbf{0} \bmod q$ with $\|\mathbf{x}\| \leq B$ for some length bound B . The concrete hardness of the MSIS problem is in large part determined by this length bound B . Furthermore, for certain parameter choices, the MSIS problem can be proven to be at least as hard as standard lattice problems restricted to module lattices [22].

Given an efficient algorithm for forging signatures, the security reduction for the signature scheme can be used to produce short vectors \mathbf{x} such that $\mathbf{A}\mathbf{x} \equiv \mathbf{0} \bmod q$. The length of the produced \mathbf{x} vector is directly related to the length of the \mathbf{z} vector in the forged signatures. Thus, if \mathbf{z} follows a narrower distribution, the reduction produce shorter vectors and thus solve the MSIS problem with a smaller length bound. As such, \mathbf{z} following a narrower distribution allows the security of the signature scheme to be based on a harder version of the MSIS problem.

Besides the width of the distribution of \mathbf{z} , the hardness of the underlying MSIS problem also depends on the dimension of $\mathbf{A} \in \mathcal{R}^{m \times k}$, with a larger m leading to a harder problem. The value of k can also have some effect on the hardness of the SIS problem, with a larger k sometimes making the problem easier. However, a more significant reason to prefer a smaller k is that \mathbf{z} is a k -dimensional vector, and the size of signatures therefore directly depend on k .

1.3.2 Secure Key Generation

As described above, for the analysis of the scheme's security against forgeries, it is assumed that the public key is constructed from elements that are computationally hard to distinguish from uniformly random. There are multiple alternatives for how to sample the secret vector \mathbf{s} and the verification key \mathbf{A} to achieve this.

A natural choice is to base the scheme on the Module Learning With Errors (MLWE) problem, as done in for instance HAETAETAE [10, 11] and G+G [14]. In such a scheme, small $\mathbf{s}_0 \in \mathcal{R}^\ell$ and $\mathbf{e} \in \mathcal{R}^m$ are sampled as the secret of the scheme and $\mathbf{A}_0 \in \mathcal{R}^{m \times \ell}$ is sampled uniformly at random modulo q . The verification key is then given by \mathbf{A}_0, \mathbf{b} where $\mathbf{b} = \mathbf{A}_0 \mathbf{s}_0 + \mathbf{e} \pmod q$, and the public matrix \mathbf{A} and secret vector \mathbf{s} are given as

$$\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}_m] \text{ and } \mathbf{s} = [1, \mathbf{s}_0^T, \mathbf{e}^T]^T.$$

This results in the \mathbf{z} part of the signatures being an $(\ell+m+1)$ -dimensional vector over \mathcal{R} . As such, \mathbf{z} has a total of $(\ell+m+1)n$ coefficients, all of which are relatively small integers.

For a more compact scheme, we could instead base the scheme on NTRU lattices, similar to how it is done for BLISS [16]. To target a similar security level, the ring $\mathcal{R}' = \mathbb{Z}[x]/(x^{n'} + 1)$, with $n' = \ell n$ would have to be used. The key generation would then sample small $f, g \in \mathcal{R}'$ as the private key and compute $h = (2g + 1)f^{-1} \pmod q$ as the public key. The public matrix \mathbf{A} and secret vector \mathbf{s} are then given by

$$\mathbf{A} = [2h, q - 2] \text{ and } \mathbf{s} = [f, 2g + 1]^T$$

and as detailed in [16], this leads to $\mathbf{A}\mathbf{s} = q \pmod{2q}$.

For such an NTRU-based scheme, \mathbf{z} has $2n' = 2\ell n$ coefficients. Meanwhile, parametrizing an MLWE-based version of the scheme with $m = \ell$ results in \mathbf{z} having $(2\ell + 1)n$ coefficients. When targeting similar rejection probability, the coefficients of \mathbf{z} for the NTRU-based scheme have similar size as those for an MLWE-based version. As such, signatures of an NTRU-based version of the scheme are somewhat smaller than those of a comparable MLWE-based scheme. However, the NTRU structure limits the flexibility of parameter selection as, in contrast to an MLWE-based scheme, it is not directly possible to select ℓ and m independently from each other. Additionally, for the most efficient NTT calculations n' is limited to powers of two, further limiting the flexibility of parameter selection.

To get the compact signatures of an NTRU based scheme with the flexibility of an MLWE based scheme, we instead rely on the NTWE problem introduced by Gärtner [20]¹. The secret key of such an NTWE-based scheme consists of $\mathbf{s}_0 \in \mathcal{R}^{\ell'}$, $\mathbf{e} \in \mathcal{R}^m$ and $f \in \mathcal{R}$, where $f = 2f_0 + 1$ with the elements of \mathbf{s}_0, \mathbf{e} and f_0 sampled from some narrow distribution. The verification key of such a scheme consists of $(\mathbf{A}_0, \mathbf{b})$ where $\mathbf{A}_0 \in \mathcal{R}^{m \times \ell'}$ is sampled with elements uniformly at random modulo q and

$$\mathbf{b} = (\mathbf{A}_0 \mathbf{s}_0 + \mathbf{e})f^{-1} \pmod q.$$

The public matrix and private vector are then given by

$$\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}_m] \pmod{2q} \text{ and } \mathbf{s} = [f, \mathbf{s}_0^T, \mathbf{e}^T]^T$$

¹This same problem was actually first implicitly used for the signature scheme of [2], although without being formally defined. Followup work from this led to a later independent reintroduction of the NTWE problem under the name v-MNTRU in [4]

which ensures that $\mathbf{A}\mathbf{s} = q\mathbf{j} \bmod 2q$, as desired.

To target a comparable security level, this NTWE-based version can be parametrized with $\ell' = \ell - 1$ with the coefficients of \mathbf{z} still having similar sizes as the ones in the MLWE and NTRU-based versions. Thus, \mathbf{z} can be represented by $n(\ell' + m + 1) = n\ell + nm$ relatively small integers and, with $m = \ell$, the size of \mathbf{z} for this NTWE-based version is on par with the NTRU-based version. Furthermore, as with the MLWE based scheme, we can keep n fixed to some power of two and freely change ℓ' and m in order to alter the targeted security level. As such, this NTWE-based scheme provides the flexibility of an MLWE-based scheme with the compactness of an NTRU-based scheme.

2 Preliminaries

2.1 Notation

We denote matrices by bold upper case letters and vectors by bold lower case letters. We use $\mathbf{j} = [1, 0, \dots, 0]^T$ to denote the first unit vector, with its dimension implicit from the context. The uniform distribution over a set \mathcal{S} we denote by $\mathcal{U}(\mathcal{S})$ and taking a sample x from a distribution \mathcal{D} we denote by $x \leftarrow \mathcal{D}$.

We primarily make use of the Euclidean ℓ_2 norm, which we denote by $\|\cdot\|$. We additionally use the infinity norm which we denote by $\|\cdot\|_\infty$.

Our signature scheme works in the ring $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$ where n is a power of two. To each element in \mathcal{R} we associate an n -dimensional coefficient vector. Vectors in \mathcal{R}^k can thus be associated with an nk -dimensional vector consisting of the coefficients of the k ring elements. We let the norm $\|\mathbf{v}\|$ of a vector $\mathbf{v} \in \mathcal{R}^k$ be the norm of this coefficient vector.

We make use both of standard modular reduction $y = x \bmod p$ such that y is the unique value in $[0, p)$ that satisfies $y = x + kp$ for some integer k . Additionally we use the centered modular reduction $y = x \bmod^\pm p$ with y the unique value in $[-p/2, p/2)$ such that $y = x + kp$ for some integer k . We also denote by $\text{LSB}(x)$ the least significant bit of x . All of these functions are naturally extended to vectors, to elements of \mathcal{R} through their coefficient vectors, and to vectors over \mathcal{R} .

For a function f and a set \mathcal{S} , we let

$$f(\mathcal{S}) = \sum_{x \in \mathcal{S}} f(x).$$

2.2 Signature Schemes

A signature scheme consist of three algorithms (KeyGen, Sign, Verify). The KeyGen algorithm outputs a verification key vk and a signing key sk . The signing key sk can be used to sign a message μ via $\text{sig} \leftarrow \text{Sign}(\text{sk}, \mu)$ and the verification key can then verify the signature via $\text{Verify}(\text{sig}, \text{vk}, \mu)$. With the signature scheme we consider, for valid keys $(\text{vk}, \text{sk}) \leftarrow \text{KeyGen}$, every signature produced via $\text{sig} \leftarrow \text{Sign}(\text{sk}, \mu)$ is such that the verification $\text{Verify}(\text{sig}, \text{vk}, \mu)$ succeeds by outputting 1.

The security of the scheme relies on it being hard to forge a signature that passes verification. Three different security notions are relevant for our signature scheme and the proof of its security.

The weakest of the security notions we consider, which is only used in proving the stronger security properties, is unforgeability under no message attacks (UF-NMA). The advantage of an adversary \mathcal{A} in the UF-NMA security game is given by

$$\Pr [\text{Verify}(\text{sig}, \text{vk}, \mu) = 1 \mid (\text{sig}, \mu) \leftarrow \mathcal{A}(\text{vk})]$$

where vk is a verification key given by KeyGen .

A stronger security notion is unforgeability under chosen message attacks (UF-CMA). The advantage of an adversary in this security game is similar to that of the UF-NMA game, but the adversary is additionally given access to an oracle that signs any message chosen by the adversary. However, the adversary is required to provide a forgery for a message for which the oracle has not provided a signature.

The strongest security notion we consider is strong unforgeability under chosen message attacks (sUF-CMA). The advantage of an adversary in this security game is defined almost the same as for the UF-CMA game, but the adversary is now successful as long as it provides a valid signature which it has not received from the oracle, even if the forgery is for a message which the oracle has provided a different signature for.

2.3 Lattice Assumptions

The security of our scheme can be based on the assumed hardness of the decision NTWE problem and the MSIS problem. The decision NTWE problem and normal form MSIS problem that we rely upon are given by the following definitions.

Definition 1 (NTWE distribution). *Let q be a prime, m and ℓ be positive integers and \mathcal{D} be some distribution over \mathcal{R} . Furthermore, let \mathbf{s} be a vector in \mathcal{R}^ℓ and f be an invertible element of \mathcal{R}_q . Then, a sample from the NTWE distribution $\mathcal{W}_{\mathbf{s}, f, \mathcal{D}}^m$ is given by (\mathbf{A}, \mathbf{b}) for $\mathbf{A} \leftarrow \mathcal{U}(\mathcal{R}_q^{m \times \ell})$ and $\mathbf{b} = (\mathbf{A}\mathbf{s} + \mathbf{e})f^{-1} \bmod q$ where $\mathbf{e} \leftarrow \mathcal{D}^m$.*

Definition 2 (Decision NTWE problem). *The advantage of an adversary \mathcal{A} against the $\text{NTWE}_{q, m, \ell, \mathcal{D}}$ problem is given by*

$$\left| \frac{\Pr [\mathcal{A}(\mathbf{A}, \mathbf{b}) = 1 \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m \times \ell}, \mathbf{b} \leftarrow \mathcal{R}_q^m]}{\Pr [\mathcal{A}(\mathbf{A}, \mathbf{b}) = 1 \mid (\mathbf{A}, \mathbf{b}) \leftarrow \mathcal{W}_{\mathbf{s}, f, \mathcal{D}}^m]} - \right|$$

where \mathbf{s} is sampled from \mathcal{D}^ℓ and $f = 2f_0 + 1$ with f_0 sampled from \mathcal{D} conditioned on f being invertible in \mathcal{R}_q .

Definition 3 (Normal form MSIS problem). *Let q be a prime, m and k be positive integers and $B > 0$ be some bound. The advantage of an adversary \mathcal{A} against the $\text{MSIS}_{q, m, k, B}$ problem is given by*

$$\Pr \left[\|\mathbf{y}\| \leq B \wedge [\mathbf{A}, \mathbf{I}_m] \cdot \mathbf{y} \equiv 0 \pmod{q} \mid \mathbf{A} \leftarrow \mathcal{R}_q^{m \times k}, \mathbf{y} \leftarrow \mathcal{A}(\mathbf{A}) \right].$$

The UF-NMA security of our scheme in the random oracle model (ROM) can be based on the assumed hardness of the MSIS problem by making use of the forking lemma. However, to construct a more efficient scheme, we instead rely on a special self-target variant of the MSIS problem. This same approach is taken by most other similar lattice-based signature schemes that are constructed using the Fiat-Shamir paradigm.

Such a self-target variant of the MSIS problem could potentially be easier than the ordinary MSIS problem with the same parameters. However, as in previous works, we assume that the self-target MSIS problem

actually is as hard as the corresponding MSIS problem. This is motivated by the fact that the only known way to solve this self-target MSIS problem is through solving the corresponding MSIS problem. More specifically, we rely on a bimodal self target MSIS problem defined below, which is the same problem that HAETAE [10, 11] relies on.

This BimodalSelfTargetMSIS problem is parametrized by a size bound $B > 0$ and integers $q, m, \ell > 0$. Furthermore, it depends on a hash function \mathcal{H} from $\mathcal{R}_{2q}^m \times \mathcal{M}$ to \mathcal{C} where $\mathcal{M} \subset \{0, 1\}^*$ is a message space and $\mathcal{C} \subseteq \mathcal{R}_2$ is a challenge space.

Definition 4 (BimodalSelfTargetMSIS $_{\mathcal{H}, q, m, \ell, B}$ problem). *An adversary \mathcal{A} against the BimodalSelfTargetMSIS $_{\mathcal{H}, q, m, \ell, B}$ problem is given $\mathbf{A} \in \mathcal{R}^{m \times \ell}$ and $\mathbf{b} \in \mathcal{R}^m$ with elements sampled uniformly at random in \mathcal{R}_q . The adversary is to produce $(\mathbf{z}, c, \mu) \in (\mathcal{R}^{m+\ell+1}, \mathcal{C}, \mathcal{M})$ such that $\mathbf{z} \neq \mathbf{0}$, $\|\mathbf{z}\| < B$ and*

$$\mathcal{H}(\mathbf{A}\mathbf{z} - qc\mathbf{j} \bmod 2q, \mu) = c$$

where $\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}] \bmod 2q$.

2.4 Fiat-Shamir Transform

The Fiat-Shamir transform allows transforming an identification scheme into a signature scheme. The security of the resulting signature scheme requires that the underlying identification scheme has the following zero-knowledge property.

Definition 5 (paHVZK). *Let ID be a canonical identification scheme with the scheme producing some output distribution of $(\mathbf{w}, c, \mathbf{z})$. The scheme is Perfect Accepting Honest Verifier Zero-Knowledge (paHVZK) if there exist a probabilistic polynomial time simulator that, when given only the verification key of the identification scheme, outputs $(\mathbf{w}', c', \mathbf{z}')$ following the same distribution as the output $(\mathbf{w}, c, \mathbf{z})$ conditioned on $\mathbf{z} \neq \perp$.*

If the identification scheme is paHVZK, the UF-CMA security of the transformed signature scheme directly follows from its UF-NMA security, via Theorem 1 below. This same theorem also proves the sUF-CMA security of our scheme assuming that no adversary has a noticeable advantage against the computationally unique response (CUR) property, as defined below, of the underlying identification scheme.

Definition 6 (Computationally Unique Response (CUR)). *Let ID be a canonical identification scheme with instance generator Gen and let $(x, y) \leftarrow Gen$, with x the public part of the key for the prover. The advantage of an adversary \mathcal{A} against the CUR property of the scheme is given by the probability that $\mathcal{A}(x)$ outputs a commitment w , a challenge c and two separate responses z and z' such that the verifier accepts both (w, c, z) and (w, c, z') as valid transcripts.*

The theorem we use to prove the UF-CMA and sUF-CMA security of our scheme is an adaptation of Theorem 2 of [6]. Our adaptation of the original theorem is similar to that of HAETAE [10].

Theorem 1 (Adapted from Theorem 2 of [6]). *Let ID be a identification scheme with commitment min-entropy γ that satisfies paHVZK with probability of aborting β . Let \mathcal{A} be a quantum UF-CMA attacker against the Fiat-Shamir transform of ID with random oracle H . Furthermore, let q_S and q_H be the number of queries \mathcal{A} makes to the signing oracle and*

to the random oracle H respectively. Then, there exists a quantum NMA attacker \mathcal{B} such that

$$\begin{aligned} \text{Adv}^{\text{UF-CMA}}(\mathcal{A}) \leq \text{Adv}^{\text{UF-NMA}}(\mathcal{B}) &+ \frac{2q_S 2^{-\gamma/2}}{1-\beta} \sqrt{q_H + 1 + \frac{q_S}{1-\beta}} \\ &+ 2^{-\gamma/2+1} (q_H + 1) \sqrt{\frac{q_S}{1-\beta}}. \end{aligned}$$

Furthermore, if \mathcal{A} is an adversary against the sUF-CMA security, there exist an adversary \mathcal{B}' against the CUR property of ID such that the previous bound holds by adding $\text{Adv}_{\text{ID}}^{\text{CUR}}(\mathcal{B}')$ on the right hand side. The running time of \mathcal{B} and \mathcal{B}' is approximately that of \mathcal{A} plus q_S times the running time of the paHVZK simulator.

2.5 Discrete Gaussian Distributions

We use $\rho_r(\mathbf{v})$ to denote the Gaussian function $\exp(-\|\mathbf{v}\|^2/(2r^2))$, and let $\rho(\mathbf{v}) = \rho_1(\mathbf{v})$. The discrete Gaussian distribution $\mathcal{D}_{\mathcal{L},r}$ with parameter r over a lattice \mathcal{L} is the distribution where the probability of a vector $\mathbf{v} \in \mathcal{L}$ is given by

$$\frac{\rho_r(\mathbf{v})}{\sum_{\mathbf{w} \in \mathcal{L}} \rho_r(\mathbf{w})} = \frac{\rho_r(\mathbf{v})}{\rho_r(\mathcal{L})}$$

and the probability of $\mathbf{v} \notin \mathcal{L}$ is 0. We also consider discrete Gaussian distributions over the ring $\mathcal{R} = \mathbb{Z}[x]/(x^n + 1)$, where we sample the coefficient vector of an element in \mathcal{R} . This also naturally extends to sampling vectors over \mathcal{R} .

To argue that the signatures produced do not have too large norm, we make use of the following lemma of Banaszczyk to bound the norm of vectors sampled from discrete Gaussian distributions.

Lemma 2 (Lemma 1.5 from [5]). *Let $C \geq 1$, $r > 0$ be reals and $n \geq 1$ be an integer. Then*

$$\Pr [\|\mathbf{z}\| > Cr\sqrt{n} \mid \mathbf{z} \leftarrow \mathcal{D}_{\mathbb{Z}^n, r}] \leq \left(C \cdot \exp\left(\frac{1-C^2}{2}\right) \right)^n$$

3 Our New Rejection Sampling Method

In this section we describe our new rejection sampling method and its advantages over previous methods. Given a sample \mathbf{y} from some distribution and some vector \mathbf{v} with limited norm, the rejection sampling should either abort and output \perp or output a vector \mathbf{z} such that $\mathbf{z} = \mathbf{y} + k\mathbf{v}$ for arbitrary odd integer k . Additionally, conditioned on $\mathbf{z} \neq \perp$, the output from the rejection sampling should follow some distribution that is independent of \mathbf{v} . Quite similar to bimodal rejection sampling, unless it aborts, our sampling outputs either $\mathbf{y} + \mathbf{v}$ or $\mathbf{y} - \mathbf{v}$.

The novelty of our new method is that the rejection sampling is given more freedom in selecting whether to output $\mathbf{y} + \mathbf{v}$ or $\mathbf{y} - \mathbf{v}$. Previous methods for rejection sampling from bimodal distributions first randomly selects $\mathbf{y} + \mathbf{v}$ or $\mathbf{y} - \mathbf{v}$ with equal probability and then either rejects or outputs the result. We instead select the output via the procedure $R_{\mathbf{v}}(\mathbf{y})$ in Figure 1 for more general functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$.

For the output of our new rejection sampling to be independent of the vector \mathbf{v} , the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ must satisfy certain properties. The following lemma details conditions on these functions under which the output of $R_{\mathbf{v}}(\mathbf{y})$ follows some given distribution \mathcal{Z} .

$$\begin{array}{l}
R_{\mathbf{v}}(\mathbf{y}) \\
\hline
1: (a, b) = (f_{\mathbf{v}}(\mathbf{y}), g_{\mathbf{v}}(\mathbf{y})) \\
2: r \leftarrow \mathcal{U}([0, 1]) \\
3: \text{if } r < a \\
4: \quad \text{return } \mathbf{y} - \mathbf{v} \\
5: \text{if } r < a + b \\
6: \quad \text{return } \mathbf{y} + \mathbf{v} \\
7: \text{return } \perp
\end{array}$$

Figure 1: New method for rejection sampling, dependent on functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$.

Lemma 3. Let \mathcal{Y}, \mathcal{Z} be some distributions with $p_{\mathbf{y}}(\mathbf{y})$ the probability of $\mathbf{y} \leftarrow \mathcal{Y}$ and $p_{\mathbf{z}}(\mathbf{z})$ the probability of $\mathbf{z} \leftarrow \mathcal{Z}$ and let \mathbf{v} be some fixed non-zero vector. Furthermore, let $f_{\mathbf{v}}(\mathbf{z})$ and $g_{\mathbf{v}}(\mathbf{z})$ be functions and $M \geq 1$ be a parameter such that

$$f_{\mathbf{v}}(\mathbf{z}) + g_{\mathbf{v}}(\mathbf{z}) \leq 1 \quad (1)$$

$$f_{\mathbf{v}}(\mathbf{z}) \geq 0, \quad g_{\mathbf{v}}(\mathbf{z}) \geq 0 \quad (2)$$

$$p_{\mathbf{y}}(\mathbf{z} + \mathbf{v})f_{\mathbf{v}}(\mathbf{z} + \mathbf{v}) + p_{\mathbf{y}}(\mathbf{z} - \mathbf{v})g_{\mathbf{v}}(\mathbf{z} - \mathbf{v}) = \frac{p_{\mathbf{z}}(\mathbf{z})}{M} \quad (3)$$

for all \mathbf{z} . Then, the distribution of $\mathbf{z} \leftarrow R_{\mathbf{v}}(\mathbf{y})$ conditioned on $\mathbf{z} \neq \perp$ is \mathcal{Z} and the probability that $\mathbf{z} = \perp$ is $1 - 1/M$.

Proof. Based on the properties of $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ we are guaranteed that $R_{\mathbf{v}}(\mathbf{y})$ returns $\mathbf{y} - \mathbf{v}$ with probability $f_{\mathbf{v}}(\mathbf{y})$ and $\mathbf{y} + \mathbf{v}$ with probability $g_{\mathbf{v}}(\mathbf{y})$. The probability of an input \mathbf{y} is $p_{\mathbf{y}}(\mathbf{y})$, and the probability of an output \mathbf{z} is therefore

$$p_{\mathbf{y}}(\mathbf{z} + \mathbf{v})f_{\mathbf{v}}(\mathbf{z} + \mathbf{v}) + p_{\mathbf{y}}(\mathbf{z} - \mathbf{v})g_{\mathbf{v}}(\mathbf{z} - \mathbf{v}),$$

which via (3) is equal to $p_{\mathbf{z}}(\mathbf{z})/M$. As this is a scaling of the desired output probability distribution, this gives that conditioned on $\mathbf{z} \neq \perp$, the output is distributed as if from \mathcal{Z} .

The probability that the procedure does not output \perp is given by the sum of $p_{\mathbf{z}}(\mathbf{z})/M$ over all $\mathbf{z} \neq \perp$. As $p_{\mathbf{z}}$ is a probability distribution and thus sums to 1, the probability over \mathbf{y} that $R_{\mathbf{v}}(\mathbf{y})$ outputs something other than \perp is $1/M$ and the probability of $\mathbf{z} = \perp$ is thus $1 - 1/M$. \square

In the following subsections we detail how we construct functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ that fulfill the properties of Lemma 3.

First, in Section 3.1, we consider requirements of such functions when the input and output distributions are the same distribution. In this relatively general setting, we are unable to construct $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ that are guaranteed to fulfill the properties of Lemma 3.

In Section 3.2, we consider a more specialized setting where we can adapt the construction of $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ from Section 3.1 to get functions that are guaranteed to fulfill all the properties of Lemma 3. In this more specialized setting, an outcome \mathbf{x} of the relevant distribution only depends on $\|\mathbf{x}\|$, with the probability of \mathbf{x} decreasing as $\|\mathbf{x}\|$ increases.

For our signature scheme, we rely on these functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ for the specific case when the relevant distribution is a discrete Gaussian distribution. In Section 3.3, we further analyze the resulting functions

$f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ in this setting. In particular, for this setting we somewhat simplify the definition of these functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ and show that they can be efficiently computed to sufficient precision.

Finally, in Section 3.4 we detail the concrete advantage our new rejection sampling compared to previous rejection sampling methods. In particular, we focus on a comparison to rejection sampling from bimodal Gaussian distributions, as used with BLISS, since this seems like the most relevant comparison.

3.1 Same Input and Output Distributions

For our analysis, we limit ourselves to the case where $\mathcal{Z} = \mathcal{Y}$, and thus $p_{\mathbf{y}}(\mathbf{x}) = p_{\mathbf{z}}(\mathbf{x}) = p(\mathbf{x})$ for every \mathbf{x} . Functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ that satisfy the requirements of Lemma 3 must thus satisfy that

$$\frac{p(\mathbf{z})}{M} = p(\mathbf{z} + \mathbf{v})f_{\mathbf{v}}(\mathbf{z} + \mathbf{v}) + p(\mathbf{z} - \mathbf{v})g_{\mathbf{v}}(\mathbf{z} - \mathbf{v}). \quad (4)$$

This can be seen as $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ redistributing part of the input probability on the points $\mathbf{z} + \mathbf{v}$ and $\mathbf{z} - \mathbf{v}$ to ensure the correct probability for the output \mathbf{z} .

The functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ also influence the probability of other outputs. In particular, the function $g_{\mathbf{v}}$ also distribute part of the probability of the input $\mathbf{z} + \mathbf{v}$ to ensure that the probability for the output $\mathbf{z} + 2\mathbf{v}$ is correct. Similarly, the function $f_{\mathbf{v}}$ distribute part of the probability of the input $\mathbf{z} - \mathbf{v}$ to ensure the correct probability for the output $\mathbf{z} - 2\mathbf{v}$. The dependence on other inputs cascade, and the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ thus redistribute the probability of inputs that are an odd multiple of \mathbf{v} away from \mathbf{z} to ensure the correct probability of outputs that are even multiples of \mathbf{v} away from \mathbf{z} .

For the procedure $R_{\mathbf{v}}$ to never reject, the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ must satisfy the properties of Lemma 3 with $M = 1$. For this to be possible for all points related to any given \mathbf{z} , the probability weight on inputs must equal the probability weight on outputs. This corresponds to the requirement that

$$\sum_{k \in \mathbb{Z}} p(\mathbf{z} + 2k\mathbf{v}) = \sum_{k \in \mathbb{Z}} p(\mathbf{z} + (2k + 1)\mathbf{v})$$

which is equivalent to

$$\sum_{k \in \mathbb{Z}} (-1)^k p(\mathbf{z} + k\mathbf{v}) = 0. \quad (5)$$

However, note that although this requirement is necessary, it is not sufficient for $M = 1$ to be possible. For instance, no version of $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ would work if $p(\mathbf{z}) = t \neq 0$ and $p(\mathbf{z} + 3\mathbf{v}) = t$ while $p(\mathbf{z} + k\mathbf{v}) = 0$ for all other k .

If (5) holds, there is a natural description of $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ that may satisfy the properties of Lemma 3 with $M = 1$. This pair of functions is determined by considering the consequences of (4) being satisfied for $\mathbf{z}' = \mathbf{z} + 2k\mathbf{v}$ for every integer $k \neq 0$. With (4) satisfied for every other related output, the probability weight left for $f_{\mathbf{v}}(\mathbf{z} + \mathbf{v})$ and $g_{\mathbf{v}}(\mathbf{z} - \mathbf{v})$ can be determined, and this gives our definition for these functions.

In order for (4) to be satisfied for every output of the form $\mathbf{z} + 2k\mathbf{v}$ with $k > 0$, the remaining probability weight on input $\mathbf{z} + \mathbf{v}$ is

$$p(\mathbf{z} + \mathbf{v}) - \sum_{k=2}^{\infty} (-1)^k p(\mathbf{z} + k\mathbf{v}).$$

Letting f_v distribute the remaining probability weight to the output \mathbf{z} gives that

$$f_v(\mathbf{z} + \mathbf{v}) = \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{z} + (k+1)\mathbf{v})}{p(\mathbf{z} + \mathbf{v})}$$

or equivalently

$$f_v(\mathbf{y}) = \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})}.$$

Similarly, letting the remaining probability weight on the input $\mathbf{z} - \mathbf{v}$ be distributed by g_v to $p(\mathbf{z})$ gives that

$$g_v(\mathbf{y}) = \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{y} - k\mathbf{v})}{p(\mathbf{y})}.$$

These functions fulfill

$$\begin{aligned} & f_v(\mathbf{z} + \mathbf{v})p(\mathbf{z} + \mathbf{v}) + g_v(\mathbf{z} - \mathbf{v})p(\mathbf{z} - \mathbf{v}) \\ &= \sum_{k=0}^{\infty} (-1)^k (p(\mathbf{z} + (k+1)\mathbf{v}) + p(\mathbf{z} - (k+1)\mathbf{v})) \\ &= p(\mathbf{z}) - \sum_{k \in \mathbb{Z}} (-1)^k p(\mathbf{z} + k\mathbf{v}) = p(\mathbf{z}) \end{aligned}$$

with final equality as (5) holds.

For our signature scheme, we can not directly apply this approach as (5) does not hold for every \mathbf{z} . Furthermore, even with (5) satisfied, these definitions do not guarantee that $f_v(\mathbf{z}) \geq 0$ and $g_v(\mathbf{z}) \geq 0$ for every \mathbf{z} . In the next section, we add some additional requirements on $p(\mathbf{z})$ and allow for $M > 1$. This allows us to alter the definitions of f_v and g_v so that they do fulfill all the properties of Lemma 3. As we consider $M > 1$, these altered definitions work even if (5) does not hold.

3.2 Norm-Dependent Distribution

In this section, we limit ourselves to $p(\mathbf{z})$ that depend only on the norm of \mathbf{z} and decrease with increasing norm. This is for instance the case for Gaussian distributions, which is the distribution we use for our signature scheme. With such a limitation on $p(\mathbf{z})$, we can find a suitable definition for f_v and g_v that satisfy all requirements of Lemma 3.

For this analysis, we define the function

$$S_v(\mathbf{y}) = \sum_{k \geq 0} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})}$$

which is the definition of $f_v(\mathbf{y})$ from the previous section. Furthermore, as $p(\mathbf{y})$ is only determined by the norm of \mathbf{y} , we have that

$$S_v(-\mathbf{y}) = \sum_{k \geq 0} (-1)^k \frac{p(-\mathbf{y} + k\mathbf{v})}{p(-\mathbf{y})} = \sum_{k \geq 0} (-1)^k \frac{p(\mathbf{y} - k\mathbf{v})}{p(\mathbf{y})}$$

which is the definition of $g_v(\mathbf{y})$ from the previous section.

We are guaranteed that either

$$\|\mathbf{y} + (k+1)\mathbf{v}\|^2 \geq \|\mathbf{y} + k\mathbf{v}\|^2 \text{ or } \|\mathbf{y} - (k+1)\mathbf{v}\|^2 \geq \|\mathbf{y} - k\mathbf{v}\|^2$$

for every $k \geq 0$ and we know that $p(\mathbf{x})$ is decreasing with increasing $\|\mathbf{x}\|$. From this it follows that either $S_v(\mathbf{y}) \geq 0$ or $S_v(-\mathbf{y}) \geq 0$, since for at least one of these functions, the absolute values of terms in the sum are strictly decreasing. The idea is to define both $f_v(\mathbf{y})$ and $g_v(\mathbf{y})$ based on the one of $S_v(\mathbf{y})$ and $S_v(-\mathbf{y})$ that is guaranteed to be positive. As such, we decide whether to sum up positive or negative multiples of \mathbf{v} based on which choice moves the sum away from $\mathbf{0}$.

In particular, if $\langle \mathbf{y}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2$, then we let $f_v(\mathbf{y}) = S_v(\mathbf{y})/M \geq 0$ for some $M > 1$. In order for (4) to be fulfilled for $\mathbf{z} = \mathbf{y} + \mathbf{v}$, we then require that

$$g_v(\mathbf{y} - 2\mathbf{v}) = \frac{p(\mathbf{y} - \mathbf{v}) - S_v(\mathbf{y})p(\mathbf{y})}{Mp(\mathbf{y} - 2\mathbf{v})} = \frac{1 - S_v(\mathbf{y} - 2\mathbf{v})}{M}.$$

Similarly, if $\langle \mathbf{y}, \mathbf{v} \rangle < \|\mathbf{v}\|^2$, we instead let $f_v(\mathbf{y}) = (1 - S_v(-\mathbf{y}))/M$ which leads to the requirement that

$$g_v(\mathbf{y} - 2\mathbf{v}) = \frac{p(-\mathbf{y} + \mathbf{v}) + (S_v(-\mathbf{y}) - 1)p(\mathbf{y})}{Mp(-\mathbf{y} + 2\mathbf{v})} = \frac{S_v(-\mathbf{y} + 2\mathbf{v})}{M}.$$

in order for (4) to be fulfilled for $\mathbf{z} = \mathbf{y} - \mathbf{v}$.

These definitions for f_v and g_v are detailed in the theorem below, where we prove that, for a suitable M , these functions fulfill all requirements of Lemma 3. Note that if (5) is fulfilled for all relevant \mathbf{y} , then these functions satisfy the requirements of Lemma 3 with $M = 1$ and we recover the same functions considered in the previous section.

Theorem 4. *Let \mathcal{K}_v be the set of \mathbf{y} such that $|\langle \mathbf{y}, \mathbf{v} \rangle| \leq \|\mathbf{v}\|^2$ and let $M \geq 1$ be such that*

$$M \geq \max_{\mathbf{y} \in \mathcal{K}_v} 1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})}.$$

Then, the functions

$$f_v(\mathbf{y}) = \begin{cases} \frac{S_v(\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2 \\ \frac{1 - S_v(-\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle < \|\mathbf{v}\|^2 \end{cases}$$

and

$$g_v(\mathbf{y}) = \begin{cases} \frac{1 - S_v(\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2 \\ \frac{S_v(-\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle < -\|\mathbf{v}\|^2 \end{cases}$$

fulfill all the requirements of Lemma 3.

Proof. We begin by showing that (3) holds for these definitions of f_v and g_v . For \mathbf{z} such that $\langle \mathbf{z}, \mathbf{v} \rangle \geq 0$, we have $\langle \mathbf{z} - \mathbf{v}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$ and $\langle \mathbf{z} + \mathbf{v}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2$. As such, we have

$$\begin{aligned} & p(\mathbf{z} + \mathbf{v})f_v(\mathbf{z} + \mathbf{v}) + p(\mathbf{z} - \mathbf{v})g_v(\mathbf{z} - \mathbf{v}) \\ &= \frac{S_v(\mathbf{z} + \mathbf{v})p(\mathbf{z} + \mathbf{v}) + (1 - S_v(\mathbf{z} - \mathbf{v}))p(\mathbf{z} - \mathbf{v})}{M} \\ &= \frac{p(\mathbf{z} - \mathbf{v})}{M} + \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{z} + (k+1)\mathbf{v}) - p(\mathbf{z} + (k-1)\mathbf{v})}{M} \\ &= \frac{p(\mathbf{z})}{M} \end{aligned}$$

as expected. Furthermore, when $\langle \mathbf{z}, \mathbf{v} \rangle < 0$, we have $\langle \mathbf{z} - \mathbf{v}, \mathbf{v} \rangle < -\|\mathbf{v}^2\|$ and $\langle \mathbf{z} + \mathbf{v}, \mathbf{v} \rangle < \|\mathbf{v}^2\|$ and thus

$$\begin{aligned} & p(\mathbf{z} + \mathbf{v})f_v(\mathbf{z} + \mathbf{v}) + p(\mathbf{z} - \mathbf{v})g_v(\mathbf{z} - \mathbf{v}) \\ &= \frac{(1 - S_v(-\mathbf{z} - \mathbf{v}))p(\mathbf{z} + \mathbf{v}) + S_v(-\mathbf{z} + \mathbf{v})p(\mathbf{z} - \mathbf{v})}{M} \\ &= \frac{p(\mathbf{z} + \mathbf{v})}{M} + \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{z} - (k-1)\mathbf{v}) - p(\mathbf{z} - (k+1)\mathbf{v})}{M} \\ &= \frac{p(\mathbf{z})}{M} \end{aligned}$$

as expected.

Next, we show that $f_v(\mathbf{y}) \geq 0$ and $g_v(\mathbf{y}) \geq 0$ for every \mathbf{y} . This follows from showing that, when $\langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$ then $S_v(\mathbf{y}) \leq 1$ and when $\langle \mathbf{y}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2$ then $S_v(\mathbf{y}) \geq 0$. To this end, we first note that

$$\|\mathbf{y} + k\mathbf{v}\|^2 = \|\mathbf{y}\|^2 + 2k\langle \mathbf{y}, \mathbf{v} \rangle + k^2\|\mathbf{v}\|^2$$

and see that if $\langle \mathbf{y}, \mathbf{v} \rangle \geq 0$, then $\|\mathbf{y} + k\mathbf{v}\|$ is greater than $\|\mathbf{y} + (k-1)\mathbf{v}\|$ for every $k \geq 1$.

When $\langle \mathbf{y}, \mathbf{v} \rangle \geq 0$, and thus also when $\langle \mathbf{y}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2$, we have

$$S_v(\mathbf{y}) = \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} = \sum_{k=0}^{\infty} \frac{p(\mathbf{y} + 2k\mathbf{v}) - p(\mathbf{y} + (2k+1)\mathbf{v})}{p(\mathbf{y})} \geq 0$$

as every term in the second sum is non-negative. Furthermore, when $\langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$, we have

$$S_v(\mathbf{y}) = \sum_{k=0}^{\infty} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} = 1 - \sum_{k=1}^{\infty} (-1)^{k+1} \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} = 1 - S_v(\mathbf{y} + \mathbf{v})$$

and this is upper bounded by 1 as $S_v(\mathbf{y} + \mathbf{v}) \geq 0$ since $\langle \mathbf{y} + \mathbf{v}, \mathbf{v} \rangle \geq 0$.

Finally, left to show is only that $f_v(\mathbf{y}) + g_v(\mathbf{y}) \leq 1$ for every \mathbf{y} . To this end, we note that when $|\langle \mathbf{y}, \mathbf{v} \rangle| > \|\mathbf{v}\|^2$ this is trivially true as

$$f_v(\mathbf{y}) + g_v(\mathbf{y}) = \begin{cases} \frac{S_v(\mathbf{y}) + 1 - S_v(\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle > \|\mathbf{v}\|^2 \\ \frac{S_v(-\mathbf{y}) + 1 - S_v(-\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle < -\|\mathbf{v}\|^2 \end{cases} = \frac{1}{M}.$$

Meanwhile, when $|\langle \mathbf{y}, \mathbf{v} \rangle| \leq \|\mathbf{v}\|^2$, we have that

$$\begin{aligned} f_v(\mathbf{y}) + g_v(\mathbf{y}) &= \frac{2 - S_v(\mathbf{y}) - S_v(-\mathbf{y})}{M} \\ &= \frac{1}{M} \left(1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} \right) \leq 1. \end{aligned}$$

with final inequality due to the definition of M in the theorem statement. \square

3.3 Discrete Gaussian Distributions

To actually make use of the definitions of $f_v(\mathbf{y})$ and $g_v(\mathbf{y})$ in Theorem 4 we must be able to efficiently compute $S_v(\mathbf{y})$ with sufficiently high precision. In this section, we show that this is possible when we work with a discrete Gaussian distribution, which is the case we focus on in this work.

When considering a discrete Gaussian distribution, $p(\mathbf{y})$ is proportional to $\rho_r(\mathbf{y})$ for some Gaussian parameter r . As such, we have that

$$S_{\mathbf{v}}(\mathbf{y}) = \sum_{k \geq 0} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})}$$

and we want to efficiently compute a sufficiently good approximation of this $S_{\mathbf{v}}(\mathbf{y})$. The following lemma bounds the error incurred by only including the t first terms of the sum in the infinite sum in the expression for $S_{\mathbf{v}}(\mathbf{y})$. This shows that only accounting for a few terms of this sum provides a sufficiently good approximation of $S_{\mathbf{v}}(\mathbf{y})$. In particular, this lemma gives that the first 20 terms of the sum differ significantly less than 2^{-256} from the true value of the infinite sum, even for the worst case where $\|\mathbf{v}\| = 1$.² Since the lemma follows from simple calculations, we omit the proof from here and instead include it in Appendix A.

Lemma 5. *The function $S_{\mathbf{v}}(\mathbf{y})$ only depends on $\langle \mathbf{y}, \mathbf{v} \rangle$ and $\|\mathbf{v}\|^2$. Furthermore, for arbitrary $t \geq 1$, fixed \mathbf{v} and with $\langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$ then*

$$\left| S_{\mathbf{v}}(\mathbf{y}) - \sum_{k=0}^t (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \right| \leq \frac{\rho_r(t\mathbf{v})}{\rho_r(\mathbf{v}) - \rho_r(2\mathbf{v})}$$

To actually compute $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$, we must also have an efficient way to compute a value for M . The value of M must be no smaller than the infinite sum in Theorem 4 and we determine a simple expression that satisfy this in Lemma 6 below. This bound is very tight for relevant parameters, and there is therefore no significant downside to using this expression for M . However, note that the main purpose of this is to simplify the expression for M , and we therefore do not include the proof here, instead including it in Appendix A where it is proven using the Poisson summation formula.

Lemma 6. *Let $\mathcal{K}_{\mathbf{v}}$ be the set of \mathbf{y} such that $|\langle \mathbf{y}, \mathbf{v} \rangle| \leq \|\mathbf{v}\|^2$ and let $\alpha = r/\|\mathbf{v}\|$. Then*

$$\max_{\mathbf{y} \in \mathcal{K}_{\mathbf{v}}} 1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \leq 1 + \frac{2\alpha\sqrt{2\pi}\rho(\pi\alpha)}{\rho_{\alpha}(1) \cdot (1 - \rho(2\pi\alpha))} = M_{\alpha}$$

and M_{α} is strictly decreasing with α .

This finally gives the expression for the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ that we use for the rejection sampling via $R_{\mathbf{v}}$ for our signature scheme. These definitions for $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ are given in Corollary 7 below, which follows from combining Theorem 4 with Lemma 6.

Corollary 7. *Let \mathbf{w} be a vector of maximal norm in some set \mathcal{V} and let $M = M_{r/\|\mathbf{w}\|}$ where*

$$M_{\alpha} = 1 + \frac{2\alpha\sqrt{2\pi}\rho(\pi\alpha)}{\rho_{\alpha}(1) \cdot (1 - \rho(2\pi\alpha))}$$

Then, the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ defined via

$$f_{\mathbf{v}}(\mathbf{y}) = \begin{cases} \frac{S_{\mathbf{v}}(\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle \geq \|\mathbf{v}\|^2 \\ \frac{1 - S_{\mathbf{v}}(-\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle < \|\mathbf{v}\|^2 \end{cases}$$

²The special case $\mathbf{v} = \mathbf{0}$ is never relevant for our rejection sampling.

and

$$g_{\mathbf{v}}(\mathbf{y}) = \begin{cases} \frac{1 - S_{\mathbf{v}}(\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2 \\ \frac{S_{\mathbf{v}}(-\mathbf{y})}{M} & \text{If } \langle \mathbf{y}, \mathbf{v} \rangle < -\|\mathbf{v}\|^2 \end{cases}$$

fulfills all the requirements of Lemma 3 for every $\mathbf{v} \in \mathcal{V}$.

3.4 Benefit of Our Method

Using the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$, as given by Corollary 7 to rejection sample with $R_{\mathbf{v}}$ leads to a repetition rate of M . We want to compare this M with the repetition rate of previous works for lattice-based signatures relying on Fiat-Shamir with aborts, in particular those based on rejection sampling from bimodal Gaussian distributions.

For this comparison, we define the set \mathcal{V} to contain all possible vectors \mathbf{v} that we want the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ given by Corollary 7 to handle. The rejection rate of both our approach and previous ones is dependent on $\alpha = r/\|\mathbf{w}\|$, where \mathbf{w} is the element of \mathcal{V} of maximal norm. In this setting, bimodal Gaussian schemes such as BLISS [16], have a repetition rate of $\exp(1/(2\alpha^2)) = 1/\rho_{\alpha}(1)$, and the expected number of rejections is thus $1/\rho_{\alpha}(1) - 1$.

In order to illustrate the benefit of our new method, we provide a plot in Figure 2 with the logarithm of the expected number of rejections for our new method compared to the same value for BLISS. For the parametrizations in BLISS, a value of α between 0.5 and 1 is used. As can be seen from the plot in Figure 2, our new method has a noticeable advantage in the rejection rate for the values of α used for BLISS. From this it also follows that, with the same rejection rate, our new method allows for a parametrization with somewhat smaller α , leading to somewhat smaller signatures.

Note that the advantage in terms of rejection rate of our method increase as α increase. For somewhat large α the ordinary bimodal Gaussian rejection sampling method still has to reject relatively frequently, whereas the rejection rate for our new method quickly approaches 0 with larger α . By itself, this does not provide a significant advantage as, from a performance perspective, there is not that big of a difference between rejecting one out of ten signatures compared only rejecting signatures very rarely.

The low rejection rate for larger α can, however, be used to ensure that even if the sampling is repeated multiple times, there is still a high probability that all repetitions succeed. This allows our new method to gain a significant advantage over previous methods, as it enables an iterative construction of \mathbf{z} . As described in Section 1.2.2, this is accomplished by performing rejection sampling for each coefficient of \mathbf{c} sequentially. This iterative process starts with $\mathbf{z}_0 = \mathbf{y}$ and then constructs $\mathbf{z}_i = \mathbf{z}_{i-1} \pm \mathbf{S}\mathbf{c}_i$, where \mathbf{c}_i has at most one non-zero coefficient for $i \in \{1, \dots, \kappa\}$ with $\mathbf{c} = \sum_i \mathbf{c}_i$ and where the final output of the rejection sampling is $\mathbf{z} = \mathbf{z}_{\kappa}$.

Similar to without the iterative approach, the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ used for the iterative rejection sampling are defined via Corollary 7. Compared to the non-iterative approach, the set \mathcal{V} that the functions has to handle vectors \mathbf{v} from is however different. With the iterative rejection sampling \mathcal{V} consists of all possible columns of \mathbf{S} , whereas without it, the set consists of all possible values of $\mathbf{S}\mathbf{c}$. As the longest column of \mathbf{S} is significantly shorter than the longest vector of the form $\mathbf{S}\mathbf{c}$, this allows the functions $f_{\mathbf{v}}$ and $g_{\mathbf{v}}$ to be defined with $M = M_{\alpha}$ for a significantly larger α .

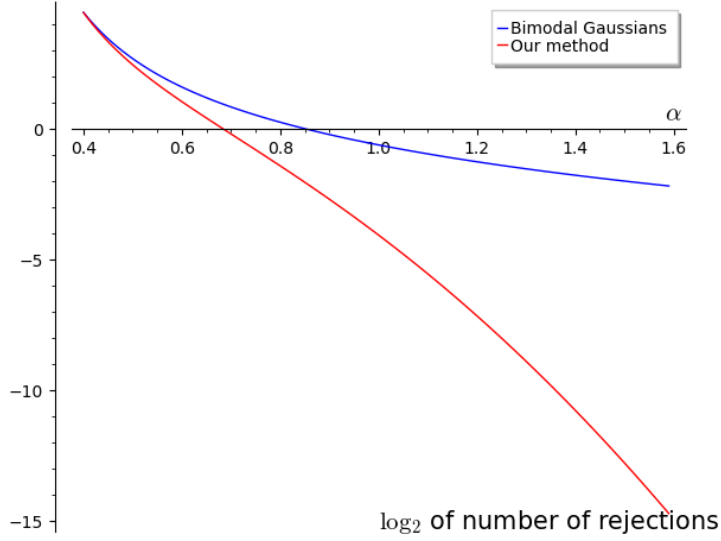


Figure 2: The expected number of rejections for each signature with normal bimodal Gaussian rejection sampling and with our new method. Note that it is the base 2 logarithm of the expected number of rejections that is plotted, where the number of rejections is the number of repetitions minus one.

Another benefit of our new method for rejection sampling is that the rejection probability decrease exponentially with increasing α . As such, it is easy to select parameters that target a negligible rejection probability and therefore allow the rejection condition to be safely ignored. To quantify this, we note that the rejection probability is given by

$$\begin{aligned}
 1 - \frac{1}{M} &= \frac{2\alpha\sqrt{2\pi}\rho(\pi\alpha)}{\rho_\alpha(1) \cdot (1 - \rho(2\pi\alpha)) + 2\alpha\sqrt{2\pi}\rho(\pi\alpha)} \\
 &\leq \frac{2\alpha\sqrt{2\pi}\rho(\pi\alpha)}{\rho_\alpha(1) \cdot (1 - \rho(2\pi\alpha))} < 4\alpha\sqrt{2\pi}\rho(\pi\alpha) \quad (6)
 \end{aligned}$$

with the final inequality holding when $\alpha \geq 1$. In particular, this bound gives that $\alpha \geq 4$ leads to a rejection probability that is smaller than 2^{-108} .

A signature scheme parametrized with $\alpha \geq 4$ could thus quite likely be used to sign on the order of 2^{100} messages without ever triggering the rejection condition. Furthermore, only a little information about the secret is leaked if a single signature that should have been rejected is emitted by the signature scheme. As such, a scheme parametrized with $\alpha \geq 4$ can be implemented while ignoring the rejection condition, without this significantly impacting the security of the scheme.

4 Signature Scheme

The high-level construction of our signature scheme is the same as that of BLISS. However, our scheme differ from BLISS by using our new method for rejection sampling in an iterative manner to construct the signatures. Additionally, our scheme constructs the public key while relying on the

assumed hardness of the NTWE problem whereas BLISS relied upon the NTRU problem.

The iterative construction for our signature scheme results in κ separate chances for rejection, once for each non-zero coefficient of c . The same iterative construction is possible also with previous rejection sampling methods. However, with previous rejection sampling methods, using the iterative construction of \mathbf{z} would require parameters to be increased to ensure that there is a sufficiently high probability that all κ iterative steps succeed. With our new rejection sampling, the iterative construction instead allows us to use smaller parameters, as the low rejection probability we have for larger α more than makes up for having to repeat the sampling κ times.

The signature scheme makes use of the functions f_v and g_v from Corollary 7 to implement the rejection sampling procedure $R_v(\mathbf{y})$ described in Figure 1. For the definition of these functions, we let \mathcal{V} be the set of all possible secret vectors \mathbf{s} , ensuring that the scheme has the same rejection probability for all possible secrets. We furthermore ensure that only keys with $\|\mathbf{s}\| \leq B_k$ are generated, and thus $\alpha = r/B_k$ is used to determine the value of $M = M_\alpha$.

To illustrate the high level structure of the scheme, we begin by presenting an uncompressed version of our scheme in the next section. Significantly more compact signatures are possible through utilizing compression techniques similar to ones that have been used in many previously proposed schemes [3, 11, 14, 25]. As usage of this compression technique somewhat complicates the description of the scheme, we present the compressed version separately in Section 4.2.

4.1 Uncompressed Signature Scheme

An uncompressed version of our scheme is presented in Figure 3. For the key-generation \mathbf{A}_0 and \mathbf{b} are generated as an NTWE instance, and, assuming the hardness of the decision NTWE problem, the pair $(\mathbf{A}_0, \mathbf{b})$ are computationally indistinguishable from uniformly random. Then, the verification key \mathbf{A} and secret key \mathbf{s} are constructed such that $\mathbf{A}\mathbf{s} \equiv q\mathbf{j} \pmod{2q}$, with \mathbf{A} constructed from \mathbf{A}_0 and \mathbf{b} whereas \mathbf{s} is constructed from the secret of the NTWE instance.

For signing a message, a relatively short vector \mathbf{y} is sampled and a commitment $\mathbf{w} = \mathbf{A}\mathbf{y} \pmod{2q}$ is computed. From this, and the message to be signed, a challenge c is derived using a random oracle $\mathcal{H} : \mathcal{R}_q^m \times \mathcal{M} \rightarrow \mathcal{C}$ where \mathcal{M} is the message space and \mathcal{C} is some challenge space. In practice this random oracle is instantiated from a secure hash function combined with a way to sample from the challenge space, in the same way as for HAETAE.

A signature consist of this challenge, and a short vector \mathbf{z} such that

$$\mathbf{A}\mathbf{z} \equiv \mathbf{w} + qc\mathbf{j} \pmod{2q}. \quad (7)$$

To construct this \mathbf{z} , the rejection sampling procedure R_v is used with the functions f_v and g_v from Corollary 7. This rejection sampling is performed for $\mathbf{v} = \mathbf{s}x^j$, for each monomial x^j in c . The end result is $\mathbf{z} = \mathbf{y} + \mathbf{s}c'$ for some ternary $c' \equiv c \pmod{2}$ which ensures that (7) holds.

Verification verifies that \mathbf{z} is sufficiently short, and implicitly verifies that (7) holds by checking that

$$\mathcal{H}(\mathbf{A}\mathbf{z} - qc\mathbf{j} \pmod{2q}, \mu) = \mathcal{H}(\mathbf{w}, \mu) = c.$$

KeyGen	Sign($\mathbf{A}, \mathbf{s}, \mu$)
1: $A_0 \leftarrow \mathcal{U}(\mathcal{R}_q)^{m \times \ell}$ 2: $\mathbf{s}_0 \leftarrow \mathcal{D}_{\mathcal{R}, \sigma}^\ell$ 3: $\mathbf{e} \leftarrow \mathcal{D}_{\mathcal{R}, \sigma}^m$ 4: $f_0 \leftarrow \mathcal{D}_{\mathcal{R}, \sigma}$ 5: $f = 2f_0 + 1$ 6: if f is not invertible in \mathcal{R}_q 7: Goto line 4 8: $\mathbf{b} = (\mathbf{A}_0 \mathbf{s} + \mathbf{e}) / f \bmod q$ 9: $\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}_m]$ 10: $\mathbf{s} = [f, \mathbf{s}_0, \mathbf{e}]$ 11: if $\ \mathbf{s}\ \geq B_k$ 12: Goto line 2 13: return $vk = \mathbf{A}, sk = \mathbf{s}$	1: $\mathbf{y} \leftarrow \mathcal{D}_{\mathcal{R}, r}^{\ell+m+1}$ 2: $\mathbf{w} = \mathbf{A}\mathbf{y} \bmod 2q$ 3: $c = \mathcal{H}(\mathbf{w}, \mu)$ 4: $\mathbf{z} = \text{RejectSample}(\mathbf{y}, c)$ 5: if $\mathbf{z} = \perp$ or $\ \mathbf{z}\ > B_s$ 6: Goto line 1 7: return (\mathbf{z}, c)
Verify($(\mathbf{z}, c), \mathbf{A}, \mu$)	RejectSample(\mathbf{z}_0, c)
1: Reject if $\ \mathbf{z}\ > B_s$ 2: Accept if $\mathcal{H}(\mathbf{A}\mathbf{z} - qc\mathbf{j} \bmod 2q, \mu) = c$	1: $k = 0$ 2: for Each monomial x^j in c 3: $k = k + 1$ 4: $\mathbf{z}_k = R_{sx^j}(\mathbf{z}_{k-1})$ 5: if $\mathbf{z}_k = \perp$ 6: return \perp 7: $r \leftarrow \mathcal{U}([0, 1])$ 8: if $r > M^{k-\kappa}$ 9: return \perp 10: return \mathbf{z}_k

Figure 3: Key-generation, signing and verification for an uncompressed version of our new scheme with an auxiliary function for rejection sampling.

4.1.1 Challenge Space

The challenge space \mathcal{C} of our signature scheme contains elements in \mathcal{R} with at most κ non-zero coefficients. The challenge space \mathcal{C} is selected such that the number of possible challenges is sufficiently large. For most parametrizations, this is accomplished by letting \mathcal{C} consist of elements with exactly κ coefficients equal to one and the remaining coefficients equal to zero.

For the parametrizations which target the highest security level, the challenge space \mathcal{C} is selected somewhat differently in order to increase its size. As with HAETAE [10, 11], for these parametrizations \mathcal{C} instead consist of exactly half of all possible degree n polynomials with only 0 and 1 coefficients. Furthermore, the challenge polynomials all contain at most $n/2$ non-zero coefficients. In this case, the challenge space thus contains 2^{n-1} different elements. For details on exactly which polynomials are included, and how to sample from this challenge space, we refer to Section 3.3 of [10].

In order to ensure that the rejection sampling does not influence the distribution of challenges c in signatures that are output, we ensure that every challenge c has the same rejection probability. Over a random \mathbf{y} , the function $R_{\mathbf{v}}(\mathbf{y})$ has the same rejection rate for every \mathbf{v} , and thus the rejection rate for a challenge c is influenced only by the number of non-zero coefficients in c .

As mentioned above, for the highest security level, the number of non-zero coefficients vary and challenges with fewer non-zero coefficients require fewer iterative steps for the rejection sampling. As each iterative step has the same rejection rate, if not handled separately, challenges with fewer non-zero coefficients would be more likely in output signatures. To ensure that every challenge c is equally likely, we therefore reject signatures with challenges that have fewer than κ non-zero coefficients with a suitable probability.

4.1.2 Bounds

For our scheme, the secret distribution $\mathcal{D}_{\mathcal{R},\sigma}$ is a discrete Gaussian distribution with parameter σ . To bound the rejection probability, we ensure that the norm of the secret vector is no larger than B_{κ} by rejecting secret keys with norm larger than this. We select B_{κ} as $\sigma\sqrt{n(m+\ell+4)}$, resulting in approximately half of the secret keys being rejected.

Limiting the norm of the secret key by B_{κ} results in secrets sampled from a narrower distribution than otherwise, and should result in the security of the scheme relying on a somewhat easier lattice problem. However, as around half of the secret keys are still accepted, the security against lattice attacks should not decrease by significantly more than 1 bit. For simplicity, we do not account for this small loss in the hardness of the underlying lattice problem for our security estimates.

As \mathbf{z} follow a discrete Gaussian distribution, there is no upper bound on $\|\mathbf{z}\|$. However, in order to ensure that signature forgery is hard, only sufficiently short \mathbf{z} can be accepted as valid signatures. As such, only signatures that satisfy $\|\mathbf{z}\| \leq B_s$ are emitted. If a \mathbf{z} with larger norm is generated during a signing attempt, the signature is rejected and the signing of the message is retried.

The bound B_s must be set to not result in too many signature rejections, while still only accepting relatively short \mathbf{z} . We therefore set $B_s =$

CSign ($\mathbf{A}, \mathbf{s}, \mu$)	HighBits (\mathbf{w})
1: $\mathbf{y} \leftarrow \mathcal{D}_{\mathcal{R}, r}^{\ell+m+1}$ 2: $\mathbf{w} = \mathbf{A}\mathbf{y} \bmod 2q$ 3: $\mathbf{w}_0 = \text{LSB}(\mathbf{w})$ 4: $c = \mathcal{H}(\text{HighBits}(\mathbf{w}), \mathbf{w}_0, \mu)$ 5: $\mathbf{z} = \text{RejectSample}(\mathbf{y}, c)$ 6: if $\mathbf{z} = \perp$ or $\ \mathbf{z}\ > B_s$ 7: Goto line 1 8: $(\mathbf{z}_1, \mathbf{z}_2) = \text{Split}(\mathbf{z})$ 9: $\tilde{\mathbf{w}} = \mathbf{w} - 2\mathbf{z}_2 \bmod 2q$ 10: $\mathbf{h} = \text{HighBits}(\mathbf{w}) - \text{HighBits}(\tilde{\mathbf{w}}) \bmod 2(q-1)$ 11: return $(\mathbf{z}_1, \mathbf{h}, c)$	1: return $\lfloor \mathbf{w}/\tau \rfloor \cdot \tau \bmod 2(q-1)$ LowBits (\mathbf{w})
	1: return $\mathbf{w} - \text{HighBits}(\mathbf{w}) \bmod^{\pm} 2q$
	CVerify ($(\mathbf{z}_1, \mathbf{h}, c), \mathbf{A}, \mu$)
	1: $\tilde{\mathbf{w}}' = \mathbf{A}_1 \mathbf{z}_1 - qc\mathbf{j} \bmod 2q$ 2: $\mathbf{w}' = \text{HighBits}(\tilde{\mathbf{w}}') + \mathbf{h} \bmod 2(q-1)$ 3: Let z_0 be the first element of \mathbf{z}_1 4: $\mathbf{w}_0 = \text{LSB}(z_0 - c)\mathbf{j}$ 5: $c' = \mathcal{H}(\mathbf{w}', \mathbf{w}_0, \mu)$ 6: $\mathbf{z}'_2 = (\mathbf{w}' - \tilde{\mathbf{w}}' + \mathbf{w}_0)/2 \bmod^{\pm} q$ 7: $\mathbf{z}' = [\mathbf{z}'_1, \mathbf{z}'_2]^T$ 8: Accept if $c = c'$ and $\ \mathbf{z}'\ \leq B_v$

Figure 4: Compressed variant of signing algorithm and corresponding verification algorithm for our new scheme.

$\lceil 1.02r\sqrt{n(\ell+m+1)} \rceil$, which from testing results in rejecting around 20% of constructed signatures. For the parametrizations targeting higher security levels, this rejection probability is somewhat lower. Note that this rejection of \mathbf{z} with too large norm only applies to the final signature, and not to the intermediate \mathbf{z}_i during the iterative rejection sampling.

4.2 More Compact Signatures

Compressing the signatures and encoding them in a compact way leads to significantly smaller signatures. We present the compressed variant of our signature scheme in Figure 4. The compression and encoding we use is very similar to that used for HAETAE, which in turn is similar to what many other lattice-based signature schemes use. In Section 4.2.1 we describe the signature compression used for this scheme, and in Section 4.2.2 we describe how the resulting signatures can be encoded in a compact way.

The technique we use for signature compression somewhat alters the security assumptions for the scheme and results in the security of the scheme relying on a somewhat easier MSIS problem. Meanwhile, the choice of signature encoding has no security implications and besides decreasing signature sizes, only impacts details in how the scheme is implemented.

An additional size optimization that is almost always used for lattice-based cryptosystems is to pseudorandomly derive uniformly random values from a short seed. In our case, this allows the verification key to be represented by \mathbf{b} and a short seed from which \mathbf{A}_0 is derived. For the concrete verification key sizes we present, we therefore account for the size of \mathbf{A}_0 only as a 32-byte seed.

4.2.1 Signature Compression

For our scheme, we make use of $\mathbf{A} = [q\mathbf{j} - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}_m]$. This is a similar situation as the one in for instance HAETAETAE [10, 11], with the last m coordinates of \mathbf{z} only having a small impact on the value of \mathbf{Az} . This leads to the idea of compressing the signature by not including these last m coordinates of \mathbf{z} in the signature. To this end, we define the function $\text{Split}(\mathbf{z})$ which splits $\mathbf{z} \in \mathcal{R}^{m+\ell+1}$ into $\mathbf{z}_1 \in \mathcal{R}^{\ell+1}$ and $\mathbf{z}_2 \in \mathcal{R}^m$, with $\mathbf{z} = [\mathbf{z}_1^T, \mathbf{z}_2^T]^T$.

With \mathbf{z}_2 not included in the signature, the verifier can no longer recompute \mathbf{w} . However, as these last coordinates only have a small impact, the verifier is still able to recompute a good approximation of the committed vector \mathbf{w} . To ensure that a signature without \mathbf{z}_2 actually is useful, the signer therefore does not commit to the full vector $\mathbf{w} = \mathbf{Ay} \bmod 2q$. Instead, the signer commits only to the parts of \mathbf{w} which are not heavily influenced by \mathbf{z}_2 .

To ensure that the committed value can be correctly recomputed by the verifier, the signer also includes a small hint \mathbf{h} with the signature. This hint describes the behaviour that \mathbf{z}_2 should have on the parts of \mathbf{w} that the signer committed to, enabling the verifier to correctly recompute this part of \mathbf{w} . This is quite similar to the approach in Dilithium [25], and even more similar to the approach of a preliminary version of Dilithium [17] which made use of discrete Gaussian distributions.³

As with Dilithium, we want to commit to high order bits of \mathbf{w} . To this end, we consider some integer τ that divides $2(q-1)$ and round each coefficient of \mathbf{w} to the nearest multiple of τ . Even without knowledge of \mathbf{z}_2 , the verifier can compute a good approximation $\tilde{\mathbf{w}}'$ of \mathbf{w} . Even though $\tilde{\mathbf{w}}'$ is a good approximation of \mathbf{w} , rounding the coefficients of $\tilde{\mathbf{w}}'$ to the nearest multiple of τ is not guaranteed to give the same result as doing the same with \mathbf{w} . This is handled by the hint vector that consist of the number of multiples of τ that differ between the rounded values of corresponding coefficients in \mathbf{w} and $\tilde{\mathbf{w}}'$.

In a similar manner as with Dilithium, we note that the difference modulo $2q$ between 0 and $2(q-1)$ is only 2. To somewhat decrease the size of the hint vectors at a minimal cost, we therefore consider 0 and $2(q-1)$ to be the same multiple of τ . This corresponds to first rounding coefficients to the nearest multiple of τ and then taking the result modulo $2(q-1)$.

For the security proof of the scheme, we must also commit to the least significant bits of \mathbf{w} . This is without extra cost, as the least significant bit of $\mathbf{Az} \bmod 2q$ is unaffected by \mathbf{z}_2 since the last m columns of \mathbf{A} consist of $2\mathbf{I}_m$.

4.2.2 Signature Encoding

To encode our signatures, we follow [19] and use the entropy coding called range asymmetric numeral systems (rANS) by Duda [18]. We apply this coding method to the hint \mathbf{h} and to \mathbf{z}_1 . The size of the encoding of \mathbf{h} and \mathbf{z}_1 is close to the entropy of the distributions that \mathbf{h} and \mathbf{z}_1 follow. To encode \mathbf{c} , we simply include an n -bit string with non-zero bits in the string corresponding to non-zero coefficients of \mathbf{c} .

For the coding of \mathbf{z}_1 , we follow the same approach as [19] and split each coefficient of \mathbf{z}_1 into two parts. The tails, consisting of the $k = 2^{\lceil \log(\tau) \rceil}$

³This preliminary version is available via <https://eprint.iacr.org/archive/2017/633/20170627:201152>

least significant bits of the coefficients, are included as is without use of rANS encoding. Encoding the tail would not significantly benefit the scheme as the coefficient bits included in the tail are close to distributed uniformly at random. Meanwhile, the head part of \mathbf{z}_1 , consisting of the coefficient bits not included in the tail, is encoded using rANS. Furthermore, the entire hint vector \mathbf{h} is encoded using rANS.

The encoding of \mathbf{h} and \mathbf{z}_1 does not result in a fixed output size. The size of the output is, however, quite close to the entropy of the distributions that these parts of the signature follow, and the variance of the size is not that large.

To encode the signatures, we determine a maximal acceptable size of the encoding of \mathbf{h} and the head part of \mathbf{z}_1 . With rANS, the combination of these are encoded into a single integer, and assuming this integer is not too large, it is padded to have exactly the maximal acceptable size. This leads to a constant size for the signatures, but also results in an additional rejection condition for the signature scheme, with the scheme retrying with a different \mathbf{y} if the compressed signature is too large.

To determine the maximal size of signatures, the entropy of the distribution of \mathbf{h} and \mathbf{z}_1 must be estimated. The entropy of \mathbf{z}_1 is determined by using that the entropy of $\mathcal{D}_{z,r}$ is close to $\log(2\pi e r^2)/2$ when r is larger than the smoothing parameter for the integers. To estimate the entropy of \mathbf{h} , we assume that it follows a discrete Gaussian distribution with parameter $2r/\tau$, which is a good approximation when r is sufficiently large in relation to τ . Total entropy is thus estimated as

$$\frac{n(\ell + 1) \log(2\pi e r^2) + nm \log(8\pi e (r/\tau)^2)}{2} \quad (8)$$

bits and we round up this value to an integer number of bytes.

We have not analyzed in detail how likely it is that signatures are larger than our selected fixed maximal size. However, for all parameters we propose we have verified that, at least a significant majority of the times, the signature can be encoded into the maximal accepted size.

For more details regarding how the signatures are encoded, and how rANS works, see Appendix B

4.2.3 Correctness

For the compressed variant of our signature scheme in Figure 4, it is not directly obvious that correctly generated signatures always pass verification. That this is the case follows from the lemma below. This lemma and its proof are very similar to the analogous statements for the HAETAE signature scheme [10, 11], and we therefore include the proof separately in Appendix A.

Lemma 8. *Let $(\mathbf{A}, \mathbf{s}) \leftarrow \text{KeyGen}()$ and let $B_v \geq B_s + \sqrt{nm}(\tau/4 + 1)$. Then, for every message $\mu \in \mathcal{M}$ the verification*

$$C\text{Verify}(C\text{Sign}(\mathbf{A}, \mathbf{s}, \mu), \mathbf{A}, \mu)$$

always succeeds.

4.3 Variant Without Rejection

Our signature scheme can be instantiated in such a way that the rejection conditions during signing can be safely ignored. The two different reasons for signatures being rejected are that the RejectSample procedure

outputs $\mathbf{z} = \perp$ or that the produced vector $\mathbf{z} \neq \perp$ has too large norm. With sufficiently large r and B_s , the probability to trigger these rejection conditions is negligible.

The probability that `RejectSample` outputs $\mathbf{z} \neq \perp$ is $M^{-\kappa}$, and M can be bounded by using (6). In particular, this bound shows that if $r \geq 4B_k$, then $1/M \geq 1 - 2^{-108}$. As $\kappa \leq 128 = 2^7$ for all parametrizations that we consider, with $r \geq 4B_k$ the probability that `RejectSample` outputs \perp is upper bounded by 2^{-100} .

Meanwhile, to ensure that the probability of $\|\mathbf{z}\| > B_s$ is sufficiently small, we set $B_s = \lceil Cr\sqrt{n(\ell + m + 1)} \rceil$ for a sufficiently large constant C .

The probability that $\|\mathbf{z}\| > B_s$ is then bounded by $(Ce^{(1-C^2)/2})^{n(\ell+m+1)}$ via Lemma 2. This leads to us requiring $C \in [1.2, 1.3]$ for our different parametrizations to ensure that the probability of $\mathbf{z} > B_s$ is at most 2^{-128} .

An additional reason for rejections when using rANS encoding of signatures is due to requiring that the signatures can be padded to a fixed size. Because of this, if a produced signature can not be encoded into at most this fixed size, it has to be rejected. This rejection condition can easily be fixed by allowing variable size signatures, and this also results in most signatures being shorter than our determined fixed size.

4.4 Theoretical Security

To prove the security of our signature scheme, we take the same approach as HAETAE [10, 11] and use Theorem 1. We thus consider a canonical identification (CID) scheme from which our signature scheme can be constructed by using the Fiat-Shamir transform. The UF-CMA and sUF-CMA security of our signature scheme then follows from the UF-NMA security of the scheme and properties of the CID scheme.

The UF-NMA security of our scheme is proven in Lemma 10 and in Lemma 11 we prove that the scheme is paHVZK. Combining these lemmas with Theorem 1 proves the UF-CMA security of our signature scheme. Additionally, Lemma 12 proves the CUR property for our signature scheme, and thus allows proving the sUF-CMA security of the scheme via Theorem 1.

The provable security of our scheme is detailed in the following theorem, which also relies upon the commitment min-entropy of the scheme. We briefly discuss the min-entropy of our scheme in Section 4.4.2.

Theorem 9. *Let γ be the commitment min-entropy of our signature scheme and β the probability that it restarts during signing. Furthermore, let \mathcal{A} be a UF-CMA attacker against our signature scheme that makes q_S queries to the signing oracle and q_H queries to the random oracle \mathcal{H} . Then there are adversaries \mathcal{B} and \mathcal{B}' with essentially the same running time as \mathcal{A} such that*

$$\begin{aligned} \text{Adv}^{UF-CMA}(\mathcal{A}) &\leq \text{Adv}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}^{NTWE}(\mathcal{B}) + \text{Adv}_{\mathcal{H},q,m,\ell,B_v}^{BimodalSelfTargetMSIS}(\mathcal{B}') \\ &\quad + \frac{2q_S 2^{-\gamma/2}}{1-\beta} \sqrt{q_H + 1 + \frac{q_S}{1-\beta}} + 2^{-\gamma/2+1}(q_H + 1) \sqrt{\frac{q_S}{1-\beta}}. \end{aligned}$$

Additionally, let \mathcal{A}' be an adversary against the sUF-CMA security of our signature scheme that makes q'_S queries to the signing oracle and q'_H queries to the random oracle \mathcal{H} . Then there are adversaries $\mathcal{F}, \mathcal{F}', \mathcal{F}''$

with essentially the same running time as \mathcal{A} such that

$$\begin{aligned} \text{Adv}^{s\text{UF-CMA}}(\mathcal{A}') &\leq 2\text{Adv}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}^{\text{NTWE}}(\mathcal{F}) + \text{Adv}_{\mathcal{H},q,m,\ell,B_v}^{\text{BimodalSelfTargetMSIS}}(\mathcal{F}') \\ &\quad + \text{Adv}_{q,m,\ell,2B_v}^{\text{MSIS}}(\mathcal{F}'') + \frac{2q'_S 2^{-\gamma/2}}{1-\beta} \sqrt{q'_H + 1 + \frac{q'_S}{1-\beta}} \\ &\quad + 2^{-\gamma/2+1}(q'_H + 1) \sqrt{\frac{q'_S}{1-\beta}}. \end{aligned}$$

4.4.1 UF-NMA Security

The UF-NMA security of the uncompressed version of our scheme is almost directly given by the assumed hardnesses of the decision NTWE problem and the BimodalSelfTargetMSIS problem. For the compressed version of our scheme, the UF-NMA security similarly relies on the assumed hardness of these problems, but not quite as obviously. For both versions of our scheme, we argue for their UF-NMA security by the following lemma.

Lemma 10. *Let \mathcal{A} be an adversary against the UF-NMA security of our signature scheme. Then, there exist two adversaries \mathcal{B} and \mathcal{B}' , each using \mathcal{A} once and performing negligible additional work, such that*

$$\text{Adv}^{\text{UF-NMA}}(\mathcal{A}) \leq \text{Adv}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}^{\text{NTWE}}(\mathcal{B}) + \text{Adv}_{\mathcal{H},q,m,\ell,B_v}^{\text{BimodalSelfTargetMSIS}}(\mathcal{B}').$$

Proof. For the proof, we consider a variant U of our scheme for which the \mathbf{A}_0 and \mathbf{b} in the public key are uniformly random instead of given by an NTWE instance. If \mathcal{A} behaves differently against U than against our actual scheme, this can be used to solve the decision NTWE problem. The adversary \mathcal{B} is defined to exploit this potential difference in the behaviour of \mathcal{A} by using an input decision NTWE instance to construct the verification key for \mathcal{A} . If a forgery is created, it determines the input to be an NTWE instances and otherwise it determines it to be uniformly random.

The algorithm \mathcal{B}' instead relies on whatever advantage \mathcal{A} has against the variant U of our scheme. To achieve this, the input to the target BimodalSelfTargetMSIS problem is used to construct the verification key that is provided to \mathcal{A} . This verification key is thus constructed from uniformly random \mathbf{A}_0 and \mathbf{b} . For the uncompressed version of our signature scheme, a forgery produced by \mathcal{A} directly corresponds to a solution for the BimodalSelfTargetMSIS problem of \mathcal{B}' , which proves the statement for the uncompressed version.

For the compressed version of our signature scheme, the random oracle that \mathcal{B}' provides when simulating \mathcal{A} is given by

$$\mathcal{H}'(\mathbf{w}', w_0, \mu) = \mathcal{H}(\mathbf{w}' + w_0, \mu)$$

where \mathcal{H} is the random oracle for the BimodalSelfTargetMSIS instance. A forgery from the signature scheme consists of a signature $(\mathbf{z}_1, \mathbf{h}, c)$ and a message μ such that

$$\mathcal{H}'(\mathbf{w}', \mathbf{w}_0, \mu) = \mathcal{H}(\mathbf{w}' + \mathbf{w}_0, \mu) = c$$

for \mathbf{w}' and \mathbf{w}_0 defined as in $\text{CVerify}((\mathbf{z}_1, \mathbf{h}, c), \mathbf{A}, \mu)$.

A valid forgery additionally corresponds to a vector $\mathbf{z}' = [\mathbf{z}'_1, \mathbf{z}'_2]^T$ such that $\|\mathbf{z}'\| \leq B_v$, where $\mathbf{z}'_2 = (\mathbf{w}' + \mathbf{w}_0 - \tilde{\mathbf{w}}')/2$. Given such a forgery,

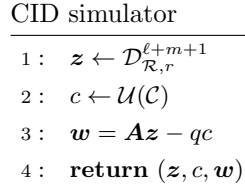
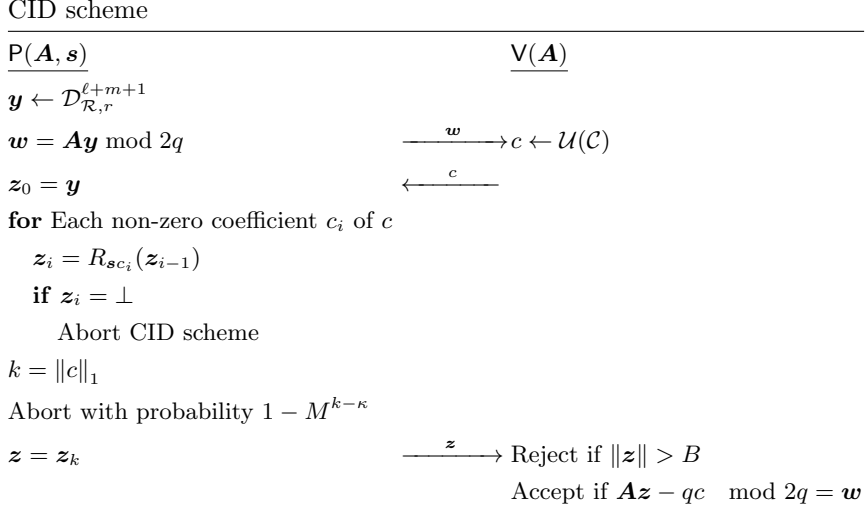


Figure 5: Canonical identification scheme and simulator for the scheme.

we claim that (\mathbf{z}', c, μ) solves the target BimodalSelfTargetMSIS instance. This is the case as

$$\mathbf{Az}' = \mathbf{A}_1 \mathbf{z}_1 + 2\mathbf{z}'_2 = \mathbf{w}' + \mathbf{w}_0 + qc\mathbf{j}$$

and thus

$$\mathcal{H}(\mathbf{Az}' - qc\mathbf{j}, \mu) = \mathcal{H}(\mathbf{w}' + \mathbf{w}_0, \mu) = c$$

as claimed. Furthermore, valid signatures satisfy that $\|\mathbf{z}'\|' \leq B_v$ and therefore the norm bound for the BimodalSelfTargetMSIS instance is satisfied. \square

4.4.2 CID Properties

The CID scheme corresponding to the uncompressed version of our scheme is illustrated in Figure 5, together with a simulator for the scheme. For simplicity, we do not provide an explicit definition of the CID scheme corresponding to the compressed variant of our signature scheme, but we still prove properties about both variants. We do this by detailing how the proofs can be adapted from the uncompressed setting to the compressed setting.

Zero knowledge First, in the following lemma we show that the CID scheme is paHVZK. For the scheme to be paHVZK, it must be possible to simulate the output from the scheme conditioned on $\mathbf{z} \neq \perp$ without

knowledge of the secret key. To simulate the CID scheme, Lemma 3 is used to show that the output of $R_v(\mathbf{y})$ is independent of \mathbf{v} , and thus the distribution of $\mathbf{z} \neq \perp$ is independent of the secret key.

Lemma 11. *The CID schemes that corresponds to the compressed and uncompressed variants of our signature scheme are paHVZK. Furthermore, the prover produces a $\mathbf{z} \neq \perp$ with probability $M^{-\kappa}$.*

Proof. The CID scheme for the uncompressed variant of our signature scheme is given in figure 5 together with a simulator for the CID scheme. Lemma 3 combined with Corollary 7 give that, conditioned on not being \perp , \mathbf{z}_i is distributed as if from $\mathcal{D}_{\kappa, r}^{\ell+m+1}$ for every $i \in \{1, \dots, k\}$. As such, conditioned on the prover producing an output, the distribution of \mathbf{z} is the same in the simulator and the CID scheme.

As $R_v(\mathbf{y})$ has a probability $1/M$ of producing a valid output, the probability that it successfully produces an output all k times it is executed is M^{-k} . After z_k has been produced, the scheme aborts with probability $1 - M^{k-\kappa}$, ensuring that the probability of an output $\mathbf{z} \neq \perp$ is $M^{-\kappa}$ for every possible challenge c . Therefore, the rejection sampling does not impact the distribution of challenges in transcripts where $\mathbf{z} \neq \perp$.

Conditioned on $\mathbf{z} \neq \perp$, for both the simulator and the CID scheme, the distribution of c is as if sampled from $\mathcal{U}(\mathcal{C})$. This also implies that $\mathbf{w} = \mathbf{A}\mathbf{z} - qc\mathbf{j}$ is distributed identically in the simulator and the CID scheme. As such, the distribution of $\mathbf{z}, c, \mathbf{w}$ from the simulator is the same as the one from the CID scheme conditioned on $\mathbf{z} \neq \perp$.

In the CID scheme corresponding to the compressed signature scheme, the verifier would first send $\mathbf{w}_h = \text{HighBits}(\mathbf{w})$ and $\mathbf{w}_0 = \text{LSB}(\mathbf{w})$. The challenge c would still be the same, but the final response of the verifier would only contain parts of \mathbf{z} together with a hint derived from \mathbf{z} and \mathbf{w} . As such, the transcript from a compressed variant of the CID scheme is given directly by a deterministic transform of the transcript of the uncompressed version. The same transform can be applied to the simulated transcript, and thus this version of the CID scheme is also paHVZK. \square

Commitment min-entropy Next, we note that the commitment min-entropy of the scheme is easily lower-bounded by close to n bits for the compressed scheme. This is the case as $\mathbf{w}_0 = \text{LSB}(y_0)\mathbf{j}$ is part of the commitment with y_0 sampled from a discrete Gaussian distribution with relatively large standard deviation. As such, the coefficients of the first element of \mathbf{w}_0 are statistically close to uniformly random, and \mathbf{w}_0 therefore has a min-entropy that is not significantly smaller than n .

Note, however, that this should significantly underestimate the commitment min-entropy of the scheme, as it only accounts for the min-entropy in \mathbf{w}_0 . Although we do not prove this, similarly to in HAETAE [10], we feel that it is safe to assume that the actual commitment min-entropy is significantly larger than this. We therefore ignore terms which depend on the min-entropy when we use Theorem 9. Furthermore note that the uncompressed variant of the scheme obviously has at least as high min-entropy as the compressed variant.

Computational Unique Response To prove the sUF-CMA security of the signature scheme, we must also bound the advantage an adversary may have against the CUR property of the CID scheme. Such a bound is presented in the following lemma.

Lemma 12. *If \mathcal{A} is an adversary against the CUR property of the CID scheme, then there exist adversaries \mathcal{B} and \mathcal{B}' , each using \mathcal{A} once and performing negligible additional work, such that*

$$\text{Adv}^{CUR}(\mathcal{A}) \leq \text{Adv}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}^{NTWE}(\mathcal{B}) + \text{Adv}_{q,m,\ell,2B_v}^{MSIS}(\mathcal{B}').$$

Proof. A successful adversary \mathcal{A} against the CUR property of the CID scheme produces $\mathbf{w}, c, \mathbf{z}_a, \mathbf{z}_b$ with $\mathbf{z}_a \neq \mathbf{z}_b \pmod{2q}$ such that $\mathbf{w}, c, \mathbf{z}_a$ and $\mathbf{w}, c, \mathbf{z}_b$ both are accepting transcripts for the CID scheme. Thus,

$$\mathbf{A}\mathbf{z}_a - qcj \pmod{2q} = \mathbf{A}\mathbf{z}_b - qcj \pmod{2q} = \mathbf{w}$$

and therefore $\mathbf{A}(\mathbf{z}_a - \mathbf{z}_b) = \mathbf{0} \pmod{2q}$. We can reduce this modulo q and multiply by the inverse of 2 modulo q , which gives

$$[-\mathbf{b}, \mathbf{A}_0, \mathbf{I}_m](\mathbf{z}_a - \mathbf{z}_b) = \mathbf{0} \pmod{q}$$

as $\mathbf{A} = [qcj - 2\mathbf{b}, 2\mathbf{A}_0, 2\mathbf{I}_m]$. As such, if \mathbf{b} was uniformly random, then $\mathbf{z}_a - \mathbf{z}_b$ would correspond to a solution to an MSIS problem.

In the scheme \mathbf{b} is generated as an NTWE instance and is not uniformly random. However, if \mathcal{A} works when \mathbf{b} is given by an NTWE instance and not when it is uniformly random, then \mathcal{A} provides a distinguisher for the decision NTWE problem. Thus, \mathcal{B} is constructed by using its input decision NTWE instance as verification key for \mathcal{A} whereas \mathcal{B}' constructs a verification key for \mathcal{A} by sampling both \mathbf{b} and \mathbf{A}_0 uniformly at random. With $\|\mathbf{z}_a\|$ and $\|\mathbf{z}_b\|$ both at most B_v , this proves the lemma for the uncompressed variant of our scheme.

For a compressed variant of the CID scheme, the adversary \mathcal{A} would instead produce $\mathbf{w}', c, \mathbf{z}_{a,1}, \mathbf{h}_a, \mathbf{z}_{b,1}, \mathbf{h}_b$, with $(\mathbf{z}_{a,1}, \mathbf{h}_a) \neq (\mathbf{z}_{b,1}, \mathbf{h}_b)$. Assuming that this guarantees that the reconstructed \mathbf{z}'_a and \mathbf{z}'_b differ, the same argument as for the uncompressed variant holds. As the reconstructed \mathbf{z}' is given by $[\mathbf{z}'_1, \mathbf{z}'_2]^T$, that $\mathbf{z}'_a = \mathbf{z}'_b$ implies that $\mathbf{z}_{a,1} = \mathbf{z}_{b,1}$.

Additionally, with $\mathbf{z}_{a,1} = \mathbf{z}_{b,1}$, for $\mathbf{z}_{a,2} = \mathbf{z}_{b,2}$ it is required that $\mathbf{w}'_a \equiv \mathbf{w}'_b \pmod{2(q-1)}$ and this is only possible if $\mathbf{h}_a \equiv \mathbf{h}_b \pmod{2(q-1)}$. To ensure that there is a computationally unique response it should thus be verified that \mathbf{h} is the expected representative modulo $2(q-1)$. With such a verification, it is guaranteed that two different verifying transcripts for the compressed CID scheme imply two different reconstructed \mathbf{z}'_a and \mathbf{z}'_b , both of which have norm at most B_v . In the same way as for the uncompressed variant, this directly provides a solution to the MSIS variant with size bound $2B_v$. \square

5 Concrete Parameters

In Table 1 we present some suggested parametrizations of our scheme which have a noticeable, but acceptable, rejection probability. Additionally, in Table 2 we present some parametrizations where the rejection conditions can be securely ignored, as described in Section 4.3. We also provide a comparison of the verification key and signature sizes of our scheme to that of alternative schemes in Table 3.

The public key size presented in the tables is

$$\left\lceil \frac{nm \lceil \log(q) \rceil}{8} \right\rceil + 32$$

bytes. This corresponds to representing each coefficient of \mathbf{b} as a $\log(q)$ bit integer, while \mathbf{A}_0 is represented by a 32-byte seed. The concrete signature

size we provide for our scheme is an approximation of the entropy for the distributions of \mathbf{z}_1 and \mathbf{h} plus 64 bytes. This includes 32 bytes for the representation of c and 32 additional bytes as leeway for our fixed signature size.

We have not extensively investigated the probability that constructed signatures can be encoded into the size we present in the tables. However, from testing we have determined that at least a large majority of all signatures can be encoded to the presented size for all parametrizations we consider here.

Note that we do not include BLISS in our comparison of schemes. This is due to the parametrizations of BLISS predating much of the work on estimating the concrete hardness of lattice problems, and the claimed security in the BLISS paper is significantly higher than it would be in the core-SVP metric. Due to this difference in how the security is estimated, even the highest security level proposed for BLISS has less than 120 bits of security in the core-SVP metric.

To fairly compare a BLISS type signature scheme to more modern lattice-based signature schemes would require new parametrizations of BLISS, which is out of scope for this work. However, it has been noted that such a scheme would be approximately as compact as HAETA [10, 11].

5.1 Selecting Parameters

The security estimates of our proposed parametrizations presented in Tables 1 and 2 are computed with the lattice estimator [1]. For reproducibility, in Appendix C we present the concrete parameters used with the lattice estimator to get our security estimates. The security estimates are given via the

`*.estimate.rough`

functions in order to get an approximation of the security in the core-SVP metric.

In Sections 5.1.1 and 5.1.2 we detail how the security of the scheme is related to the hardness of the standard lattice problems that we estimate with the lattice estimator. Both for the MSIS security and the NTWE security, we make the assumption that the structure present in \mathcal{R} can not be efficiently exploited, and that the structured lattice problems are as hard as unstructured counterparts.

The concrete parameters proposed in Tables 1 and 2 have been selected to provide the desired security levels while having an acceptable rejection probability. Furthermore, the parameters are selected so as to attempt to minimize the combined size of a verification key and a signature.

5.1.1 MSIS Security

A classical reduction in the ROM shows that an efficient algorithm for solving the `BimodalSelfTargetMSIS` $_{\mathcal{H},q,m,\ell,B_v}$ can be used to efficiently solve the `MSIS` $_{q,m,\ell,2B_v}$ problem. However, this reduction is not tight and as it makes use of rewinding and the forking lemma it can not easily be translated to a reduction in the quantum random oracle model.

Instead of using this reduction to argue for the security of our scheme, we heuristically assume that the `BimodalSelfTargetMSIS` $_{\mathcal{H},q,m,\ell,B_v}$ problem is as hard as the `MSIS` $_{q,m,\ell,B_v}$ problem. Motivation for this approach is that the only way to solve this `BimodalSelfTargetMSIS` problem seems

Security	120	180	260
q	12289	50177	50177
ℓ	1	3	4
m	2	2	3
σ	2.60	1.00	1.50
r	128	55	95
κ	58	58	128
τ	256	128	128
B_k	110.07	48.00	79.60
B_s	4178	2199	4386
B_v	5649	2946	5300
Rejection rate	0.460	0.525	0.617
Rejection rate per coefficient	0.0106	0.0127	0.0075
Verification Key Size (bytes)	928	1056	1568
Signature Size (bytes)	775	1184	1694
Combined (bytes)	1703	2240	3262
BKZ block size to break SIS (Strong UF)	420 (332)	629 (500)	899 (730)
Core-SVP cost	122 (96)	183 (146)	262 (213)
BKZ block size to break NTWE	414	618	881
Core-SVP cost	120	180	257

Table 1: Proposed parametrizations of our signature scheme. The reported rejection rates only accounts for the probability that $\mathbf{z} = \perp$ is produced by $\text{RejectSample}(\mathbf{y}, c)$ and not the probability that $\|\mathbf{z}\| > B_s$.

Security	120	180	260
q	50177	50177	50177
ℓ	2	3	4
m	2	3	4
σ	0.90	1.00	1.45
r	165	205	325
κ	58	80	128
τ	512	512	256
B_k	40.73	50.60	80.37
B_s	7541	10679	18769
B_v	10460	14254	20849
Rejection rate	2^{-111}	2^{-112}	2^{-111}
Verification Key Size (bytes)	1056	1568	2080
Signature Size (bytes)	1059	1475	2161
Combined (bytes)	2115	3043	4241
BKZ block size to break SIS (Strong UF)	416 (337)	670 (552)	889 (742)
Core-SVP cost	121 (98)	195 (161)	259 (216)
BKZ block size to break NTWE	422	618	874
Core-SVP cost	123	180	255

Table 2: Alternative parametrizations of our signature scheme which have a negligible rejection probability, allowing producing signatures without rejection.

to be to solve this corresponding MSIS problem. This same approach is used in many previous works, such as Dilithium [25] and HAETAE [10, 11].

5.1.2 NTWE Security

We estimate the hardness of the NTWE problem in the same way as in [20]. This estimate assumes that the NTWE problem with $\mathbf{A} \in \mathcal{R}^{m \times \ell}$ is essentially as hard as the rank $\ell + 1$ module-LWE problem.

More specifically, the $\text{NTWE}_{q,m,\ell,\mathcal{D}_{\mathcal{R},\sigma}}$ problem is estimated to be as hard to solve as the unstructured LWE problem with $\mathbf{A} \in \mathbb{Z}_q^{m' \times \ell'}$ where $m' = mn$ and $\ell' = (\ell + 1)n - 1$. Furthermore, for the unstructured LWE problem, it is assumed that all elements of the secret are sampled from $\mathcal{D}_{\mathbb{Z},\sigma}$. With this estimate, as $f = 2f_0 + 1$ is deterministically constructed from $f_0 \leftarrow \mathcal{D}_{\mathcal{R},\sigma}$, we do not account for the fact that f is larger than if directly sampled from $\mathcal{D}_{\mathcal{R},\sigma}$. The NTWE security estimate presented for our proposed parameters are the estimated core-SVP hardness of solving such a search LWE problem.

Our security estimate assumes that solving the decision NTWE problem is not significantly easier than the search version of the problem. It can be proven that the decision LWE problem is essentially as hard as the search version of the problem, but such a reduction is not available for the NTWE problem. This is similar to the situation for the NTRU problem, where it is quite common to simply assume that the decision and search versions of the problem are equally hard.

5.2 Comparison to Other Schemes

5.2.1 Compactness

The verification key and signature sizes of our scheme and that of other lattice-based signature schemes are presented in Table 3. Out of the presented signature schemes, only Falcon [27] and HAWK [8] have comparable sizes to our new system. Both HAWK and Falcon are based on the hash-and-sign paradigm using NTRU lattices, with the other schemes in the comparison based on some variant of the Fiat-Shamir transform.

As can be seen in Table 3, our scheme is significantly more compact than all other Fiat-Shamir-based schemes in our comparison. Furthermore, even when parametrized to have negligible rejection probability, our scheme is still more compact than previous Fiat-Shamir-based schemes. The smaller verification keys and signatures are the main advantage of our new signature scheme compared to other Fiat-Shamir-based schemes.

5.2.2 Secure Implementation

The more compact hash-and-sign-based schemes Falcon and HAWK have some downsides compared to the Fiat-Shamir-based schemes, and our scheme also has similar downsides. Compared to ML-DSA, Falcon has the downside of implementations requiring floating point operations, which can be hard to implement in a side-channel secure manner.

Our scheme may also be hard to implement in a side-channel secure manner, as it has previously been observed that implementations of rejection sampling from Gaussian distributions may be vulnerable to side-channel attacks [9, 26]. Furthermore, our new more advanced rejection

sampling may be even harder to protect from side-channel attacks. Although developing a secure implementation for our scheme almost certainly is harder than for schemes such as Dilithium and HAETAE, it is not directly clear how this aspect of our scheme compares to Falcon.

5.2.3 Hardness Assumptions

Compared to other lattice-based schemes, HAWK has the downside of relying on the assumed hardness of a problem that is not as well studied as other lattice problems. Our scheme has the similar downside on relying on the relatively new NTWE problem. However, as the NTWE problem can be seen as a natural combination of the NTRU and LWE problems, it can be argued to be better understood than the lattice isomorphism problem upon which HAWK is based.

Furthermore, as explained in Section 1.3.2, our scheme could also have been based on NTRU or MLWE lattices. An NTRU-based variant of our scheme would have similar sizes as the one we propose, but with less flexibility in how parameters can be selected. Meanwhile, an MLWE-based variant of our scheme would have the same flexibility in parameter choices, but with somewhat larger sizes. Directly transferring our NTWE-based parametrization to a MLWE-based scheme leads to a scheme with the same verification key sizes but with signatures that are 250–300 bytes larger for all targeted security levels. Thus, an MLWE-based variant of our scheme would still be significantly more compact than previous lattice-based Fiat-Shamir signature schemes.

5.2.4 Efficiency

Another point of comparison is how long it takes to sign a message. Due to the more advanced rejection sampling of our scheme, signing may be somewhat slower with our scheme than with previous Fiat-Shamir-based signature schemes. Additionally, our signing must repeat the rejection sampling multiple times in an iterative manner in order to actually produce a signature, increasing the runtime of signing.

We do, however, believe that each iterative step of the rejection sampling should not be that much more costly than previous approaches. Although the functions that should be computed are expressed as an infinite sum, Lemma 5 shows that only accounting for a few terms of the sum is sufficient. The same lemma also shows that computing the sum only requires knowledge of the two integers $\langle \mathbf{y}, \mathbf{v} \rangle$ and $\|\mathbf{v}\|^2$. As such, only minimal computations that handle high dimensional vectors are required to compute the sum.

As we do not have an optimized implementation of our scheme, we can not fairly compare the execution time of our scheme to that of previous schemes. However, we believe that in many applications, the smaller signatures and verification keys of our scheme more than make up for the somewhat longer signing time.

6 Conclusion

Thanks to our new rejection sampling, we are able to construct a signature scheme that is significantly more compact than previous lattice-based Fiat-Shamir signature schemes. There are, however, downsides to our new scheme and still potential for it to be further improved.

Security Level	Scheme	VK Size	Signature Size	Total
120	Falcon-512	897	666	1563
120	HAWK-512	1024	555	1579
120	HAETAE-120	992	1474	2466
120	G+G-120	1472	1677	3149
120	Dilithium-2	1312	2420	3732
120	Ours with rejection	928	775	1703
120	Ours without rejection	1056	1059	2115
180	HAETAE-180	1472	2349	3821
180	G+G-180	1952	2143	4095
180	NTRU G+G-180	2080	1769	3849
180	Dilithium-3	1952	3293	5245
180	Ours with rejection	1056	1184	2240
180	Ours without rejection	1568	1475	3043
260	Falcon-1024	1793	1280	3073
260	HAWK-1024	2440	1221	3661
260	HAETAE-260	2080	2948	5028
260	G+G-260	2336	2804	5140
260	Dilithium-5	2592	4595	7187
260	Ours with rejection	1568	1694	3262
260	Ours without rejection	2080	2161	4241

Table 3: Signature sizes in bytes for different lattice-based signature schemes.

A notable downside with our new sampling is its reliance on discrete Gaussian distributions which may be hard to implement in a side-channel secure manner. Meanwhile, uniformly random distributions over hypercubes and hyperballs can be sampled from relatively easily, and this is a large part of the reason why these distributions are used for Dilithium and HAETAE respectively. It would therefore be interesting if a variant of our new approach to rejection sampling could be applied to, and provide a significant advantage for, distributions that are easier to sample from.

Improvements of our rejection sampling may also be possible by further using the available flexibility in how the \mathbf{z} vector in signatures can be constructed. In the same way as for the G+G signature scheme, \mathbf{z} can be constructed as $\mathbf{y} + \mathbf{sc} + 2\mathbf{sk}$ for arbitrary $k \in \mathcal{R}$. We only use this flexibility to iteratively select the sign of coefficients of c , but further improvements may be possible by not imposing this limitation in how k is selected. However, selecting k more freely would complicate the analysis needed to guarantee that the sampling produces the correct output distribution.

Acknowledgments

This research has been supported in part by the Swedish Armed Forces and was conducted at KTH Center for Cyber Defense and Information Security (CDIS). The author would like to thank Johan Håstad for extensive comments and useful discussions on early versions of this manuscript. The author also thanks Martin Ekerå for helpful feedback and comments.

References

- [1] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.
- [2] Shi Bai, Austin Beard, Floyd Johnson, Sulani K. B. Vidhanalage, and Tran Ngo. Fiat-shamir signatures based on module-NTRU. In Khoa Nguyen, Guomin Yang, Fuchun Guo, and Willy Susilo, editors, *ACISP 22: 27th Australasian Conference on Information Security and Privacy*, volume 13494 of *Lecture Notes in Computer Science*, pages 289–308, Wollongong, NSW, Australia, November 28–30, 2022. Springer, Cham, Switzerland.
- [3] Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In Josh Benaloh, editor, *Topics in Cryptology – CT-RSA 2014*, volume 8366 of *Lecture Notes in Computer Science*, pages 28–47, San Francisco, CA, USA, February 25–28, 2014. Springer, Cham, Switzerland.
- [4] Shi Bai, Hansraj Jangir, Hao Lin, Tran Ngo, Weiqiang Wen, and Jinwei Zheng. Compact encryption based on module-NTRU problems. In Markku-Juhani Saarinen and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 15th International Workshop, PQCrypto 2024, Part I*, pages 371–405, Oxford, UK, June 12–14, 2024. Springer, Cham, Switzerland.
- [5] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.
- [6] Manuel Barbosa, Gilles Barthe, Christian Doczkal, Jelle Don, Serge Fehr, Benjamin Grégoire, Yu-Hsuan Huang, Andreas Hülsing, Yi Lee, and Xiaodi Wu. Fixing and mechanizing the security proof of Fiat-Shamir with aborts and Dilithium. In Helena Handschuh and Anna Lysyanskaya, editors, *Advances in Cryptology – CRYPTO 2023, Part V*, volume 14085 of *Lecture Notes in Computer Science*, pages 358–389, Santa Barbara, CA, USA, August 20–24, 2023. Springer, Cham, Switzerland.
- [7] Rouzbeh Behnia, Yilei Chen, and Daniel Masny. On removing rejection conditions in practical lattice-based signatures. In Jung Hee Cheon and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 12th International Workshop, PQCrypto 2021*, pages 380–398, Daejeon, South Korea, July 20–22, 2021. Springer, Cham, Switzerland.
- [8] Joppe W. Bos, Olivier Bronchain, Léo Ducas, Serge Fehr, Yu-Hsuan Huang, Thomas Pornin, Eamonn W. Postlethwaite, Thomas Prest, Ludo N. Pulles, and Wessel van Woerden. HAWK. Technical report, National Institute of Standards and Technology, 2023. available at <https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures>.
- [9] Leon Groot Bruinderink, Andreas Hülsing, Tanja Lange, and Yuval Yarom. Flush, gauss, and reload - A cache attack on the BLISS lattice-based signature scheme. In Benedikt Gierlichs and Axel Y. Poschmann, editors, *Cryptographic Hardware and Embedded Systems – CHES 2016*, volume 9813 of *Lecture Notes in Computer Science*, pages 323–345, Santa Barbara, CA, USA, August 17–19, 2016. Springer, Berlin, Heidelberg, Germany.

- [10] Jung Hee Cheon, Hyeongmin Choe, Julien Devevey, Tim Güneysu, Dongyeon Hong, Markus Krausz, Georg Land, Marc Möller, Damien Stehlé, and MinJune Yi. HAETAE: Shorter lattice-based fiat-shamir signatures. Cryptology ePrint Archive, Report 2023/624, 2023.
- [11] Jung Hee Cheon, Hyeongmin Choe, Julien Devevey, Tim Güneysu, Dongyeon Hong, Markus Krausz, Georg Land, Junbum Shin, Damien Stehlé, and MinJune Yi. HAETAE. Technical report, National Institute of Standards and Technology, 2023. available at <https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures>.
- [12] Rafael del Pino, Thomas Espitau, Shuichi Katsumata, Mary Maller, Fabrice Mouhartem, Thomas Prest, Mélissa Rossi, and Markku-Juhani Saarienen. Raccoon. Technical report, National Institute of Standards and Technology, 2023. available at <https://csrc.nist.gov/Projects/pqc-dig-sig/round-1-additional-signatures>.
- [13] Julien Devevey, Omar Fawzi, Alain Passelègue, and Damien Stehlé. On rejection sampling in Lyubashevsky’s signature scheme. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022, Part IV*, volume 13794 of *Lecture Notes in Computer Science*, pages 34–64, Taipei, Taiwan, December 5–9, 2022. Springer, Cham, Switzerland.
- [14] Julien Devevey, Alain Passelègue, and Damien Stehlé. G+G: A fiat-shamir lattice signature based on convolved gaussians. In Jian Guo and Ron Steinfeld, editors, *Advances in Cryptology – ASIACRYPT 2023, Part VII*, volume 14444 of *Lecture Notes in Computer Science*, pages 37–64, Guangzhou, China, December 4–8, 2023. Springer, Singapore, Singapore.
- [15] Léo Ducas. Accelerating bliss: the geometry of ternary polynomials. Cryptology ePrint Archive, Report 2014/874, 2014.
- [16] Léo Ducas, Alain Durmus, Tancrede Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal Gaussians. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 40–56, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Heidelberg, Germany.
- [17] Léo Ducas, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS – Dilithium: Digital signatures from module lattices. Cryptology ePrint Archive, Report 2017/633, 2017.
- [18] Jarek Duda. Asymmetric numeral systems: entropy coding combining speed of huffman coding with compression rate of arithmetic coding, 2014.
- [19] Thomas Espitau, Mehdi Tibouchi, Alexandre Wallet, and Yang Yu. Shorter hash-and-sign lattice-based signatures. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology – CRYPTO 2022, Part II*, volume 13508 of *Lecture Notes in Computer Science*, pages 245–275, Santa Barbara, CA, USA, August 15–18, 2022. Springer, Cham, Switzerland.
- [20] Joel Gärtner. NTWE: A natural combination of NTRU and LWE. In Thomas Johansson and Daniel Smith-Tone, editors, *Post-Quantum Cryptography - 14th International Workshop, PQCrypto 2023*, pages

321–353, College Park, USA, August 16–18, 2023. Springer, Cham, Switzerland.

- [21] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 197–206, Victoria, BC, Canada, May 17–20, 2008. ACM Press.
- [22] Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.
- [23] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *Advances in Cryptology – ASIACRYPT 2009*, volume 5912 of *Lecture Notes in Computer Science*, pages 598–616, Tokyo, Japan, December 6–10, 2009. Springer, Berlin, Heidelberg, Germany.
- [24] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Heidelberg, Germany.
- [25] Vadim Lyubashevsky, Léo Ducas, Eike Kiltz, Tancrede Lepoint, Peter Schwabe, Gregor Seiler, Damien Stehlé, and Shi Bai. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [26] Peter Pessl. Analyzing the shuffling side-channel countermeasure for lattice-based signatures. In Orr Dunkelman and Somitra Kumar Sanadhya, editors, *Progress in Cryptology - INDOCRYPT 2016: 17th International Conference in Cryptology in India*, volume 10095 of *Lecture Notes in Computer Science*, pages 153–170, Kolkata, India, December 11–14, 2016. Springer, Cham, Switzerland.
- [27] Thomas Prest, Pierre-Alain Fouque, Jeffrey Hoffstein, Paul Kirchner, Vadim Lyubashevsky, Thomas Pornin, Thomas Ricosset, Gregor Seiler, William Whyte, and Zhenfei Zhang. FALCON. Technical report, National Institute of Standards and Technology, 2022. available at <https://csrc.nist.gov/Projects/post-quantum-cryptography/selected-algorithms-2022>.
- [28] Claus-Peter Schnorr. Efficient signature generation by smart cards. *Journal of Cryptology*, 4(3):161–174, January 1991.

A Missing Proofs

A.1 Proof of Lemma 5

Here, we prove Lemma 5. For reference, we begin by restating the lemma.

Lemma 5. *The function $S_{\mathbf{v}}(\mathbf{y})$ only depends on $\langle \mathbf{y}, \mathbf{v} \rangle$ and $\|\mathbf{v}\|^2$. Furthermore, for arbitrary $t \geq 1$, fixed \mathbf{v} and with $\langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$ then*

$$\left| S_{\mathbf{v}}(\mathbf{y}) - \sum_{k=0}^t (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \right| \leq \frac{\rho_r(t\mathbf{v})}{\rho_r(\mathbf{v}) - \rho_r(2\mathbf{v})}$$

Proof. To see that $S_{\mathbf{v}}(\mathbf{y})$ only depends on $\langle \mathbf{y}, \mathbf{v} \rangle$ and $\|\mathbf{v}\|^2$, note that

$$\rho_r(\mathbf{y} + k\mathbf{v}) = \exp\left(-\frac{\|\mathbf{y}\|^2 + k^2\|\mathbf{v}\|^2 + 2k\langle \mathbf{y}, \mathbf{v} \rangle}{2r^2}\right).$$

As such, we have

$$S_{\mathbf{v}}(\mathbf{y}) = \sum_{k=0}^{\infty} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} = \sum_{k=0}^{\infty} (-1)^k \exp\left(-\frac{k^2\|\mathbf{v}\|^2 + 2k\langle \mathbf{y}, \mathbf{v} \rangle}{2r^2}\right)$$

which is a function that depends only on $\|\mathbf{v}\|^2$ and $\langle \mathbf{y}, \mathbf{v} \rangle$.

Next, we note that

$$\begin{aligned} \left| S_{\mathbf{v}}(\mathbf{y}) - \sum_{k=0}^t (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \right| &= \left| \sum_{k=t+1}^{\infty} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \right| \\ &\leq \sum_{k=t+1}^{\infty} \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} = \sum_{k=t+1}^{\infty} \exp\left(-\frac{k^2\|\mathbf{v}\|^2 + 2k\langle \mathbf{y}, \mathbf{v} \rangle}{2r^2}\right) \end{aligned}$$

and when $\langle \mathbf{y}, \mathbf{v} \rangle \geq -\|\mathbf{v}\|^2$ this is upper bounded by

$$\begin{aligned} \sum_{k=t+1}^{\infty} \exp\left(-\frac{(k^2 - 2k)\|\mathbf{v}\|^2}{2r^2}\right) &= \sum_{k=t+1}^{\infty} \exp\left(-\frac{((k-1)^2 - 1)\|\mathbf{v}\|^2}{2r^2}\right) \\ &= \exp\left(\frac{\|\mathbf{v}\|^2}{2r}\right) \sum_{k=t}^{\infty} \exp\left(-\frac{k^2\|\mathbf{v}\|^2}{2r^2}\right) \\ &\leq \exp\left(-\frac{(t^2 - 1)\|\mathbf{v}\|^2}{2r^2}\right) \sum_{k=0}^{\infty} \exp\left(-\frac{k^2\|\mathbf{v}\|^2}{2r^2}\right) \\ &\leq \exp\left(-\frac{(t^2 - 1)\|\mathbf{v}\|^2}{2r^2}\right) \sum_{k=0}^{\infty} \exp\left(-\frac{k\|\mathbf{v}\|^2}{2r^2}\right) \\ &\leq \frac{\exp\left(-\frac{(t^2 - 1)\|\mathbf{v}\|^2}{2r^2}\right)}{1 - \exp\left(-\frac{\|\mathbf{v}\|^2}{2r^2}\right)} = \frac{\rho_r(t\mathbf{v})}{\rho_r(\mathbf{v}) - \rho_r(2\mathbf{v})} \end{aligned}$$

as claimed. \square

A.2 Proof of Lemma 6

Here, we prove Lemma 6. For reference, we begin by restating the lemma.

Lemma 6. *Let $\mathcal{K}_{\mathbf{v}}$ be the set of \mathbf{y} such that $|\langle \mathbf{y}, \mathbf{v} \rangle| \leq \|\mathbf{v}\|^2$ and let $\alpha = r/\|\mathbf{v}\|$. Then*

$$\max_{\mathbf{y} \in \mathcal{K}_{\mathbf{v}}} 1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} \leq 1 + \frac{2\alpha\sqrt{2\pi}\rho(\pi\alpha)}{\rho_{\alpha}(1) \cdot (1 - \rho_1(2\pi\alpha))} = M_{\alpha}$$

and M_{α} is strictly decreasing with α .

Proof of Lemma 6. To begin with, we observe that

$$\begin{aligned} \sum_{k \in \mathbb{Z}} (-1)^k \frac{\rho_r(\mathbf{y} + k\mathbf{v})}{\rho_r(\mathbf{y})} &= \sum_{k \in \mathbb{Z}} \frac{\rho_r(\mathbf{y} + 2k\mathbf{v}) - \rho_r(\mathbf{y} - (2k+1)\mathbf{v})}{\rho_r(\mathbf{y})} \\ &= 2F(2\mathbb{Z}) - F(\mathbb{Z}) \end{aligned}$$

where we define

$$\begin{aligned} F(j) &= \frac{\rho_r(\mathbf{y} + j\mathbf{v})}{\rho_r(\mathbf{y})} \\ &= \exp\left(-\frac{j^2\|\mathbf{v}\|^2 + 2j\langle\mathbf{y}, \mathbf{v}\rangle}{2r^2}\right) \\ &= \frac{\rho_{r/\|\mathbf{v}\|}(j + \langle\mathbf{y}, \mathbf{v}\rangle/\|\mathbf{v}\|^2)}{\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)}. \end{aligned}$$

With this definition of $F(j)$, we thus want to bound $1 + F(\mathbb{Z}) - 2F(2\mathbb{Z})$.

By the Poisson summation formula we have that

$$F(\mathbb{Z}) = \hat{F}(\mathbb{Z}) \text{ and } 2F(2\mathbb{Z}) = \hat{F}(\mathbb{Z}/2)$$

where \hat{F} is the Fourier transform of F . As

$$f(\mathbf{y}) = \rho_r(\mathbf{y}) = \exp(-\|\mathbf{y}\|^2/(2r^2))$$

has the Fourier transform $\hat{f}(\mathbf{w}) = r\sqrt{2\pi}\rho_{1/r}(2\pi\mathbf{w})$, we have that

$$\hat{F}(k) = \frac{r\sqrt{2\pi}}{\|\mathbf{v}\|\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)} \cdot \exp\left(2\pi i \frac{k\langle\mathbf{y}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}\right) \cdot \rho_{\|\mathbf{v}\|/r}(2\pi k)$$

This gives that

$$\begin{aligned} \hat{F}(\mathbb{Z}) &= \frac{\sqrt{2\pi} \cdot r}{\|\mathbf{v}\|\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)} \sum_{k \in \mathbb{Z}} \exp\left(2\pi i \frac{k\langle\mathbf{y}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}\right) \cdot \rho_{\|\mathbf{v}\|/r}(2\pi k) \\ &= \frac{\sqrt{2\pi} \cdot r}{\|\mathbf{v}\|\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)} \left(1 + \sum_{k=1}^{\infty} 2 \cos\left(2\pi k \frac{\langle\mathbf{y}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}\right) \cdot \rho_{\|\mathbf{v}\|/r}(2\pi k)\right) \end{aligned}$$

while

$$2F(2\mathbb{Z}) = \frac{\sqrt{2\pi} \cdot r}{\|\mathbf{v}\|\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)} \left(1 + \sum_{k=1}^{\infty} 2 \cos\left(\pi k \frac{\langle\mathbf{y}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}\right) \cdot \rho_{\|\mathbf{v}\|/r}(\pi k)\right).$$

As such, we have that $F(\mathbb{Z}) - 2F(2\mathbb{Z})$ equals

$$-\frac{\sqrt{2\pi} \cdot r}{\|\mathbf{v}\|\rho_{r/\|\mathbf{v}\|}(\langle\mathbf{y}, \mathbf{v}\rangle)} \sum_{k=1}^{\infty} 2 \cos\left((2k-1)\pi \frac{\langle\mathbf{y}, \mathbf{v}\rangle}{\|\mathbf{v}\|^2}\right) \cdot \rho_{\|\mathbf{v}\|/r}((2k-1)\pi)$$

which in the relevant regime is maximized when $|\langle\mathbf{y}, \mathbf{v}\rangle| = \|\mathbf{v}\|^2$ and we thus have

$$1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} \leq 1 + \frac{2\sqrt{2\pi} \cdot r}{\|\mathbf{v}\|\rho_r(\|\mathbf{v}\|)} \sum_{k=1}^{\infty} \rho_{\|\mathbf{v}\|/r}((2k-1)\pi) \quad (9)$$

Finally, by introducing $\alpha = r/\|\mathbf{v}\|$ we get

$$\begin{aligned}
1 - \sum_{k \in \mathbb{Z}} (-1)^k \frac{p(\mathbf{y} + k\mathbf{v})}{p(\mathbf{y})} &\leq 1 + \frac{2\sqrt{2\pi} \cdot \alpha}{\rho_\alpha(1)} \sum_{k=1}^{\infty} \rho_{1/\alpha}((2k-1)\pi) \\
&= 1 + \frac{2\sqrt{2\pi} \cdot \alpha}{\rho_\alpha(1)} \sum_{k=1}^{\infty} \exp(-\pi^2 \alpha^2 (4k^2 - 4k + 1)/2) \\
&= 1 + \frac{2\sqrt{2\pi} \cdot \alpha \rho(\pi\alpha)}{\rho_\alpha(1)} \sum_{k=1}^{\infty} \exp(-2\pi^2 \alpha^2 (k^2 - k)) \\
&\leq 1 + \frac{2\sqrt{2\pi} \cdot \alpha \rho(\pi\alpha)}{\rho_\alpha(1)} \sum_{k=0}^{\infty} \exp(-2\pi^2 \alpha^2 k) \\
&= 1 + \frac{2\sqrt{2\pi} \cdot \alpha \rho(\pi\alpha)}{\rho_\alpha(1)} \frac{1}{1 - \exp(-2\pi^2 \alpha^2)} \\
&= 1 + \frac{2\alpha \sqrt{2\pi} \rho(\pi\alpha)}{\rho_\alpha(1) \cdot (1 - \rho_1(2\pi\alpha))} = M_\alpha.
\end{aligned}$$

which gives our expression for M_α

To see that M_α is strictly decreasing with α , we note that all factors in the second term, besides α , are decreasing with α . Furthermore, we note that $\alpha/\rho_\alpha(1) = \alpha \exp(1/(2\alpha^2))$ is decreasing with α when $\alpha \leq 1$ while, when $\alpha > 1$ the product

$$\alpha \cdot \rho_1(\pi\alpha) = \alpha \exp(-\pi^2 \alpha^2 / 2)$$

is decreasing with α . As such, the second term is decreasing with α for all $\alpha > 0$, and the whole expression is thus also decreasing with α . \square

A.3 Proof of Lemma 8

To prove Lemma 8, we first prove the following property of the LowBits function.

Lemma 13. *Let $k > 0$ be an integer, q be a prime and τ an integer that divides $2(q-1)$. Then, it holds that*

$$\|\text{LowBits}(\mathbf{w} \bmod 2q)\|_\infty \leq \tau/2 + 2$$

for arbitrary $\mathbf{w} \in \mathcal{R}^k$.

Proof. Without loss of generality, we assume that $\mathbf{w} = \mathbf{w} \bmod 2q$ and thus $\|\mathbf{w}\|_\infty < 2q$. By the triangle inequality, we can then upper bound $\|\text{LowBits}(\mathbf{w})\|_\infty = \|\mathbf{w} - \text{HighBits}(\mathbf{w}) \bmod^{\pm} 2q\|_\infty$ by

$$\|\mathbf{w} - \lfloor \mathbf{w}/\tau \rfloor \cdot \tau\|_\infty + \|\lfloor \mathbf{w}/\tau \rfloor \cdot \tau - \text{HighBits}(\mathbf{w}) \bmod^{\pm} 2q\|_\infty.$$

In this expression, the first term $\|\mathbf{w} - \lfloor \mathbf{w}/\tau \rfloor \cdot \tau\|_\infty$ is at most $\tau/2$.

For the second term, we note that, as $\|\mathbf{w}\|_\infty < 2q$, $\text{HighBits}(\mathbf{w})$ and $\lfloor \mathbf{w}/\tau \rfloor \cdot \tau \bmod^{\pm} 2q$ differ only for coefficients c of an element of \mathbf{w} such that $\lfloor c/\tau \rfloor \cdot \tau = 2(q-1)$. For coordinates that do differ, the difference is between $-2 = 2(q-1) \bmod^{\pm} 2q$ and $0 = 2q \bmod^{\pm} 2q$, which is thus a difference of 2 modulo $2q$. As such, $\|\text{LowBits}(\mathbf{w})\|_\infty \leq \tau/2 + 2$. \square

Next, we prove Lemma 8, and thus that all generated signatures verify as expected. For reference, we begin by restating the lemma.

Lemma 8. Let $(\mathbf{A}, \mathbf{s}) \leftarrow \text{KeyGen}()$ and let $B_v \geq B_s + \sqrt{nm}(\tau/4 + 1)$. Then, for every message $\mu \in \mathcal{M}$ the verification

$$C\text{Verify}(C\text{Sign}(\mathbf{A}, \mathbf{s}, \mu), \mathbf{A}, \mu)$$

always succeeds.

Proof. For the signature $(\mathbf{z}_1, \mathbf{h}, c)$ to be accepted, it is required that the recomputed c' equals c and that the norm of the recomputed \mathbf{z}' vector is sufficiently short.

The recomputed challenge c' and the original challenge c match if

$$\text{HighBits}(\tilde{\mathbf{w}}') + \mathbf{h} \bmod 2q = \text{HighBits}(\mathbf{A}\mathbf{y} \bmod 2q) \quad (10)$$

and $\text{LSB}(\mathbf{A}\mathbf{y} \bmod 2q) = \text{LSB}(z_0 - c)\mathbf{j}$. To see that (10) holds, note that

$$\mathbf{A}_1 \mathbf{z}_1 \bmod 2q = \mathbf{A}\mathbf{z} - \mathbf{A}_2 \mathbf{z}_2 \bmod 2q = \mathbf{A}\mathbf{y} + \mathbf{A}\mathbf{s}c - 2\mathbf{z}_2 \bmod 2q$$

and we thus have

$$\tilde{\mathbf{w}}' = \mathbf{A}_1 \mathbf{z}_1 - qc\mathbf{j} \bmod 2q = \mathbf{w} - 2\mathbf{z}_2 \bmod 2q = \tilde{\mathbf{w}}$$

as $\mathbf{A}\mathbf{s} = q\mathbf{j}$. With

$$\mathbf{w}' = \text{HighBits}(\tilde{\mathbf{w}}') + \mathbf{h} \bmod 2(q-1) = \text{HighBits}(\tilde{\mathbf{w}}) + \mathbf{h} \bmod 2(q-1)$$

the definition of \mathbf{h} directly gives us that $\text{HighBits}(\mathbf{w}) = \mathbf{w}'$.

Next, note that $\mathbf{A} \bmod 2 = [\mathbf{j}, \mathbf{0}, \mathbf{0}]$ and thus, for every vector \mathbf{x} ,

$$\text{LSB}(\mathbf{A}\mathbf{x} \bmod 2q) = \text{LSB}(\mathbf{A}\mathbf{x}) = x_0\mathbf{j} \bmod 2,$$

where x_0 is the first element of \mathbf{x} . Furthermore, $\mathbf{z} \equiv \mathbf{y} + \mathbf{s}c \bmod 2$, and thus $\text{LSB}(z_0) = y_0 + s_0c \bmod 2$, where z_0, y_0 and s_0 are the first elements of \mathbf{z}, \mathbf{y} and \mathbf{s} respectively. As $s_0 = f = 2f + 1 \equiv 1 \bmod 2$, this gives that

$$\text{LSB}(z_0 - c)\mathbf{j} = \text{LSB}(y_0)\mathbf{j} = \text{LSB}(\mathbf{A}\mathbf{y} \bmod 2q) = \text{LSB}(\mathbf{w})$$

as desired.

Finally we show that, for correctly generated signatures, the recomputed $\mathbf{z}' = [\mathbf{z}'_1, \mathbf{z}'_2]^T$ vector always has norm at most B_v . During signing, the size of $\mathbf{z} = [\mathbf{z}_1, \mathbf{z}_2]^T$ is bounded by $\|\mathbf{z}\| \leq B_s$. We want to bound $\|\mathbf{z}_2 - \mathbf{z}'_2\|$ and to this end make use of the fact that

$$\mathbf{w} - \tilde{\mathbf{w}}' = \mathbf{w} - \tilde{\mathbf{w}} \equiv 2\mathbf{z}_2 \bmod 2q.$$

Since $\mathbf{z}'_2 = (\mathbf{w}' - \tilde{\mathbf{w}}' + \mathbf{w}_0)/2 \bmod^\pm q$, we thus have that

$$\mathbf{z}'_2 = \mathbf{z}_2 + (\text{LowBits}(\mathbf{w}) - \mathbf{w}_0)/2 \bmod^\pm q \quad (11)$$

by using that $\mathbf{w}' = \text{HighBits}(\mathbf{w})$ and the definition of LowBits .

To bound $\|\mathbf{z}'_2\|$ we first note that, $\|\mathbf{x} \bmod^\pm q\| \leq \|\mathbf{x}\|$ for every vector \mathbf{x} . This gives that

$$\|\mathbf{z}'_2\| \leq \|\mathbf{z}_2\| + \|\text{LowBits}(\mathbf{w}) + \mathbf{w}_0\|/2$$

and it follows that

$$\|\mathbf{z}'\| \leq \|\mathbf{z}\| + \sqrt{nm}(\tau/4 + 2) \leq B_v$$

by using $\|\text{LowBits}(\mathbf{w}) + \mathbf{w}_0\|^2 \leq nm\|\text{LowBits}(\mathbf{w}) + \mathbf{w}_0\|_\infty^2$ which is then bounded by the triangle inequality with Lemma 13 and by using that $\|\mathbf{w}_0\|_\infty \leq 1 < 2$. As such, all correctly generated signatures satisfy that the recomputed \mathbf{z}' has norm at most B_v and are thus accepted by the verification procedure. \square

B rANS Encoding

To encode our signatures, we make use of the rANS encoding of Duda [18]. This leads to signature sizes that are close to the entropy of the distribution that the signatures follow. As described in Section 4.2.2, rANS is used to encode the high order bits of \mathbf{z} and the full hint vector \mathbf{h} . The procedure in both cases is almost the same, but with the coding function parametrized differently, due to \mathbf{h} and the high order bits of \mathbf{z} following different distributions.

The encoding is parametrized by a precision 2^k , where we select $k = 16$ in our test implementation that we use to test sizes of signatures in practice. We then define $f(s)$ as an integer approximately equal to $2^k \cdot p(s)$ where $p(s)$ is an estimate for the probability that an entry of the data to encode is given by the symbol s .

For the encoding of the higher order bits of \mathbf{z} , we determine $p(s)$ by summing over the different outcomes of a discrete Gaussian distribution that correspond to the rounded coefficient s . For the encoding of \mathbf{h} we determine $f(s)$ similarly, but account for the coefficients of $\text{LowBits}(\mathbf{w})$ by assuming that they are uniformly random. It is only the difference in $p(s)$ that differs between the encoding of the high bits of \mathbf{z}_1 and the encoding of \mathbf{h} .

Although not defined as such, we can think of the symbols to encode, corresponding to the coefficients of \mathbf{h} and the high bits of \mathbf{z}_1 , as sequential integers. This allows us to define $\text{CDF}(s) = \sum_{v < s} f(v)$ and $s = \text{symbol}(y)$ as the unique symbol s such that $\text{CDF}(s) \leq y < \text{CDF}(s + 1)$.

The coding function $C(x, s)$ is defined via

$$C(x, s) = 2^k \cdot \left\lfloor \frac{x}{f(s)} \right\rfloor + (x \bmod (f(s))) + \text{CDF}(s)$$

Meanwhile, the decoding function $D(x)$ first determines the symbol to decode as $s = \text{symbol}(x \& (2^k - 1))$ where $\&$ is the bitwise AND operation. The decoding function is then defined via

$$D(x) = \left(f(s) \cdot \left\lfloor x \cdot 2^{-k} \right\rfloor + (x \& (2^k - 1)) - \text{CDF}(s), s \right).$$

To encode a vector over \mathcal{R} , we begin with $x = x_0$ for arbitrary positive integer x_0 as initial value. A coefficient s of an entry in the vector is encoded via $x = C(x, s)$, thus updating the integer x . This continues for every coefficient of every entry of the vector, resulting in some large integer $x = x_e$ as the final output of the encoding.

Decoding begins with a large integer $x = x_e$, applies the decoding function to get (x, s) for a new x and a decoded symbol s . This continues until the expected number of symbols have been decoded, which results in $x = x_0$. This gives the same sequence of symbols that were encoded into x_e , although returned in reverse.

As the encoding can start from an arbitrary initial value x_0 , we can encode both \mathbf{h} and the high bits of \mathbf{z} into a single value. This is accomplished by first encoding the high bits of \mathbf{z} with $x_0 = 0$ to get some value $x = x_z$. Then, the hint is similarly encoded, but starting with $x = x_z$ resulting in the final state $x = x_{zh}$. To decode, the hint vector is first recovered via the decoding function starting from x_{zh} , with the final decoding function call resulting in $x = x_z$. The high bits of \mathbf{z} can then be decoded from this value x_z , thus recovering all the encoded data.

C Parameters for Lattice Estimator

Normal Parametrs

For 120 bits of security

Unforgeability

SISParameters(n=512, q=3329, length_bound=3195, m=1024, norm=2, tag=None)

Strong unforgeability

SISParameters(n=512, q=3329, length_bound=6390, m=1024, norm=2, tag=None)

NTWE security

LWEParameters(n=511, q=3329, Xs=D($\sigma=1.50$), Xe=D($\sigma=1.50$), m=+Infinity, tag=None)

For 180 bits of security

Unforgeability

SISParameters(n=512, q=50177, length_bound=2946, m=1536, norm=2, tag=None)

Strong unforgeability

SISParameters(n=512, q=50177, length_bound=5892, m=1536, norm=2, tag=None)

NTWE security

LWEParameters(n=1023, q=50177, Xs=D($\sigma=1.00$), Xe=D($\sigma=1.00$), m=+Infinity, tag=None)

For 260 bits of security

Unforgeability

SISParameters(n=768, q=50177, length_bound=5300, m=2048, norm=2, tag=None)

Strong unforgeability

SISParameters(n=768, q=50177, length_bound=10600, m=2048, norm=2, tag=None)

NTWE security

LWEParameters(n=1279, q=50177, Xs=D($\sigma=1.50$), Xe=D($\sigma=1.50$), m=+Infinity, tag=None)

Parametrs Without Rejection

For 120 bits of security

Unforgeability

SISParameters(n=512, q=12289, length_bound=5850, m=1280, norm=2, tag=None)

Strong unforgeability

SISParameters(n=512, q=12289, length_bound=11700, m=1280, norm=2, tag=None)

NTWE security

LWEParameters(n=767, q=12289, Xs=D($\sigma=0.65$), Xe=D($\sigma=0.65$), m=+Infinity, tag=None)

For 180 bits of security

Unforgeability

SISParameters(n=768, q=50177, length_bound=18349, m=1792, norm=2, tag=None)

Strong unforgeability

SISParameters(n=768, q=50177, length_bound=36698, m=1792, norm=2, tag=None)

NTWE security

LWEParameters(n=1023, q=50177, Xs=D($\sigma=1.00$), Xe=D($\sigma=1.00$), m=+Infinity, tag=None)

For 260 bits of security

Unforgeability

SISParameters(n=1024, q=50177, length_bound=20685, m=2304, norm=2, tag=None)

Strong unforgeability

SISParameters(n=1024, q=50177, length_bound=41370, m=2304, norm=2, tag=None)

NTWE security

LWEParameters(n=1279, q=50177, Xs=D($\sigma=1.45$), Xe=D($\sigma=1.45$), m=+Infinity, tag=None)