An improved compression technique for signatures based on learning with errors

Shi Bai and Steven D. Galbraith

Department of Mathematics,
University of Auckland,
New Zealand.
S.Bai@auckland.ac.nz
S.Galbraith@math.auckland.ac.nz

Abstract. We present a new approach to the compression technique of Lyubashevsky et al. [18, 14] for lattice-based signatures based on learning with errors (LWE). Our ideas seem to be particularly suitable for signature schemes whose security, in the random oracle model, is based on standard worst-case computational assumptions. Our signatures are shorter than any previous proposal for provably-secure signatures based on standard lattice problems: at the 128-bit level we improve signature size from (more than) 16500 bits to around 9000 to 12000 bits.
This is the full version of the paper. The conference version was published at CT-RSA 2014.

Keywords: Lattice-based signatures, learning with errors.

1 Introduction

An important problem is to obtain practical and provably secure public key signature schemes based on lattice assumptions. One approach is to use trapdoor functions and the hash-and-sign methodology (see Gentry, Peikert and Vaikuntanathan [13], Stehlé and Steinfeld [23]). However, the most promising avenue for practical signatures (with security in the random oracle model) has long been the use of the Fiat-Shamir paradigm; this is the approach used for all currently deployed discrete-logarithm-based digital signature schemes. (For signatures that are proven secure in the standard model, we refer to Boyen [7] and Böhl et al. [6].)

A series of works by Lyubashevsky and others [16, 18, 14, 10] have developed schemes based on the Fiat-Shamir paradigm that are secure in the random oracle model. There are several challenges when implementing lattice-based signature schemes, including the size of the public key, the size of the signature and the requirement to sample from discrete Gaussians during the signing process. Our main focus in this paper is to reduce the size of signatures.

The basic idea of Lyubashevsky's signatures in the case of LWE is to have a public key of the form $(\mathbf{A}, \mathbf{T} = \mathbf{AS} + \mathbf{E} \pmod q)$ where $\mathbf{A}$ is an $m \times n$ matrix and $m \approx n$. The signing procedure starts by choosing vectors $\mathbf{y}_1, \mathbf{y}_2$ of

small norm and computing $\mathbf{v} = \mathbf{A}\mathbf{y}_1 + \mathbf{y}_2 \pmod q$. Then, using the Fiat-Shamir paradigm, the signer computes $\mathbf{c} = H(\mathbf{v}, \mu)$ where μ is the message and H is a hash function. Finally, the signer computes $\mathbf{z}_1 = \mathbf{y}_1 + \mathbf{S}\mathbf{c}$ and $\mathbf{z}_2 = \mathbf{y}_2 + \mathbf{E}\mathbf{c}$. The signature is $(\mathbf{z}_1, \mathbf{z}_2, \mathbf{c})$. The verifier checks that $\|\mathbf{z}_1\|$ and $\|\mathbf{z}_2\|$ are small enough and that $H(\mathbf{A}\mathbf{z}_1 + \mathbf{z}_2 - \mathbf{T}\mathbf{c} \pmod q, \mu)$ is equal to $\mathbf{c}$. A significant obstacle to short signatures is the need to send the length m vector $\mathbf{z}_2$. Recent work [14, 10] has introduced compression techniques that greatly reduce the amount of data to be sent for the vector $\mathbf{z}_2$. The main contribution of our paper is to give a variant of the signature scheme with the feature that $\mathbf{z}_2$ can be omitted entirely.

1.1 Related work

At Eurocrypt 2012, Lyubashevsky [18] gave a signature scheme whose security (at around the 100-bit security level) relies on SIS and LWE, and for which signatures are 16500 bits. To our knowledge, this is the current record in the literature for signatures whose security is reduced to worst-case assumptions in general lattices. The signing algorithm for that scheme requires sampling from discrete Gaussians. At the 128-bit security level, signatures for this scheme would be around 20000 bits.

Güneysu, Lyubashevsky and Pöppelmann [14] introduced an important compression technique and gave a signature scheme that does not require sampling from Gaussians. The security depends on the Ring-SIS and DCK (an NTRU-like variant of Ring-LWE with small parameters) assumptions, however a full security analysis is not given in their paper. The signatures are around 9000 bits. The compression technique can be modified to shorten the signatures from [18] (of course, it is necessary to change the rejection sampling used in [14] to a Gaussian distribution).

Recently Ducas, Durmus, Lepoint and Lyubashevsky [10] have given a new scheme with several further tricks to reduce the signature size. For security based on SIS (and hence on standard worst-case lattice problems) their scheme has signatures of size more than 20000 bits[1]. They also give a variant, based on a non-standard computational assumption related to NTRU, that has signatures of around 5000 bits.

1.2 Our contribution

As mentioned already, our main contribution is to present a variant of the Lyubashevsky signature scheme based on LWE that does not require sending any information about the $\mathbf{z}_2$ vector. At a high level, Lyubashevsky's scheme [14, 18] based on LWE has public key $(\mathbf{A}, \mathbf{b} = \mathbf{A}\mathbf{s} + \mathbf{e} \pmod q)$ and a signature is like a proof of knowledge of the pair $(\mathbf{s}, \mathbf{e})$. The key feature of our scheme is to prove

[1] It may seem paradoxical that the improved techniques of [10] lead to larger signatures than [18]. This is due to the requirement that the matrix $\mathbf{A}$ in the public key be indistinguishable from a uniformly chosen matrix, which makes m larger. Indeed, it is necessary to take $m = O(\frac{n \log q}{\log n})$.

knowledge of only $\mathbf{s}$. The smallness of $\mathbf{e}$ becomes implicit in the verification equation, so we no longer need to send any information about $\mathbf{e}$. Since $\mathbf{s}$ has length n and $\mathbf{e}$ has length $m \approx n$, not needing to prove knowledge of $\mathbf{e}$ has the potential to provide a significant reduction in signature size.

Briefly, the public key for our scheme is an LWE instance $(\mathbf{A}, \mathbf{T} = \mathbf{AS} + \mathbf{E} \pmod q)$ where all terms are matrices, and $\mathbf{S}, \mathbf{E}$ have small entries. A signature is formed by first choosing a vector $\mathbf{y}$ and computing $\mathbf{v} = \mathbf{Ay} \pmod q$. One then throws away the least significant bits of $\mathbf{v}$ and hashes the remaining bits together with the message μ to get a hash value c. The value c is used to create a low weight vector $\mathbf{c}$ and the signature is the pair $(\mathbf{z} = \mathbf{y} + \mathbf{Sc}, c)$. As in [10, 14, 18], we use rejection sampling to ensure that the distribution of $\mathbf{z}$ is independent of the secret. To verify the signature one computes $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} \equiv \mathbf{Ay} - \mathbf{Ec} \pmod q$. Assuming that $\mathbf{Ec}$ is small enough then the most significant bits of $\mathbf{w}$ will match those of $\mathbf{v}$ and so the hash value computed using the most significant bits of $\mathbf{w}$ equals c.

Our work employs several ideas from [10, 14, 18]. We prove the security of our scheme using the proof methodology from [18].

For signatures based on worst-case lattice assumptions we improve signature size from more than 16500 bits to around 9000 bits. The security level (at the 128 bits-level) of our signatures is supported by Regev's reduction for LWE and also arguments about BKZ 2.0 lattice reduction due to Chen and Nguyen [9]. Hence, we match the signature size of [14], and still with the beneficial feature of using uniform distributions, but with security based on standard assumptions. Relaxing the conditions of Regev's theorem also allows signatures of size under 10000 bits (see Section B.2). We also give signatures of under 8000 bits with security based on a non-standard matrix-NTRU-like problem (see Section B.3).

One problematic aspect of our result is that we use standard LWE rather than Ring-LWE. Previous work on lattice signatures assumed that using Ring-LWE or NTRU would give more practical signatures. While there are certainly significant practical benefits from using Ring-LWE (such as smaller public keys), there are also some constraints (such as preferring n to be a power of 2), and so the Ring-LWE case is a little less flexible. To conclude, the two advantages of using standard LWE are: we get security based on standard assumptions in general lattices; we have complete flexibility in the parameters (n, m) for LWE (rather than being stuck with $m = 2n$ and $n = 2^d$). Nevertheless, it is interesting to consider implementing our scheme using Ring-LWE or NTRU. For the parameters in Table 1 one can see that column IV can be easily implemented using Ring-LWE with $n = 512$ and $m = 1024$, and signatures should still be around 12400 bits. Column V invites an implementation of the signature scheme using a ring $\mathbb{Z}_q[x]/(F(x))$ where $\deg(F(x)) = 400$; it is an interesting question to choose a suitable polynomial $F(x)$ of this degree that enables fast implementation.

2 Preliminaries

2.1 Basic notation and Gaussians

Let $q \in \mathbb{N}$ be a prime. We write $\mathbb{Z}_q$ for the integers modulo q and represent this set by integers in the range $(-q/2, q/2]$. We write (column) vectors in bold face as $\mathbf{v} = (v_1, \ldots, v_n)^T$, where $\mathbf{v}^T$ denotes the transpose of the vector, and matrices in bold face as $\mathbf{A}$. The $n \times n$ identity matrix is denoted $\mathbf{I}_n$. The Euclidean norm is $\|\mathbf{v}\| = \|\mathbf{v}\|_2 = \sqrt{\sum_{i=1}^{n} v_i^2}$ and the infinity norm (or sup norm) is $\|\mathbf{v}\|_\infty = \max_{1 \leq i \leq n} |v_i|$.

For $a \in \mathbb{Z}$ and $d \in \mathbb{N}$, define $[a]_{2^d}$ to be the unique integer in the set $(-2^{d-1}, 2^{d-1}]$ such that $a \equiv [a]_{2^d} \pmod{2^d}$. For $a \in \mathbb{Z}$, we define $\lfloor a \rceil_d = (a - [a]_{2^d})/2^d$ (dropping the d-least significant bits). Note that it satisfies $\lfloor 2^{d-1} \rceil_d = \lfloor -2^{d-1} + 1 \rceil_d = 0$ and $\lfloor 2^{d-1} + 1 \rceil_d = -\lfloor -2^{d-1} \rceil_d = 1$. We extend this function to vectors: on input a length m vector $\mathbf{v} = (v_1, \ldots, v_m)^T \in \mathbb{Z}^m$ the function $\lfloor \mathbf{v} \rceil_d$ is the length m vector with entries $\lfloor v_i \rceil_d$. A lattice in $\mathbb{Z}^m$ is a subgroup of $\mathbb{Z}^m$; for background see [19, 20].

Let A be a finite set. We write $a \leftarrow A$ to denote that a is sampled uniformly from A. We write $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$ to denote that $\mathbf{A}$ is an $m \times n$ matrix with entries uniformly and independently sampled from $\mathbb{Z}_q$. Let $\sigma \in \mathbb{R}_{>0}$. Define $\rho_\sigma(x) = \exp(-x^2/(2\sigma^2))$ and $\rho_\sigma(\mathbb{Z}) = 1 + 2\sum_{x=1}^{\infty} \rho_\sigma(x)$. The discrete Gaussian distribution on $\mathbb{Z}$ with standard deviation σ is the distribution that associates to $x \in \mathbb{Z}$ the probability $\rho_\sigma(x)/\rho_\sigma(\mathbb{Z})$. We denote this distribution D_σ. Some authors write $s = \sqrt{2\pi}\sigma$ and define $\rho_s(x) = \exp(-\pi x^2/s^2)$ and denote the distribution D_s. The tail of a discrete Gaussian variable can be bounded by the following result.

Lemma 1. *(Lemma 4.4, full version of [18]) For any $k > 0$,*

$$Pr_{x \leftarrow D_\sigma}(|x| > k\sigma) \leq 2e^{-k^2/2}. \tag{1}$$

Taking $k = 13$ gives tail probability approximately 2^{-121}, taking $k = 13.5$ gives 2^{-130} and $k = 14$ gives 2^{-140}.

One can also define discrete Gaussian distributions on vectors. We write $\mathbf{y} \leftarrow D_\sigma^n$ to mean that the vector $\mathbf{y} = (y_1, \ldots, y_n)^T \in \mathbb{Z}^n$ is sampled such that each entry y_i is independently sampled according to the distribution D_σ.

2.2 Learning with errors

The learning with errors problem (LWE) was introduced by Regev [24]. It is parameterised by integers $n, q \in \mathbb{N}$ and distributions χ and ϕ on $\mathbb{Z}$ (typically χ is the uniform distribution on $\mathbb{Z}_q$ and $\phi = D_{\alpha q}$ for some fixed real number $0 < \alpha < 1$).

Definition 1. *Let $n, q \in \mathbb{N}$ and let χ and ϕ be distributions on $\mathbb{Z}$. The LWE distribution for a given vector $\boldsymbol{s} \in \mathbb{Z}_q^n$ is the set of pairs $(\boldsymbol{a}, \boldsymbol{a} \cdot \boldsymbol{s} + e \pmod{q})$ where $\boldsymbol{a} \in \mathbb{Z}_q^n$ is sampled uniformly and where e is sampled from ϕ.*

- *The computational-LWE problem is: For a vector* $\boldsymbol{s} \leftarrow \chi^n$ *and given arbitrarily many samples from the LWE distribution for* $\boldsymbol{s}$*, to compute* $\boldsymbol{s}$*.*
- *The decisional-LWE problem is: Given arbitrarily many samples from* $\mathbb{Z}_q^{n+1}$ *to distinguish whether the samples are distributed uniformly or whether they are distributed as the LWE distribution for some fixed vector* $\boldsymbol{s} \leftarrow \chi^n$*.*

We sometimes use notation like (n, q, ϕ)-LWE to mean the computational LWE problem with these parameters. We also write (n, q, α)-LWE to mean LWE where $\phi = D_{\alpha q}$.

If the error distribution is small enough compared with q and if one has enough samples from the LWE-distribution then it can be shown that these computational problems are well-defined. Well-defined for decisional-LWE means that the LWE-distribution, for all vectors likely to be sampled as $\mathbf{s} \leftarrow \chi^n$, is not statistically close to the uniform distribution. Well-defined for computational-LWE means that there is a unique solution $\mathbf{s}$ that is most likely to be the one used to generate the samples from the LWE distribution (in other words, computational-LWE is well-defined as a maximum likelihood problem). There is a reduction (Lemma 4.2 of Regev [25]) from the computational-LWE problem to the decisional-LWE problem. So if one problem is hard then so is the other.

Regev's main theorem is that the LWE problems are as hard as worst-case assumptions in general lattices when χ is the uniform distribution and when ϕ is a discrete Gaussian with standard deviation $\sigma = \alpha q$ for some fixed real number $0 < \alpha < 1$.

Theorem 1. *(Regev) Let* $n, q \in \mathbb{N}$ *and* $0 < \alpha < 1$ *be such that* $\alpha q \geq 2\sqrt{n}$*. Then there exists a quantum reduction from worst-case* $GapSVP_{\tilde{O}(n/\alpha)}$ *to* (n, q, α)*-LWE.*

One can also fix an integer m and consider the case of *LWE with a bounded number of samples*. We often write the LWE instance in this case as $(\mathbf{A}, \mathbf{b} \equiv \mathbf{As} + \mathbf{e} \pmod q)$ where $\mathbf{A}$ is an $m \times n$ matrix over $\mathbb{Z}_q$, $\mathbf{s}$ is a length n column vector, and $\mathbf{e}$ is a length m vector with entries sampled independently from ϕ. As long as the bounded samples LWE instance is well-defined then this problem cannot be easier than the general LWE instance. Consider the bounded samples LWE problem when χ is the uniform distribution on $\mathbb{Z}_q$ and when ϕ is such that error values satisfy $|e| \leq E$ with overwhelming probability (in our application we will have $E = 2^{d-1}$ or $E = 2^d$). Then there are at most $q^n(2E+1)^m$ choices for $(\mathbf{s}, \mathbf{e})$ compared with q^m choices for $\mathbf{b}$. Hence, as a rule of thumb, we need $q^m > q^n(2E+1)^m$ for the bounded samples LWE problem to be well-defined.

Another well-known fact (see [2]) is that one may reduce LWE to the case where $\chi = \phi$. Suppose we have m samples, where m is significantly larger than n, and write the LWE instance as $(\mathbf{A}, \mathbf{b} \equiv \mathbf{As} + \mathbf{e} \pmod q)$. With overwhelming probability, $\mathbf{A}$ has rank n and (swapping rows of $\mathbf{A}$ if necessary) we may write

$$\mathbf{A} = \begin{pmatrix} \mathbf{A}_1 \\ \mathbf{A}_2 \end{pmatrix}$$

where $\mathbf{A}_1$ is an invertible $n \times n$ matrix and $\mathbf{A}_2$ is an $(m-n) \times n$ matrix. Write $\mathbf{b} = (\frac{\mathbf{b}_1}{\mathbf{b}_2})$ and $\mathbf{e} = (\frac{\mathbf{e}_1}{\mathbf{e}_2})$ where $\mathbf{b}_1$ and $\mathbf{e}_1$ have length n and we have $\mathbf{b}_1 = \mathbf{A}_1\mathbf{s}+\mathbf{e}_1$ and $\mathbf{b}_2 = \mathbf{A}_2\mathbf{s} + \mathbf{e}_2$. It follows that

$$\mathbf{b}_2 - \mathbf{A}_2\mathbf{A}_1^{-1}\mathbf{b}_1 = (-\mathbf{A}_2\mathbf{A}_1^{-1})\mathbf{e}_1 + \mathbf{e}_2 \pmod q$$

which gives an LWE instance where the solution $(\mathbf{e}_1, \mathbf{e}_2)$ is sampled from the error distribution. We call this problem *LWE with short secrets*.

It follows that LWE with short secrets is not easier than the general case. We can also consider the LWE problem with short secrets *and* with a bounded number of samples. As long as this problem is well defined then it is also not easier than the general case. Furthermore, fewer samples are required for the LWE with short secrets problem to be well-defined: If we again assume the distribution ϕ is such that error values satisfy $|e| \leq E$ with overwhelming probability, then we need, as a rule of thumb, $q^m > (2E+1)^{n+m}$ for the LWE problem to be well-defined. To get very short signatures one can push this further and have the distribution χ having smaller support than the error distribution ϕ (*cf.* Appendix B.2).

In our work we will consider a matrix variant of LWE. The LWE distribution is on pairs $(\mathbf{A}, \mathbf{AS}+\mathbf{E} \pmod q)$ where $\mathbf{S}$ and $\mathbf{E}$ are matrices. Each of the columns of $\mathbf{S}$ and $\mathbf{E}$ corresponds to an LWE instance $(\mathbf{A}, \mathbf{As}+\mathbf{e} \pmod q)$, so this is just a collection of individual LWE instances. However, note that the matrix $\mathbf{A}$ is shared across all instances; we call them *semi-independent instances of LWE*. In any case, it is clear that this matrix variant of LWE cannot be easier than LWE with a single vector $\mathbf{b} \equiv \mathbf{As} + \mathbf{e} \pmod q$.

To summarise, we may choose (n, q, α) such that $\alpha q > 2\sqrt{n}$ and set $\chi = \phi = D_{\alpha q}$. We should choose m such that $q^m > (28\alpha q)^{n+m}$ so that the problem is well-defined with overwhelming probability. Consider the computational LWE problem $(\mathbf{A}, \mathbf{AS} + \mathbf{E} \pmod q)$ where $\mathbf{A}$ is an $m \times n$ matrix uniformly chosen from $\mathbb{Z}_q^{m \times n}$ and where $\mathbf{S}$ and $\mathbf{E}$ are chosen to have entries sampled independently from $D_{\alpha q}$. Then this problem is not easier than $\mathrm{GapSVP}_{\tilde{O}(n/\alpha)}$ in n-dimensional lattices.

2.3 Rejection sampling

For security we will need to ensure that the signatures do not leak the private key. We use a variant of the general rejection sampling lemma of [18] (also see Chapter 2 of Devroye [11]).

Lemma 2. *Let $f : \mathbb{Z}^n \to \mathbb{R}$ be a probability distribution. Given a subset $V \subseteq \mathbb{Z}^n$, let $h : V \to \mathbb{R}$ be a probability distribution defined on V. Let $g_{\boldsymbol{v}} : \mathbb{Z}^n \to \mathbb{R}$ be a family of probability distributions indexed by $\boldsymbol{v} \in V$ such that for almost all $\boldsymbol{v}$'s from h there exists a universal upper bound $M \in \mathbb{R}$ such that*

$$Pr[Mg_{\boldsymbol{v}}(\boldsymbol{z}) \geq f(\boldsymbol{z});\ \boldsymbol{z} \leftarrow f] \geq 1 - \textit{negligible}.$$

Then the output distributions of the following two algorithms have negligible statistical difference:

1. $\boldsymbol{v} \leftarrow h$, $\boldsymbol{z} \leftarrow g_{\boldsymbol{v}}$, *output* $(\boldsymbol{z}, \boldsymbol{v})$ *with probability* $\min\left(\frac{f(\boldsymbol{z})}{Mg_{\boldsymbol{v}}(\boldsymbol{z})}, 1\right)$, *else fail.*
2. $\boldsymbol{v} \leftarrow h, \boldsymbol{z} \leftarrow f$, *output* $(\boldsymbol{z}, \boldsymbol{v})$ *with probability* $\frac{1}{M}$.

In the signature (and the security proof), distribution f is a uniform distribution over $[-B+U, B-U]^n$ where $U = 14\sigma_{\mathbf{Sc}}$. Each $\mathbf{v} = \mathbf{Sc}$ is a vector with entries in a close-to-Gaussian distribution with standard deviation $\sigma_{\mathbf{Sc}}$. With high probability, the coefficients in $\mathbf{v}$ are bounded by $14\sigma_{\mathbf{Sc}}$. This accounts for the "almost all" argument in above lemma. In the signature $\mathbf{z} = \mathbf{Sc} + \mathbf{y}$, vector $\mathbf{y}$ is generated from a uniform distribution over $[-B, B]^n$ (so each entry is set for 2^{-140} error). For success probability of roughly $1/e$, we can set $B = 14\sigma_{\mathbf{Sc}}n$.

3 Our signature scheme

In Appendix A, we recall some standard background on signature schemes. We will focus on our signature scheme (Figure 1) in this section.

The scheme depends on parameters $n, m, k, \kappa, w, q, \alpha, d, B$ and distributions D_E, D_S, D_y and D_z. The distributions D_S and D_E are the distributions for the secret and error respectively in the LWE assumption. As in [10, 18], the distribution $D^n_{y,\mathbf{Sc}}(\mathbf{z})$ is the distribution coming from the shift of the distribution D^n_y by an offset vector $\mathbf{Sc}$. Various constraints on the parameters will be given later, but we typically have $m > n = k$ and $q > 2^d \geq B$. The main security parameters are n (the security of our scheme will depend on (n, q, α)-LWE) and κ (which controls the probability of breaking the hash function).

The scheme requires a hash function H to binary strings of fixed length κ, and an encoding function F that maps binary strings of length κ to elements of the set $\mathcal{B}_{k,w}$ of length k vectors of weight w with coefficients in $\{-1, 0, 1\}$. We require F to be close to an injection in the sense that

$$\Pr_{s_1,s_2 \leftarrow \{0,1\}^\kappa}(F(s_1) = F(s_2)) \leq \frac{c_1}{2^\kappa} \tag{2}$$

for some constant c_1. We also typically choose parameters so that $2^\kappa \approx \#\mathcal{B}_{k,w} = 2^w\binom{k}{w}$. There are several ways to construct a suitable function F. One method is given in Section 4.4 of [10] and some other approaches are discussed in Appendix C (and the references) of Biswas and Sendrier [5].

The verifier wants to test that signature vectors $\mathbf{z}$ have come from the correct distribution D^n_z. This could be done in many ways, depending on D_z and how much statistical analysis the verifier wishes to perform. If D_z is a uniform distribution on $[-B, B]$ then the natural test is that $\|\mathbf{z}\|_\infty \leq B$; this could be entirely implicit if the interval is of the form $[-2^{a-1}+1, 2^{a-1}]$ and entries of $\mathbf{z}$ are represented by a bits. If D_z is a Gaussian or Gaussian-like distribution with mean 0 and standard deviation σ_z then a cheap test is to have a bound B (e.g., $B = 2\sqrt{n}\sigma_z$; see Lemma 4.4 of the full version of [18]) such that $\mathbf{z} \leftarrow D^n_z$ implies $\|\mathbf{z}\|_2 \leq B$ with high probability. Hence, in Line 4 of Algorithm 3 we write this as $\|\mathbf{z}\|_\ell \leq B$ where typically $\ell \in \{2, \infty\}$.

The scheme is given by Algorithms 1, 2 and 3 in Figure 1. While reading the protocol the reader may keep in mind the following set of parameters:

$$(n, m, k, \kappa, q, d, B) = (512, 945, 512, 132, \approx 2^{30.84}, 24, \approx 2^{20.97}).$$

The distributions $D_E = D_S$ used here are discrete Gaussians with standard deviation $\sigma_E = \sigma_S \geq 2\sqrt{n}$. We will choose the distributions D_y and D_z to be uniform distributions, such as $[-B, B]$. A minor subtlety is that D_y must cover D_z with a little slack on each side, so to keep the notation simple we choose D_y to be the uniform distribution on $[-B, B]$ and D_z to be the uniform distribution on $[-(B-U), B-U]$ where $U = 14\sqrt{w}\sigma_E \geq 28\sqrt{wn}$. If one wanted to be pedantic would one could modify line 4 of Algorithm 3 to $\|\mathbf{z}\|_\infty \leq B - U$.

The message is denoted μ. Recall that, for $a \in \mathbb{Z}$, $\lfloor a \rceil_d = (a - [a]_{2^d})/2^d$ is essentially the integer a with its d least significant bits removed. The value M used in the rejection sampling in Line 10 of Algorithm 2 is a bound for the expected number of trials until rejection sampling succeeds, as in Lemma 2. The rejection in Line 4 of Algorithm 1 occurs with probability less than $1/30$ for our parameters, and since LWE with bounded number of samples is not easier than general LWE (see Section 2.2), it follows that the outputs of the key generation algorithm are hard LWE instances.

Algorithm 1 Key generation

INPUT: $n, m, k, q, \sigma_S, \sigma_E$
OUTPUT: $\mathbf{A}, \mathbf{T}$
1: $\mathbf{A} \leftarrow \mathbb{Z}_q^{m \times n}$
2: $\mathbf{S} \leftarrow D_S^{n \times k}$
3: $\mathbf{E} \leftarrow D_E^{m \times k}$
4: **if** $|\mathbf{E}_{i,j}| > 7\sigma_E$ for any (i, j) **then**
5: Restart
6: **end if**
7: $\mathbf{T} \equiv \mathbf{AS} + \mathbf{E} \pmod q$
8: **return** $\mathbf{A}, \mathbf{T}$

Algorithm 2 Signing

INPUT:
$\mu, \mathbf{A}, \mathbf{T}, \mathbf{S}, D_y, D_z, d, w, \sigma_E, H, F, M$
OUTPUT: $(\mathbf{z}, c)$
1: $\mathbf{y} \leftarrow D_y^n$
2: $\mathbf{v} \equiv \mathbf{Ay} \pmod q$
3: $c = H(\lfloor \mathbf{v} \rceil_d, \mu)$
4: $\mathbf{c} = F(c)$
5: $\mathbf{z} = \mathbf{y} + \mathbf{Sc}$
6: $\mathbf{w} \equiv \mathbf{Az} - \mathbf{Tc} \pmod q$
7: **if** $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - 7w\sigma_E$ **then**
8: Restart
9: **end if**
10: **return** $(\mathbf{z}, c)$ with probability $\min\left(D_z^n(\mathbf{z})/(M \cdot D_{y,\mathbf{Sc}}^n(\mathbf{z})), 1\right)$

Algorithm 3 Verifying

INPUT: $\mu, \mathbf{z}, c, \mathbf{A}, \mathbf{T}, \ell, B, d, H, F$
OUTPUT: Accept or Reject
1: $\mathbf{c} = F(c)$
2: $\mathbf{w} \equiv \mathbf{Az} - \mathbf{Tc} \pmod q$
3: $c' = H(\lfloor \mathbf{w} \rceil_d, \mu)$
4: **if** $c' = c$ and $\|\mathbf{z}\|_\ell \leq B$ **then**
5: **return** "Accept"
6: **else**
7: **return** "Reject"
8: **end if**

Fig. 1: The LWE Signature Scheme

The test in Line 7 of Algorithm 2 ensures that $\lfloor \mathbf{v} \rceil_d = \lfloor \mathbf{v} - \mathbf{Ec} \rceil_d = \lfloor \mathbf{w} \rceil_d$, and so the signatures do verify. The bound $7w\sigma_E$ comes from the fact that entries of $\mathbf{E}$ are bounded by $7\sigma_E$ and that the weight of $\mathbf{c}$ is w. Assuming that $\mathbf{w}$ is distributed close to uniformly, then this condition will hold with probability $(1 - 14w\sigma_E/2^d)^m$ and so we require

$$2^d \gtrapprox 7wm\sigma_E. \tag{3}$$

The probability of acceptance is targeted between $1/3$ and $1/2$ for our parameters (see Table 1 for details).

Remark 1. The signature size essentially depends on n and the distribution D_z. Due to the rejection sampling, the distribution D_z depends on the size of $\mathbf{Sc}$, which depends on D_S (i.e., σ_S) and the weight w of $\mathbf{c}$. Hence, the signature size is driven by n, w and D_S. A surprising fact is that the signature size does not depend on m or d. In fact, it seems to be quite possible to choose 2^d rather

large and q quite a bit larger than 2^d (as a minimum we need $\lfloor \mathbf{Ay} \pmod q \rceil_d$ to provide more than κ-bits of entropy into the hash function.

3.1 Possible Attacks

We give a rigorous security proof in the random oracle model in Section 4, but to give some intuition into the design of the scheme and the idea of the proof we informally sketch some potential attacks. Note that the LWE problem appears in different ways in both attacks 1 and 4, and the SIS problem appears in attack 5. These issues are reflected in our security proofs. The modification of Lyubashevsky's proof from [18] shows how to use a forgery to solve SIS, while our new proof deals with attack 5 by choosing parameters so that the SIS instance has no solutions with overwhelming probability.

1. One can try to determine $\mathbf{S}$ from $(\mathbf{A}, \mathbf{T})$. This is n semi-independent instances of LWE.
2. A general attack on any such 3-move protocol is to guess the hash value $\mathbf{c}$. Precisely: the attacker chooses $\mathbf{z}$ and $\mathbf{c}$ and computes $c = H(\lfloor \mathbf{Az} - \mathbf{Tc} \pmod q \rceil_d, \mu)$ and hopes that $F(c) = \mathbf{c}$. This attack is prevented by ensuring that the hash function has large enough output set (i.e., κ is sufficiently large) and that F is close to a bijection.
3. A similar standard attack is to request a signature $(\mathbf{z}, c)$ on a message μ and then to try to determine a second message μ' such that

$$H(\lfloor \mathbf{Az} - \mathbf{T}F(c) \rceil_d, \mu') = H(\lfloor \mathbf{Az} - \mathbf{T}F(c) \rceil_d, \mu).$$

   In other words, we have to compute a second-preimage for the hash function. Again this forgery is prevented by taking κ large enough.
4. Another natural attack is to choose a random vector $\mathbf{v}$ and compute $\mathbf{c} = F(H(\lfloor \mathbf{v} \rceil_d, \mu))$. Then one has to find a short vector $\mathbf{z}$ such that

$$\mathbf{Az} - \mathbf{Tc} \equiv \mathbf{v} + \mathbf{e} \pmod q$$

   where $\mathbf{e}$ is a vector so that $\lfloor \mathbf{v} + \mathbf{e} \rceil_d = \lfloor \mathbf{v} \rceil_d$ (in particular, the entries of $\mathbf{e}$ are small). We re-write this equation as

$$\mathbf{Az} - \mathbf{e} \equiv (\mathbf{v} + \mathbf{Tc}) \pmod q$$

   where the right hand side is known and we want to find $(\mathbf{z}, \mathbf{e})$ subject to $\|\mathbf{z}\|_\ell \leq B$ and $\|\mathbf{e}\|_\infty \leq 2^{d-1}$. This is a variant of LWE.
5. Suppose one can find a short (yet non-zero) vector $\mathbf{x}$ such that $\lfloor \mathbf{Ax} \rceil_d = 0$. If $(\mathbf{z}, c)$ is a signature on message μ then, with high probability, so is $(\mathbf{z} + \mathbf{x}, c)$. This would lead to a successful forgery under our adaptive security definition. Finding such an $\mathbf{x}$ is essentially solving SIS: we seek a short vector $\left(\begin{smallmatrix}\mathbf{x}\\ \mathbf{e}\end{smallmatrix}\right)$ such that $(\mathbf{A}|\mathbf{I})\left(\begin{smallmatrix}\mathbf{x}\\ \mathbf{e}\end{smallmatrix}\right) \equiv 0 \pmod q$. This attack can be prevented either by choosing the parameters so that no solution exists (for example, using Lemma 7) or by making a specific computational assumption on the hardness of SIS for our parameters.

6. One can try to perform a statistical analysis of the values $(\mathbf{z}, c)$ to learn something about $\mathbf{S}$ or $\mathbf{E}$. For example, one could try to "average" the $\mathbf{z}$ to learn about $\mathbf{S}$, or use the fact that $\lfloor \mathbf{w} \rceil_d = \lfloor \mathbf{w} + \mathbf{Ec} \rceil_d$ to learn something about $\mathbf{E}$. All such attacks are prevented by ensuring that the distribution of values $(\mathbf{z}, c)$ is independent of the secret values.
   Note that we are not stating that the output distribution on pairs $(\mathbf{z}, c)$ from the signing algorithm is equal to $D_z^n \times U_\kappa$, where U_κ denotes the uniform distribution on κ-bit strings. Indeed, there could be a bias coming from the fact that certain values for c do not arise for certain values for $\mathbf{z}$ (e.g., due to the check in Line 7 of the signing algorithm). What we are claiming is that any bias in the distribution depends only on public information and not on the private key.

4 Security proofs

There are several ways to prove security of our signature scheme in the random oracle model. Each requires different conditions on the parameters. Theorem 2 follows Lyubashevsky's blueprint and seems to be the most useful for short signatures.

Theorem 2. *Let q be prime. Let parameters n, m, d, κ, B be such that*

$$(2B)^n q^{m-n} \geq (2^{d+1})^m 2^\kappa. \tag{4}$$

and suppose equation (2) holds. Let $D_y = [-B, B]$ with the uniform distribution and let $\boldsymbol{S}, \boldsymbol{E}$ have entries chosen from discrete Gaussian distributions with standard deviation $\sigma_S = \sigma_E = \alpha q$. Let A be a forger against the signature scheme in the random oracle model that makes h hash queries, s sign queries, runs in time t and succeeds with probability δ. Then there is a negligible ε and some $0 \leq \delta' \leq \delta$ such that A can be turned into either of the following two algorithms:

1. *an algorithm, running in time approximately t and with advantage $\delta - \delta' - \varepsilon$, that solves the (n, m, q, α)-decisional-LWE problem.*
2. *an algorithm, running in time approximately $2t$ and with success probability $\delta' \left(\frac{\delta'}{h} - \frac{1}{2^\kappa} \right)$, that solves the unbalanced $(m + n, m, q)$-search-SIS problem: Given an $m \times (n+m)$ matrix $\boldsymbol{A}'$ to find a length n vector $\boldsymbol{y}_1$ and a length m vector $\boldsymbol{y}_2$ such that $\|\boldsymbol{y}_1\|_\infty, \|\boldsymbol{y}_2\|_\infty \leq \max(2B, 2^{d-1}) + 2E'w$ and $\boldsymbol{A}'(\frac{\boldsymbol{y}_1}{\boldsymbol{y}_2}) \equiv 0 \pmod q$ where E' satisfies*

$$(2E')^{m+n} \geq q^m 2^\kappa. \tag{5}$$

Theorem 3. *Let q be prime. Let the parameters be chosen such that $B, 2^d \geq 14\alpha q$ and*

$$q^m \geq (4B + 1)^n (2^{d+1})^m 2^\kappa. \tag{6}$$

Suppose equation (2) holds. Let $D_y = [-B, B]$ with the uniform distribution and let $\boldsymbol{S}, \boldsymbol{E}$ have entries chosen from discrete Gaussian distributions with standard

deviation $\sigma_S = \sigma_E = \alpha q$. Let A be a forger against the signature scheme in the random oracle model that makes h hash queries, s sign queries, runs in time t and succeeds with probability δ. Then A can be turned into an algorithm that solves (n, m, q, α)-decisional-LWE, running in time approximately $2t$, and with success probability at least

$$\min_{0<\delta'<\delta} \max\left\{|\delta - \delta'|, \frac{\delta'}{h}\left(\frac{\delta'}{h} - \frac{1}{2^\kappa}\right) + O\left(\frac{s(s+h)}{2^\kappa} + \frac{m+n}{2^{140}}\right)\right\}.$$

The proof of Theorem 2 is given in Subsection 4.2. We sketch the main idea here. We first replace the signing oracle with a simulation in the random oracle model. We then replace the public key $(\mathbf{A}, \mathbf{T})$ with a different value; the decisional-LWE assumption appears at this point. The forking lemma is then used to transform a forger into an algorithm that solves SIS.

Here, we first show that there is enough entropy going into the hash function. In Algorithm 2 the vector $\mathbf{y}$ is sampled from D_y^n, and when D_y is the uniform distribution on $[-B, B]$ this means there are $(2B+1)^n$ choices for $\mathbf{y}$. There are at most $(q/2^d)^m$ choices for $\lfloor \mathbf{Ay} \rceil_d$, and these values are hashed to κ-bit strings, giving at most 2^κ possible values for c. It is necessary that the hash outputs are uniformly distributed, which requires that there is sufficient entropy in the distribution of values $\lfloor \mathbf{Ay} \rceil_d$ being hashed. Since $(2B+1)^n$ will be much greater than 2^κ (this condition is required for the computational assumptions to be reasonable), it suffices to ensure that there is a sufficiently large supply of possible values for $\lfloor \mathbf{Ay} \rceil_d$. This is the content of Lemma 3.

Lemma 3. *Let $q > 4B > 4$ and $m > n > \kappa$ and other notation be as above. Let D_y be the uniform distribution on $[-B, B]$ and suppose the condition in Equation (4) holds. Then the number of values for $\lfloor \boldsymbol{Ay} \pmod q \rceil_d$ is at least 2^κ, and the probability that two values $\boldsymbol{y}_1, \boldsymbol{y}_2$ sampled uniformly from $[-B, B]^n$ give the same value is at most $1/2^\kappa$.*

Proof. Let $\mathbf{A}$ be a randomly chosen matrix. We can assume the rank of $\mathbf{A}$ is n provided that $m \geq n$ (if not then we can re-generate $\mathbf{A}$ in the key generation). Hence $\mathbf{A}$ defines an injective linear map from $\mathbb{Z}^n$ to $\mathbb{Z}^m$.

Let $\mathbf{y}_1 \in D_y^n$ and set $u = \lfloor \mathbf{Ay}_1 \pmod q \rceil_d$. Define

$$S_u = \{\mathbf{y}_2 \in D_y^n : \lfloor \mathbf{Ay}_2 \pmod q \rceil_d = u\}.$$

It suffices to bound $\#S_u$. Note that if $\mathbf{y}_2 \in S_u$ then $\mathbf{y} = \mathbf{y}_1 - \mathbf{y}_2$ satisfies $\|\mathbf{y}\|_\infty \leq 2B$ and

$$\mathbf{Ay} \pmod q \in [-2^d, 2^d]^m.$$

Hence, to bound $\#S_u$ it suffices to bound the number of such vectors $\mathbf{y}$.

A randomly chosen matrix $\mathbf{A}$ defines a random lattice $L = \{\mathbf{v} \in \mathbb{Z}^m : \mathbf{v} \equiv \mathbf{Ay} \pmod q \text{ for some } \mathbf{y} \in \mathbb{Z}^n\}$. The volume of L is q^{m-n}. By the Gaussian heuristic, the number of elements in $L \cap [-2^d, 2^d]^m$ is expected to be $2^{(d+1)m}/q^{m-n}$. Finally, suppose $\mathbf{y}, \mathbf{y}' \in [-2B, 2B]^n$ are such that $\mathbf{Ay} \equiv \mathbf{Ay}' \pmod q$. Then $\mathbf{A}(\mathbf{y} - \mathbf{y}') \equiv 0 \pmod q$, which implies $\mathbf{y} \equiv \mathbf{y}' \pmod q$ which, due to the size constraints

and the condition $q > 4B$, implies $\mathbf{y} = \mathbf{y}'$. Hence, $\#S_u$ is upper bounded by $2^{(d+1)m}/q^{m-n}$ for all u.

There are $(2B+1)^n$ choices for $\mathbf{y}_1$, so if we choose two of uniformly, the probability of a collision is bounded by

$$\frac{2^{(d+1)m}/q^{m-n}}{(2B+1)^n} \leq \frac{1}{2^\kappa}. \tag{7}$$

□

4.1 Simulation in the random oracle model

Let A be a forger for the signature scheme. The forger takes as input a public key for the signature scheme, makes h random oracle queries and s sign queries, runs in time t, and outputs a valid signature with probability δ. Note that sign queries contain implicit hash queries, but we count those separately. So the total number of calls to the random oracle is actually $s+h$. We want to use A to solve LWE or SIS.

Game 0 is running the forger A on the real cryptosystem. Game 1 is the same as Game 0, except that the sign queries are replaced by a simulation in the random oracle model (see Algorithm 4 below) and hash queries are handled by answering with random values (as usual we use a list to ensure that the hash function responses are consistent). Our goal in this section is to show that Game 0 and Game 1 are indistinguishable.

Algorithm 4 Game 1 sign query handler.

INPUT: $\mu, \mathbf{A}, \mathbf{T}, D_y, D_z, d, w, \sigma_E, H, F, M$
OUTPUT: $(\mathbf{z}, c)$

choose uniformly a κ-bit binary string c
$\mathbf{c} = F(c)$
$\mathbf{z} \leftarrow D_z^n$
$\mathbf{w} \equiv \mathbf{Az} - \mathbf{Tc} \pmod q$
if $|[\mathbf{w}_i]_{2^d}| > 2^{d-1} - 7w\sigma_E$ **then**
    Restart
end if
if H has already been defined on $(\lfloor \mathbf{w} \rceil_d, \mu)$ **then**
    Abort game
else
    Program $H(\lfloor \mathbf{w} \rceil_d, \mu) = c$
end if
return $(\mathbf{z}, c)$ with probability $1/M$

Lemma 4. *Let notation be as above and suppose the conditions of Lemma 3 hold. Then Game 0 and Game 1 are indistinguishable.*

Proof. As with Lemma 5.3 of [18] the indistinguishability can be shown in several steps. We sketch the main ideas.

The first step is to show that, in the random oracle model, one can consider $\mathbf{c}$ as being independent of $\mathbf{y}$. We decouple $\mathbf{c}$ from $\mathbf{y}$ and show that the changes (Lines 1-2, 8-9) are statistically negligible. First, by Lemma 3 the distribution of values $\lfloor \mathbf{Ay} \pmod q \rceil_d$ has sufficient entropy that c is uniformly distributed on κ-bit strings. Hence the real signing algorithm is consistent with line 1 of the simulation.

Lemma 3 can be used to show that the values $\lfloor \mathbf{w} \rceil_d$ are well-distributed. Hence, the probability that the game aborts in line 9 of Algorithm 4 is negligible (the danger is that two values of $\lfloor \mathbf{w} \rceil_d$ might arise from different choices of c, and this cannot happen in Algorithm 2). This follows by an argument similar to that in [18]. The probability is bounded by $s(s+h)\max\left((2^{d+1}/q)^m, 2^{-\kappa}\right)$ using a hybrid argument (this term contributes to the ε in the statement of Theorem 2).

The next step of the proof is to note that the output distributions have negligible statistical difference, due to the rejection sampling (*cf.* Lemma 2) in two places in the sign algorithm. Hence, the success of any distinguisher between these two games is negligible. □

4.2 Completing the proof of Theorem 2

We want to show that a forger A can be used to solve SIS. We could apply the forking lemma to Game 1, showing that an adversary who can win Game 1 can be used to solve search-SIS. This approach is analogous to Lemma 5.4 of Lyubashevsky [18]. The argument requires there to be more than one private key for the given public key $(\mathbf{A}, \mathbf{T})$. Precisely, we need there to exist at least two pairs $(\mathbf{S}, \mathbf{E}), (\mathbf{S}', \mathbf{E}')$ such that $\mathbf{T} \equiv \mathbf{AS} + \mathbf{E} \equiv \mathbf{AS}' + \mathbf{E}' \pmod q$ and where both pairs are roughly equally likely with respect to the output distribution of the key generation algorithm. This is achieved in Lemma 5.2 of [18] in the case of SIS by taking m to be sufficiently large. This approach would require taking n large and the signature size is increased.

Instead we employ an alternative proof technique given in Section 6 of Lyubashevsky [18]. The idea is to introduce Game 2, which is Game 1 but with the public key replaced by a pair $(\mathbf{A}, \mathbf{T} \equiv \mathbf{AS}' + \mathbf{E}' \pmod q)$ of matrices over $\mathbb{Z}_q$, where $\mathbf{S}'$ and $\mathbf{E}'$ have larger entries (chosen uniformly in $[-E', E']$ where E' is as in equation (5)) than $\mathbf{S}$ and $\mathbf{E}$ do. The decisional-LWE assumption is that Game 1 and Game 2 are computationally indistinguishable: the only change happens in the public keys which the adversary can not distinguish, according to Lemma 5. Note that we do not claim that $(\mathbf{A}, \mathbf{T})$ are uniformly distributed.

Lemma 5. *If the decisional $(n, m, q, \sigma_S, \sigma_E)$-LWE assumption and decisional (n, m, q, E')-assumption holds then Game 1 and Game 2 are hard to distinguish.*

Proof. (Sketch) Let D_0 be the distribution of pairs $(\mathbf{A}, \mathbf{T})$ output by the key generation in Game 1, so that $\mathbf{T} \equiv \mathbf{AS} + \mathbf{E} \pmod q$ where $\mathbf{S}$ and $\mathbf{E}$ have entries chosen from discrete Gaussian distributions with parameter σ_S and σ_E respectively. Let D_1 be the distribution of pairs $(\mathbf{A}, \mathbf{T})$ output by the key generation

in Game 2. Namely, $\mathbf{T} \equiv \mathbf{AS}' + \mathbf{E}' \pmod q$ where $\mathbf{S}'$ is an $n \times k$ matrix with entries in $[-E', E']$ and $\mathbf{E}'$ is an $m \times k$ matrix with entries in $[-E', E']$. The inequality in Equation (5) implies that an LWE instance from D_1 has non-unique solutions with overwhelming probability. Let U be the uniform distribution on pairs $(\mathbf{A}, \mathbf{T})$. The assumptions in the proof are that it is hard to distinguish U from D_0, and that it is hard to distinguish U from D_1.

If the forger A behaves differently between Games 1 and 2, then it acts as a distinguisher between D_0 and D_1. So, for simplicity, write B for any distinguisher that takes inputs $(\mathbf{A}, \mathbf{T})$ from D_0 or D_1 and outputs a bit b. The algorithm wins if the samples were taken from D_b. The advantage of B is $|\Pr(B \text{ wins}) - \frac{1}{2}|$. Write b_{D_0} for the expected output of B when run on D_0, and b_{D_1} for the expected output of B when run on D_1. We have $|\, b_{D_0} - b_{D_1}| = \delta$ where δ is the difference in the behaviour of A between the two games.

Now, suppose we run B on samples from U. Then B also outputs $b \in \{0, 1\}$ with some average behaviour b_U. It is then easy to see that either $|b_U - b_{D_0}| \geq \delta/2$ or $|b_U - b_{D_1}| \geq \delta/2$. It follows that B distinguishes either D_0 from U or D_1 from U, which is a contradiction.

We write δ' for the success probability of the forger when running Game 2. If the decisional-LWE assumption holds then $\delta - \delta'$ is negligible. Finally, we apply the forking lemma to Game 2; this is the content of Lemma 6. Theorem 2 follows from Lemma 6.

Lemma 6. *Suppose the forger A plays Game 2, makes h hash function queries and s sign queries, runs in time t, and succeeds with probability δ'. Suppose the parameters satisfy the conditions in Theorem 2. Then there exists an algorithm running in time approx $2t$ and with success probability $\frac{\delta'}{h}\left(\frac{\delta'}{h} - \frac{1}{2^k}\right) + O(\frac{s^2}{2^\kappa} + \frac{n+m}{2^{140}})$ that solves the unbalanced search-SIS problem defined in Theorem 2.*

Proof. Let $\mathbf{A}'$ be the $m \times (n + m)$ matrix giving the input SIS instance. Taking the Hermite normal form we can write $\mathbf{A}' = (\mathbf{A}|\mathbf{I}_m)$, where $\mathbf{A}$ is an $m \times n$ matrix. The goal of the proof is to compute short non-zero vectors $\mathbf{y}_1, \mathbf{y}_2$ such that $\mathbf{A}\mathbf{y}_1 + \mathbf{y}_2 \equiv 0 \pmod q$.

As mentioned, we choose a random $n \times k$ matrix $\mathbf{S}'$ with entries in $[-E', E']$ and a random $m \times k$ matrix $\mathbf{E}'$ with entries in $[-E', E']$. Set $\mathbf{T} \equiv \mathbf{AS}' + \mathbf{E}'$ $(\text{mod } q)$. The inequality in Equation (5) implies that the LWE instance has non-unique solutions with overwhelming probability. Game 2 is to run the forger A on $(\mathbf{A}, \mathbf{T})$.

The forger makes hash and sign queries that are simulated in the random oracle model as usual. Eventually A outputs a valid signature $(\mathbf{z}, c)$ on message μ. We know that the random oracle has been queried in order for the verification equation $c = H(\lfloor \mathbf{w} \rceil_d, \mu)$ to hold for $\mathbf{w} = \mathbf{Az} - \mathbf{Tc} \pmod q$.

We will now reduce to the case where c arises from a hash query, rather than a sign query. Suppose not: then there is a sign query on a message μ' with output equal to $(\mathbf{z}', c)$, and so

$$c = H(\lfloor \mathbf{Az} - \mathbf{Tc} \pmod q \rceil_d, \mu) = H(\lfloor \mathbf{Az}' - \mathbf{Tc} \pmod q \rceil_d, \mu').$$

If $\mu' \neq \mu$ or $\lfloor \mathbf{Az} - \mathbf{Tc} \pmod q \rceil_d \neq \lfloor \mathbf{Az}' - \mathbf{Tc} \pmod q \rceil_d$ then we have a collision in H, so this event occurs with probability $1/2^\kappa$. Therefore we may assume that $\mu' = \mu$ and that $\mathbf{A}(\mathbf{z} - \mathbf{z}') \pmod q$ has entries in $[-2^d, 2^d]$. If $\mathbf{z} \neq \mathbf{z}'$ then we have a non-zero solution to $\mathbf{Ay}_1 + \mathbf{y}_2 \equiv 0 \pmod q$ with $\|\mathbf{y}_1\|_\infty \leq 2B$ and $\|\mathbf{y}_2\|_\infty \leq 2^d$ and we have solved the SIS instance and we are done. Finally, if $\mathbf{z} = \mathbf{z}'$ then $(\mu', \mathbf{z}', c)$ is equal to $(\mu, \mathbf{z}, c)$ and so it is not a forgery. Hence, for the remainder of the proof we may assume that the forgery $(\mathbf{z}, c)$ has c an output of a random oracle query (on some index I) that was not made as part of a sign query.

Now we apply the Bellare-Neven [4] version of the forking lemma. In other words, we re-wind the attack, so that $\mathbf{v}$ is the same but the I-th random oracle output is taken to be a different binary string c'. One can verify that our signature scheme is a generic signature scheme with security parameter κ. With probability $\delta'(\frac{\delta'}{h} - \frac{1}{2^\kappa})$ we obtain a valid signature $(\mathbf{z}', c')$ on the same message μ. Let $\mathbf{c} = F(c)$ and $\mathbf{c}' = F(c')$. With overwhelming probability we have $\mathbf{c} \neq \mathbf{c}'$.

Now, we have $\lfloor \mathbf{Az} - \mathbf{Tc} \pmod q \rceil_d = \lfloor \mathbf{Az}' - \mathbf{Tc}' \pmod q \rceil_d$ and so $\mathbf{Az} - \mathbf{Tc} + \mathbf{e} \equiv \mathbf{Az}' - \mathbf{Tc}' \pmod q$ for some vector $\mathbf{e}$ satisfying $\|\mathbf{e}\|_\infty \leq 2^{d-1}$ coming from the rounding. Hence, putting $\mathbf{T} \equiv \mathbf{AS}' + \mathbf{E}' \pmod q$, we see

$$\mathbf{A}(\mathbf{z} - \mathbf{z}' + \mathbf{S}'(\mathbf{c}' - \mathbf{c})) + \mathbf{e} + \mathbf{E}'(\mathbf{c}' - \mathbf{c}) \equiv 0 \pmod q. \tag{8}$$

Writing $\mathbf{y}_1 = \mathbf{z} - \mathbf{z}' + \mathbf{S}'(\mathbf{c}' - \mathbf{c})$ and $\mathbf{y}_2 = \mathbf{e} + \mathbf{E}'(\mathbf{c}' - \mathbf{c})$ we have $\|\mathbf{y}_1\|_\infty \leq 2B + 2E'w$ and $\|\mathbf{y}_2\|_\infty \leq 2^{d-1} + 2E'w$. Hence, as long as $(\mathbf{y}_1, \mathbf{y}_2) \neq (0, 0)$, we have a solution to the input SIS instance. Finally, since the matrices $(\mathbf{S}', \mathbf{E}')$ are not uniquely defined with high probability, the adversary does not know which pair $(\mathbf{S}', \mathbf{E}')$ is being used to construct the vectors $\mathbf{y}_1$ and $\mathbf{y}_2$. Hence, with probability at least $\frac{1}{2}$, we deduce that $(\mathbf{y}_1, \mathbf{y}_2) \neq (0, 0)$. □

5 Parameter selection

In Table 1, we give some concrete parameters for our signature scheme in Figure 1. The parameters are provably secure (*cf.* Theorem 2) and reduce to worst-case computational problems in general lattices. In Appendix B.2, we also give some shorter signatures that are based on non-standard LWE assumptions.

We discuss how the signature parameters in Table 1 are chosen. Given the security parameter n, the weight w is chosen such that $1/(2^w \cdot \binom{n}{w}) < 2^{-\kappa}$. Standard estimates show that $w \approx \kappa / \log(n)$. The standard deviation of Gaussian entries $\mathbf{S}$ and $\mathbf{E}$ are chosen such that the LWE problem for the key is secure. In the signature, one computes $\mathbf{z} = \mathbf{y} + \mathbf{Sc}$. By the central limit theorem the entries of $\mathbf{Sc}$ are Gaussian with mean 0 and standard deviation $\sigma_{\mathbf{Sc}} = \sqrt{w}\sigma_{\mathbf{S}}$. We bound the entries of $\mathbf{Sc}$ by $14\sigma_{\mathbf{Sc}}$, which is true with probability 2^{-140}. Let D_z and D_y be the uniform distribution on $[-B + U, B - U]$ and $[-B, B]$ respectively. The probability of acceptance in line 7 of Algorithm 2 is $(1 - 14\sigma_{\mathbf{E}} w/2^d)^m$; hence we need $q > 2^d \geq 14m\sigma_{\mathbf{E}} w$. We also need the parameters to satisfy the conditions in the statement of Theorem 2. The signature size is given by $n\lceil \log_2(2(B - U)) \rceil + \kappa \approx \lceil \log_2(2B) \rceil + \kappa$. Note that the public key size can

be effectively halved by generating $\mathbf{A}$ using a pseudo-random generator and by publishing only the seed for the generator as part of the public key (*cf.* [12]).

Table 1: Parameters for LWE Signatures using Uniform Distributions.

		I	**II**	**III**	**IV**	**V**
n		640	576	512	512	400
m		1137	969	945	1014	790
w	$2^w \cdot \binom{n}{w} \geq 2^{128}$	18	18	19	19	20
Approx. $\log_2(q)$		34.34	33.10	30.84	32.66	28.71
κ		132	132	132	132	132
$\sigma_{\mathbf{E}}$		58	68	66	224	70
$\sigma_{\mathbf{S}}$		58	68	66	224	70
$\sigma_{\mathbf{Sc}}$	$\sqrt{w}\sigma_{\mathbf{S}}$	246.07	288.50	287.69	976.39	313.05
B	$14\sigma_{\mathbf{Sc}}(n-1)$	2201370	2322422	2058115	6985118	1748695
2^d		2^{24}	2^{24}	2^{24}	2^{26}	2^{24}
Prob. acceptance in line 7 of Alg 2.	$\left(1-14\sigma_{\mathbf{E}}w/2^d\right)^m$	0.371	0.371	0.372	0.406	0.397
Hermite factor	(for breaking the key)	1.0056	1.0057	1.0057	1.0055	1.0064
Hermite factor	(for forging signature)	1.0038	1.0044	1.0048	1.0047	1.0061
Signature (bits)	$n\lceil\log_2(2B)\rceil+\kappa$	14852	13380	11396	12420	8932
Public key (Mb)	$2mn\log_2(q)$	6.0	4.4	3.6	4.0	2.2
Signing key (Mb)	$2mn\log_2(4\sigma_S)$	1.4	1.0	0.9	1.2	0.6

To evaluate the security of our parameters against practical lattice attacks we consider the LWE problem for the secret key and the SIS problem for the forgery. The security can be estimated by computing the (root) Hermite factor γ of the lattices (based on the BKZ 2.0 estimates of Chen and Nguyen [9]). Tables 2 and 3 of [9] suggest that instances with $\gamma \leq 1.0065$ should require around 2^{128} operations to solve using BKZ lattice reduction. These security estimations are standard in the field so we only sketch the details.

Solving an LWE instance $(\mathbf{A}, \mathbf{b} \equiv \mathbf{As}+\mathbf{e} \pmod q)$ corresponds to solving the closest vector problem (CVP) with target $\mathbf{b}$ in the image lattice $\{\mathbf{v} \in \mathbb{Z}^m : \mathbf{v} \equiv \mathbf{As} \pmod q\} \subseteq \mathbb{Z}^m$. It is known that the optimal dimension m, when using a lattice reduction algorithm with root Hermite factor δ, is around $\sqrt{n\log(q)/\log(\delta)}$. To get closer to the optimal dimension (which is often larger than m) one can consider the inhomogeneous SIS (ISIS) problem $\mathbf{b} = (\mathbf{A}|\mathbf{I}_m)(\mathbf{s}^T, \mathbf{e}^T)^T \pmod q$. Let $\mathbf{v}' \in \mathbb{Z}^{n+m}$ be any solution (not necessarily small) to the equation $\mathbf{b} = (\mathbf{A}|\mathbf{I}_m)\mathbf{v}'$ (mod q). One can solve the ISIS problem by solving the CVP (with target $\mathbf{v}'$) in the kernel lattice $\{\mathbf{v} \in \mathbb{Z}^{n+m} : \mathbf{b} \equiv (\mathbf{A}|\mathbf{I}_m)\mathbf{v} \pmod q\} \subseteq \mathbb{Z}^m$.

The CVP problem can be solved use the embedding technique. It is plausible to turn the CVP problem into an Unique-SVP problem in the embedded lattice. For instance, using the embedding technique, the above ISIS problem gives $(\mathbf{A}|\mathbf{I}_m|\mathbf{b})(\mathbf{s}^T, \mathbf{e}^T, -1)^T \equiv 0 \pmod q$ and so one can solve the problem by finding a short vector in this lattice. Since the short vector $(\mathbf{s}^T, \mathbf{e}^T, -1)^T$ is often very

small, the standard approach is to estimate the lattice gap $\gamma = \lambda_2(L)/\lambda_1(L)$ (see [17, 1]). We let $\lambda_1(L)$ be the length of the target vector and $\lambda_2(L)$ be the Gaussian expected shortest vector of the q-ary lattice. In the ISIS case, the target vector has norm $\sqrt{n+m+1}\,\sigma_E$ in the case $\sigma_S = \sigma_E$. The root Hermite factor δ (needed for the attack) is $\gamma^{1/(n+m+1)} = \left(\frac{q^{m/(n+m+1)}}{\sigma_E\sqrt{2\pi e}}\right)^{1/(n+m+1)}$.

We also want the SIS problem in the forgery to be hard. In the proof of Lemma 6, we choose random matrices $\mathbf{S}'$ and $\mathbf{E}'$ with entries in $[-E', E']$ for large enough E' such the there exists alternative keys. The short vectors in the forgery (*cf.* Equation (8)) have entries bounded by $\max(2B, 2^{d-1}) + 2E'w$. The short vectors $\mathbf{v}$ in the forging problem $\mathbf{A}\mathbf{v} \equiv 0 \pmod q$ have length $\|\mathbf{v}\|_2 \leq \left(\max(2B, 2^{d-1}) + 2E'w\right)\sqrt{m+n}$. Following Section 3 and equation (1) of [20], an estimate for the length of the shortest vector that we can find is $q^{m/(n+m)}\delta^{m+n}$ (where $\delta \approx 1.0065$), and for the forgery security we need this to be larger than $D = \left(\max(2B, 2^{d-1}) + 2E'w\right)\sqrt{m+n}$. In Table 1 we estimate the Hermite factor required to solve the problem by $\delta = (D/q^{m/(m+n)})^{1/(n+m)}$.

An asymptotically good set of parameters would be to choose κ and n, then set $m = 2n$, $w \approx \kappa/\log(n)$, $\sigma_{\mathbf{E}} = \sigma_{\mathbf{S}} = 2\sqrt{n}$, $B = 14n\sigma_{\mathbf{Sc}} \approx 28n\sqrt{\kappa n/\log(n)}$, $2^d \approx 56n\sqrt{n}\kappa/\log n$, $q \approx 2^{2d}/2B$. The public key and signatures for such parameters are polynomially-sized in the security parameter. By Regev's theorem, the security follows from worst-case computational problems in lattices with polynomial approximation factors.

6 Conclusion

We have described a new method for compressing lattice-based signatures in Lyubashevsky's framework. The new signature scheme, together with the compression method, is based on the standard worst-case hardness of LWE and SIS in general modular lattices. Our signature size for 128-bit security is about 12000 bits, which is shorter than the previous signatures ($\geq$ 16500 bits) whose security are based on hard problems in general lattices.

Acknowledgements

The authors are grateful to Vadim Lyubashevsky, Chris Peikert and anonymous referees for helpful comments and discussions on drafts of this paper. The authors wish to acknowledge NeSI (New Zealand eScience Infrastructure) and the Centre for eResearch at the University of Auckland for providing CPU hours (for searching the parameters in Table 1) and support.

References

1. M . R. Albrecht, R. Fitzpatrick and F. Göpfert, On the Efficacy of Solving LWE by Reduction to Unique-SVP, to appear in Proceedings of International Conference on Information Security and Cryptology 2013.

2. B. Applebaum, D. Cash, C. Peikert and A. Sahai, Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems, in S. Halevi (ed.) CRYPTO 2009, Springer LNCS 5677 (2009) 595–618.
3. S. Bai and S. D. Galbraith, Lattice Decoding Attacks on Binary LWE, IACR Cryptology ePrint Archive 2013: 839 (2013).
4. M. Bellare and G. Neven, Multi-Signatures in the Plain Public-Key Model and a General Forking Lemma, in A. Juels, R. N. Wright and S. De Capitani di Vimercati (eds.), ACM CCS 2006, ACM (2006) 390–399.
5. B. Biswas and N. Sendrier, McEliece Cryptosystem Implementation: Theory and Practice, in J. Buchmann and J. Ding (eds.), PQCrypto 2008, Springer LNCS 5299 (2008) 47–62.
6. F. Böhl, D. Hofheinz, T. Jager, J. Koch, J. H. Seo and C. Striecks, Practical Signatures From Standard Assumptions, in T. Johansson and P. Q. Nguyen (eds.), EUROCRYPT 2013, Springer LNCS 7881 (2013) 461–485.
7. X. Boyen, Lattice Mixing and Vanishing Trapdoors – A Framework for Fully Secure Short Signatures and More, in P. Q. Nguyen and D. Pointcheval (eds.), PKC 2010, Springer LNCS 6056 (2010) 499–517.
8. Z. Brakerski, A. Langlois, C. Peikert, O. Regev and D. Stehlé, Classical Hardness of Learning with Errors, in D. Boneh, T. Roughgarden and J. Feigenbaum (eds.), STOC 2013, ACM (2013) 575–584.
9. Y. Chen and P. Q. Nguyen, BKZ 2.0: Better Lattice Security Estimates, in D. H. Lee and X. Wang (eds.), ASIACRYPT 2011, Springer LNCS 7073 (2011) 1–20.
10. L. Ducas, A. Durmus, T. Lepoint and V. Lyubashevsky, Lattice Signatures and Bimodal Gaussians, in R. Canetti and J. A. Garay (eds.), CRYPTO 2013, Springer LNCS 8042 (2013) 40–56.
11. L. Devroye, Non-Uniform Random Variate Generation, Springer-Verlag, New York, 1986.
12. S. D. Galbraith, Space-efficient variants of cryptosystems based on learning with errors, preprint, 2013.
13. C. Gentry, C. Peikert and V. Vaikuntanathan, Trapdoors for Hard Lattices and New Cryptographic Constructions, in C. Dwork (ed.), STOC 2008, ACM (2008) 197-206.
14. T. Güneysu, V. Lyubashevsky and T. Pöppelmann, Practical Lattice-Based Cryptography: A Signature Scheme for Embedded Systems, in E. Prouff and P. Schaumont (eds.), CHES 2012, Springer LNCS 7428 (2012) 530–547.
15. M. Liu and P. Q. Nguyen, Solving BDD by Enumeration, An Update, in E. Dawson (ed.), CT-RSA 2013, Springer LNCS 7779 (2013) 293–309.
16. V. Lyubashevsky, Fiat-Shamir with Aborts: Applications to Lattice and Factoring-Based Signatures, in M. Matsui (ed.), ASIACRYPT 2009, Springer LNCS 5912 (2009) 598–616.
17. V. Lyubashevsky and D. Micciancio, On Bounded Distance Decoding, Unique Shortest Vectors, and the Minimum Distance Problem, in S. Halevi (ed.), CRYPTO 2009, Springer LNCS 5677 (2009) 577–594.
18. V. Lyubashevsky, Lattice Signatures without Trapdoors, in D. Pointcheval and T. Johansson (eds.), EUROCRYPT 2012, Springer LNCS 7237 (2012) 738–755.
19. D. Micciancio and S. Goldwasser, Complexity of Lattice Problems: A cryptographic Perspective, Kluwer, 2002.
20. D. Micciancio and O. Regev, Lattice-Based Cryptography, in D. J. Bernstein, J. Buchmann, and E. Dahmen (eds.), Post Quantum Cryptography, Springer (2009) 147–191.

21. D. Micciancio and C. Peikert, Hardness of SIS and LWE with Small Parameters, in R. Canetti and J. A. Garay (eds.), CRYPTO 2013, Springer LNCS 8042 (2013) 21–39.
22. D. Pointcheval and J. Stern, Security Arguments for Digital Signatures and Blind Signatures, J. Cryptology, **13** (2000) 361–396.
23. D. Stehlé and R. Steinfeld, Making NTRUEncrypt and NTRUSign as Secure as Standard Worst-Case Problems over Ideal Lattices, Cryptology ePrint Archive: Report 2013/004.
24. O. Regev, On Lattices, Learning with Errors, Random Linear Codes, and Cryptography, in H. N. Gabow and R. Fagin (eds.), STOC 2005, ACM (2005) 84–93.
25. O. Regev, On Lattices, Learning with Errors, Random Linear Codes, and Cryptography, Journal of the ACM 56(6), article 34, 2009.

A Signatures

A signature scheme comprises three randomized algorithms: KeyGen, Sign, Verify. KeyGen takes as input a security parameter and outputs a public/private key pair (pk, sk). Sign takes as input a message μ and a private key sk, and outputs a signature Σ. Verify takes as input a message μ, signature Σ and public key pk, and outputs "valid" or "invalid". We require that, at least with overwhelming probability, Verify(μ, Sign(μ, sk), pk) = "valid".

Adaptive security for signatures is defined using a game between a forgery algorithm F and a challenger. The challenger generates a public key pk for the signature scheme at a given security level and runs the forger. The forger takes as input the public key for the signature scheme, makes h random oracle queries and s sign queries, runs in time t, and outputs (μ, Σ). The forger wins if Verify(μ, Σ, pk) = "valid". The success probability (taken over all public keys generated by the challenger, all responses to the hash and sign queries, and over the random choices made by F) is denoted ϵ. The signature scheme is *secure* if there is no polynomial-time (in terms of the security parameter) algorithm F whose success probability in the above game is non-negligible.

An important tool for analysing signatures in the random oracle model is the Forking Lemma of Pointcheval and Stern [22]. We need the signature scheme to be a generic signature scheme (the scheme in this paper does satisfy that requirement) with security parameter κ and hash output of size 2^κ. Note that a sign query involves an implicit hash query, but that this is to a random value that is chosen by the challenger. Hence, when we say that F makes h hash queries we are only counting the actual queries to the random oracle, and not the additional s hash queries implicit in the signing algorithm.

The basic principle is to run a forger for the signature scheme, interacting with a specific instance of the random oracle, to get a forgery. The forgery corresponds to specific hash value corresponding to the I-th random oracle query. One then replays or rewinds the forger, with the same random tape, and answering the first $I - 1$ queries to the random oracle with the same values as before, but answering the subsequent queries with freshly chosen random values. With a certain probability, the forger outputs a new forgery that corresponds once again to the I-th hash query.

Theorem 13 of [22] considers a forger F that runs in time t, makes h queries to the random oracle (including those performed by the sign oracle), s sign queries, and outputs a forgery with probability $\epsilon > 10(s+1)(s+h)/2^\kappa$. Then the re-winding process produces, with probability one, two valid signatures with the same "y-value" but different hash values in time $t' \leq 120686\, th/\epsilon$.

An alternative formulation was given by Bellare and Neven [4]. The two main differences are a cleaner and more general presentation, and an analysis in the case where the forking lemma just runs F twice (rather than $120686\, th/\epsilon$ times). Bellare and Neven consider a forger F that outputs a valid forgery in time t with probability ϵ making h random oracle queries and s sign queries. Then the rewinding algorithm outputs two valid signatures in time approximately $2t$ and with probability at least (Lemma 1 of [4])

$$\epsilon \left(\frac{\epsilon}{h} - \frac{1}{2^\kappa} \right).$$

In Theorem 3 we use a slight variant of the forking lemma. In our case, we must guess in advance the index I of the hash query that corresponds to a successful forgery (as we need to program this hash value to be a specific element corresponding to the problem instance). Hence, we need to guess the index I from among the h possible values. We then answer the I-th hash query with a specific value c^* and then, in the re-winding, answer the I-th hash query with another specific value $c^\dagger$. Both values c^* and $c^\dagger$ are chosen uniformly at random. Hence, the probability the rewinding algorithm outputs two valid signatures in time approximately $2t$ is

$$\frac{\epsilon}{h} \left(\frac{\epsilon}{h} - \frac{1}{2^\kappa} \right). \tag{9}$$

B Variants

This section discusses some avenues to obtain even shorter signatures. Some of these ideas have already been used by other authors [18, 14, 10]. First we discuss obstructions to very short lattice signatures. The main driver of signature size is that n must be sufficiently large to ensure the lattice problems are hard. Some further issues are:

- The $\mathbf{z}$ vector has to cover all possible values for $\mathbf{Sc}$, so if $\mathbf{Sc}$ can be made smaller then signatures will be smaller. Unfortunately, we cannot just reject those $\mathbf{c}$ for which $\mathbf{Sc}$ is large, since we are unable to simulate such behaviour.
- Lemma 4 requires the simulation to be statistically very close to the real game, and this requires high-grade rejection sampling in the sign algorithm. If we want to have a constant rejection rate then this leads to a linear factor of n in the bound B for the distribution D_z. In principle, using a Gaussian distribution for D_z reduces this to a $\sqrt{n}$ factor, but large constants are introduced that prevent short signatures for concrete small parameters. Overall, the strong requirement of Lemma 4 is a major contributor to the signature size.

B.1 General tricks

One can take k very large so that w is smaller. However, the public key grows in size and the improvement is minor (for example, taking $k = 3n$ so that public keys are 3 times larger only reduces w from 18 to 14 when $n = 600$).

One can apply a further rejection sampling to ensure that $\mathbf{z}$ is small. For example, one can save n bits in the signature by replacing B by $B/2$ in the distribution D_z. In other words, we require that $\mathbf{z} \in [-B/2, B/2]^n$. Since $\mathbf{z}$ is sampled uniformly it follows that the acceptance probability goes from $1/e$ to $1/e^2$. Similarly, saving $2n$ bits by reducing D_z to $[-B/4, B/4]$ changes the acceptance probability to $1/e^4 \approx 1/55$.

B.2 Signatures based on non-standard LWE

We have seen that signature size depends on both n and $\mathbf{Sc}$. Hence, there is a temptation to choose the entries of $\mathbf{S}$ to be as small as possible. For instance, we could choose $\mathbf{S}$ to be a binary matrix (entries uniformly chosen from $\{0, 1\}$) and $\mathbf{E}$ from a discrete Gaussian distribution with standard deviation $\sigma_{\mathbf{E}}$. This is the binary secret LWE problem.

Micciancio and Peikert [21] and Brakerski, Langlois, Peikert, Regev and Stenlé [8] have studied the case of LWE with binary secrets. They give some results that imply that such variants of LWE can be hard. However, their results are not useful for our application as they require a large increase in the parameter n for the LWE problem. More precisely, Theorem 4.6 of [21] shows that (m, n, q)-binary-LWE can be hard as long as SIVP_γ is hard in $k = n/\log(n)$ dimensional lattices, where $\gamma = \tilde{O}(\sqrt{k}q)$. The value $n = 512$ corresponds to $k = 82$, and so such parameters give a weak security guarantee.

Furthermore, there is a lattice decoding attack [3] for LWE with small secrets (where the entries of $\mathbf{S}$ are uniformly chosen from $\{-d, d\}$). The algorithm first translates the binary-LWE problem into an inhomogeneous short integer solution problem and then solves the closest vector problem on the re-scaled lattice basis. The results of [3] confirm that n must be significantly enlarged if one works with binary vectors for the secret. Section 4.3 of Liu and Nguyen [15] also considers attacking a system with very small uniform errors. They find that the attack works for a much larger range of parameters than the general case, including $n = 350$. Hence, using binary secrets for the signatures application seems to not be a good way to get short signatures.

Alternatively, we could choose $\mathbf{S}$ and $\mathbf{E}$ from Gaussian with deviation $\sigma_S = \sigma_E \leq 2\sqrt{n}$. As a example we choose $(n, m, \sigma_S, \sigma_E, d, w) = (448, 886, 32, 32, 23, 19)$ and take $q \approx 2^{27.84}$. Here B is about $2^{19.74}$ which gives signatures of size $21n + 128 = 9540$ bits. The Hermite constant γ needed is roughly 1.0060. The acceptance probability is roughly 0.407.

In practice, Liu and Nguyen [15] have considered the best lattice decoding attacks and given security estimates for LWE. They use the BKZ 2.0 for basis reduction (based on estimates of Chen and Nguyen [9]) and then lattice enumeration algorithms for the decoding. Their results imply that $n = 256$ dimensional

lattices with Gaussian errors (of deviation about 3.3) should require around 2^{105} time to break. This suggests that, in practice, $n = 320$ dimensional lattices with Gaussian errors (of larger deviation) should be safe for LWE.

The size of σ_S and σ_E also affects relations in Equation (4). This turns out to be a stricter constraint: it is easier to find suitable parameters when σ_S and σ_E are large. In general, it seems that using small values for σ_S and σ_E may turn out to provide a relatively minor saving in signature size, so we do not pursue this idea further.

B.3 Bi-modal

We can also consider the bimodal technique of Ducas, Durmus, Lepoint and Lyubashevsky [10] in our setting. The main idea is to work modulo $2q$ and to choose the matrix $\mathbf{T}$ to be such that $-\mathbf{T} \equiv \mathbf{T} \pmod{2q}$. This can be achieved by ensuring that the entries of the matrix $\mathbf{T}$ are all in $\{0, q\}$. The matrix $\mathbf{T}$ can be represented using mn bits rather than $mn \log_2(2q)$ bits.

One particular choice for $\mathbf{T}$ is q times an $n \times n$ identity matrix (note that this requires $m = n$, which may result in a further increase in q. In this case the public key is further compressed, since there is no need to publish $\mathbf{T}$ at all. Let $k = n$ and suppose the $n \times n$ matrix $\mathbf{S}$ with Gaussian entries is invertible. Construct the public key $(\mathbf{A}, \mathbf{T})$ as follows. Choose $(\mathbf{S}, \mathbf{E})$ first and set $\mathbf{A} \equiv (\mathbf{T} - \mathbf{E})\mathbf{S}^{-1} \pmod{2q}$. The computational assumption is now related to the NTRU assumption: Given $\mathbf{A}$ find matrices $(\mathbf{S}, \mathbf{E})$ with small entries such that $\mathbf{A} \equiv -\mathbf{E}\mathbf{S}^{-1} \pmod{q}$. In particular, it is necessary (but not sufficient as far as we know) for this matrix-NTRU problem to be hard for our scheme to be secure. Note that the security of the short signature scheme of [10] also relies on an NTRU assumption (also in a ring) of a similar form.

The goal of using the bimodal distribution is that is makes the rejection sampling work better, and so one can use smaller distributions for D_y and D_z (indeed, Gaussian distributions). Using a Gaussian distribution for D_y with standard deviation σ and the bi-modal trick we should be able to take (as on page 5 of [10])

$$\sigma = 12\sqrt{n}\sigma_S\sqrt{w}/\sqrt{2}.$$

For instance, from parameters $(n, \sigma_S, w) = (448, 32, 19)$ we can obtain $\sigma = 12\sqrt{wn/2}\sigma_S \approx 25051$. Assuming a perfect encoding of Gaussian data that only requires $\log_2(4\sigma)$ bits to represent elements from this distribution we might hope to have signatures of around $448 \cdot \log_2(4\sigma) + 132 \approx 7575$ bits.

The security analysis of this variant requires a different use of the forking lemma, as well as a non-standard assumption. We do not give the details here.

B.4 Signatures based on Ring-LWE/NTRU

Our scheme could be implemented with Ring-LWE. The signature is a single ring element and the hash value. The public key is now a sequence $\mathbf{t}_i = \mathbf{a}_i\mathbf{s} + \mathbf{e}_i \pmod{q}$, for $1 \leq i \leq \ell$, with elements in $\mathbb{Z}_q[x]/(x^{2^k} + 1)$.

To sign we compute $\mathbf{c} = H(\lfloor \mathbf{a}_1\mathbf{y} \rceil_d \ldots \lfloor \mathbf{a}_\ell\mathbf{y} \rceil_d, \mu)$ and then $\mathbf{z} = \mathbf{y} + \mathbf{sc}$. Verification is that $\mathbf{z}$ is short and that $H(\lfloor \mathbf{a}_1\mathbf{z} - \mathbf{t}_1\mathbf{c} \rceil_d, \ldots \lfloor \mathbf{a}_\ell\mathbf{z} - \mathbf{t}_\ell\mathbf{c} \rceil_d \mu)$ equals $\mathbf{c}$.

The security proof is identical (one can always consider Ring-LWE as a particular case of the matrix problem), but of course the security now depends on the Ring-LWE assumption. Using Ring-LWE will reduce the public key size and improve speed, but it does not seem to lead to any reduction of the signature size, so we do not consider it further in this paper.

C Completing the proof of Theorem 3

This approach is rather different. When we replace the public key we insert an LWE instance into the key. The forgery therefore can be used to obtain a solution to the LWE instance.

The proof is a little less tight than the proof of Theorem 2 or the proofs in [10, 18], due to our non-standard use of the forking lemma. Our proof applies to many different distributions D_y, but it is simplest to consider D_y to be the uniform distribution $[-B, B]$. Hence, the full details are given for this case, and so we take $\ell = \infty$.

The idea of the proof is as follows: We define Game 2 to be the same as Game 1, except that the public key is replaced by a uniformly chosen pair of matrices $(\mathbf{A}, \mathbf{T})$. Game 3 is the same as Game 2, except it has a different key generation algorithm (see Algorithm 5 below). Lemma 8 shows that an adversary that can distinguish Game 1 and Game 2 is solving the matrix version of decisional-LWE. Lemma 9 shows that a distinguisher between Game 2 and Game 3 is solving decisional-LWE. Finally, we show that an adversary who can win Game 3 can be used to solve search-LWE.

Lemma 7. *Suppose q is prime and that condition (6) holds. Then, with probability $1-1/2^\kappa$, there is no non-zero vector $\boldsymbol{y} \in [-B, B]^n$ with $\lfloor \boldsymbol{A}\boldsymbol{y} \pmod q \rceil_d = 0$ and the output of $\lfloor \boldsymbol{A}\boldsymbol{y} \pmod q \rceil_d$ is uniform and the cardinality of the set*

$$\{\lfloor \boldsymbol{A}\boldsymbol{y} \pmod q \rceil_d : \boldsymbol{y} \leftarrow D_y^n\}$$

is $(2B+1)^n$.

Proof. Let q be a prime such that the equation (6) is satisfied. Fix a non-zero vector $\mathbf{y}$ and consider the set $\mathcal{A}_\mathbf{y} = \{\mathbf{A} \in \mathbb{Z}_q^{m \times n} : \lfloor \mathbf{A}\mathbf{y} \pmod q \rceil_d = 0\}$. These are the bad matrices for our vector $\mathbf{y}$. Considering each row of $\mathbf{A}$ we may choose $(n-1)$ positions arbitrarily, and one remaining entry is constrained to a set of 2^d values. Hence $\#\mathcal{A}_\mathbf{y} = \left(q^{n-1}2^d\right)^m$.

Consider any set of $(2B+1)^n$ choices for non-zero $\mathbf{y}$, then the union of all the bad sets has at most $(2B+1)^n \left(q^{n-1}2^d\right)^m$ elements. If a matrix $\mathbf{A}$ is not chosen from that union of sets then the function $\lfloor \mathbf{A}\mathbf{y} \pmod q \rceil_d$ is non-zero for all those $\mathbf{y}$. Hence, the probability a matrix $\mathbf{A}$ is bad for the set of $\mathbf{y}$ is

$$(2B+1)^n \left(q^{n-1}2^d\right)^m / q^{nm} \;=\; (2B+1)^n (2^d/q)^m < 1/2^\kappa.$$

Now, suppose that $\lfloor \mathbf{A}\mathbf{y}_1 \pmod q \rceil_d = \lfloor \mathbf{A}\mathbf{y}_2 \pmod q \rceil_d$. It follows that $\mathbf{y} = \mathbf{y}_1 - \mathbf{y}_2$ is a vector with $\mathbf{A}\mathbf{y} \pmod q$ having entries in $[-2^d, 2^d]$. In other words, $\mathbf{y} \in [-2B, 2B]^n$ is such that $\lfloor \mathbf{A}\mathbf{y} \pmod q \rceil_{d+1} = 0$. Applying the same argument as above on the enlarged parameters shows that such a vector $\mathbf{y}$ occurs with probability at most $1/2^\kappa$, and so with overwhelming probability there are no collisions in the function $\lfloor \mathbf{A}\mathbf{y} \pmod q \rceil_d$. □

Note that Lemma 7 prevents the fifth possible attack mentioned in Section 3.1.

Lemma 8. *Let the forger have success probability δ in Game 1 in the random oracle model. If the success probability of the forger in Game 2 is δ' with $\epsilon = |\delta' - \delta|$ then there is an algorithm to distinguish the distribution $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{S} + \boldsymbol{E} \pmod q)$ from the uniform distribution on $\mathbb{Z}_q^{m \times (n+k)}$ that runs in time approximately t and gives the correct answer with advantage ϵ.*

Proof. (Sketch) The signing algorithm in Game 2 is the same as Game 1, while the public key of Game 2 is a uniformly chosen pair of matrices $(\mathbf{A}, \mathbf{T})$.

Let A be a forger. We build a distinguisher D that takes as input a pair $(\mathbf{A}, \mathbf{T})$ of matrices, taken either from the uniform distribution or the LWE distribution. The distinguisher runs A on the public key $(\mathbf{A}, \mathbf{T})$. The forger A can make hash queries and sign queries, and the distinguisher answers these in the random oracle model as we have explained in Algorithm 4. Eventually the forger outputs a candidate signature. If the signature is valid then D outputs 1, else D outputs 0.

The advantage of D is defined to be the absolute value of the difference of the probabilities it outputs 1 when the input $(\mathbf{A}, \mathbf{T})$ is from each of the two distributions. This is ϵ by definition. □

Algorithm 5 Game 3 key generation ($\mathbf{T}\mathbf{w} \equiv \mathbf{b} \pmod q$).

INPUT: $\mathbf{A}, \mathbf{b}, \mathbf{w} \neq \mathbf{0}$
OUTPUT: $(\mathbf{A}, \mathbf{T})$
1: For notational convenience we suppose that $w_1 \neq 0$.
2: Write $\mathbf{w} = \left(\frac{w_1}{\mathbf{w}'}\right)$
3: $\mathbf{T}' \leftarrow \mathbb{Z}_q^{m \times (k-1)}$
4: Set $\mathbf{t} \equiv \mathbf{b} - \mathbf{T}'\mathbf{w}' \pmod q$
5: Set $\mathbf{T} = [w_1^{-1}\mathbf{t} \pmod q \mid \mathbf{T}']$ (This is now $m \times k$.)
6: **return** $(\mathbf{A}, \mathbf{T})$

Lemma 9. *Let the forger have success probability δ in Game 2 in the random oracle model. If the success probability of the forger in Game 3 is δ' with $\epsilon = |\delta' - \delta|$ then there is an algorithm to distinguish the LWE distribution $(\boldsymbol{A}, \boldsymbol{A}\boldsymbol{s} + \boldsymbol{e} \pmod q)$ from the uniform distribution on $\mathbb{Z}_q^{m \times (n+1)}$ that runs in time approximately t and gives the correct answer with advantage ϵ.*

Proof. (Sketch) Let $(\mathbf{A}, \mathbf{b})$ be the decisional-LWE instance and choose any vector $\mathbf{w}$ of low weight. Consider Algorithm 5 on input $(\mathbf{A}, \mathbf{b}, \mathbf{w})$. If $\mathbf{b}$ is sampled uniformly then the output matrix $\mathbf{T}$ is uniformly chosen, whereas if $\mathbf{b}$ is sampled from the LWE distribution then $\mathbf{T}$ is constrained. Any difference in the behaviour of the forger for these games therefore distinguishes $\mathbf{b}$ from random. □

Lemma 10. *Suppose the forger A plays Game 3, makes h hash function queries and s sign queries, runs in time t, and succeeds with probability δ'. Suppose the parameters satisfy the conditions in Theorem 3.*

Then there exists an algorithm running in time approx 2t and with success probability $\frac{\delta'}{h}\left(\frac{\delta'}{h} - \frac{1}{2^k}\right) + O(\frac{s^2}{2^\kappa} + \frac{n+m}{2^{140}})$ that solves (n, m, q, α)-LWE with short secrets.

Proof. (Sketch) Let $(\mathbf{A}, \mathbf{b})$ be the input (n, m, q, α)-LWE instance. We wish to find vectors $\mathbf{y}_1, \mathbf{y}_2$ such that $\mathbf{A}\mathbf{y}_1 + \mathbf{y}_2 \equiv \mathbf{b} \pmod q$ and we know that $\mathbf{y}_1, \mathbf{y}_2$ have been sampled from the discrete Gaussian distribution $D_{\alpha q}$. It follows that, with probability 2^{-140}, $\|\mathbf{y}_1\|_\infty, \|\mathbf{y}_2\|_\infty \leq 14\alpha q$ (see equation (1)).

Due to our choice of parameters $2B, 2^d \geq 14\alpha q$ we have that

$$\|\mathbf{y}_1\|_\infty \leq 2B \quad \text{and} \quad \|\mathbf{y}_2\|_\infty \leq 2^d. \tag{10}$$

Furthermore, equation (6) implies the LWE instance is well-defined in the sense that, with high probability, any pair $(\mathbf{y}_1', \mathbf{y}_2')$ of vectors satisfying the bounds of equation (10) is actually the desired pair $(\mathbf{y}_1, \mathbf{y}_2)$ corresponding to the distribution $D_{\alpha q}$.

Choose uniformly at random $c^*, c^\dagger \leftarrow \{0,1\}^\kappa$ and set $\mathbf{w} = F(c^*) - F(c^\dagger)$. Then $\mathbf{w}$ is a length k vector over $\mathbb{Z}$ but we do not need any restriction on its weight (as long as $\mathbf{w} \neq \mathbf{0}$, which fails with negligible probability $c_1/2^\kappa$ by equation (2)). Then perform the key generation algorithm in Algorithm 5. The point is that $\mathbf{T}\mathbf{w} \equiv \mathbf{b} \pmod q$.

We now guess an index $1 \leq I \leq h$ and run the forger A on $(\mathbf{A}, \mathbf{T})$. The forger makes hash and sign queries. Hash queries are answered by choosing fresh uniform values that are stored in a list, with the exception that the I-th hash query is answered with c^*. Sign queries are simulated in the random oracle model as usual.

Eventually A outputs a valid signature $(\mathbf{z}, c)$ on message μ. We know that the random oracle has been queried in order for the verification equation $c = H(\lfloor \mathbf{v} \rceil_d, \mu)$ to hold.

We will now explain that c arises with overwhelming probability from a hash query, not a sign query. Suppose not: then a sign query on a message μ' with output equals to $(\mathbf{z}', c)$, and hence we have $c = H(\lfloor \mathbf{A}\mathbf{z} - \mathbf{T}\mathbf{c} \pmod q \rceil_d, \mu) = H(\lfloor \mathbf{A}\mathbf{z}' - \mathbf{T}\mathbf{c} \pmod q \rceil_d, \mu')$. We have $\mu' = \mu$ with probability $1 - 1/2^\kappa$. Otherwise, we find a pre-image of c with non-negligible probability. Hence, with overwhelming probability $\mathbf{A}(\mathbf{z} - \mathbf{z}') \pmod q$ has entries in $[-2^d, 2^d]$ and so, by Lemma 3, $\mathbf{z}' = \mathbf{z}$ with probability $(1 - 1/2^\kappa)$. Hence, for the remainder of the proof we may assume that the forgery $(\mathbf{z}, c)$ has c an output of a random oracle query that was not made as part of a sign query.

Now we apply the forking lemma. In other words, we re-wind the attack, so that $\mathbf{v}$ is the same but the I-th random oracle output is taken to be the binary string $c^\dagger$ such that $F(c^*) = F(c^\dagger) + \mathbf{w}$.

One can verify that our signature scheme is a generic signature scheme with security parameter κ. As discussed in Appendix A, we use a variant of the Bellare-Neven [4] formulation given in equation (9). This takes into account that we need a forgery with hash value taken to be the specific value c^* chosen at the start of the proof, and then replace the same random oracle query with the value $c^\dagger$. Note that both values c^* and $c^\dagger$ are chosen independently and uniformly at random, as required to apply the forking Lemma. It follows that we may obtain a pair of valid signatures $(c^*, \mathbf{z}^*)$ and $(c^\dagger, \mathbf{z}^\dagger)$ corresponding to the same hash input $(\lfloor \mathbf{v} \rceil_d, \mu)$ in time approximately $2t$ and with probability at least

$$\frac{\delta'}{h}\left(\frac{\delta'}{h} - \frac{1}{2^\kappa}\right).$$

We have

$$\mathbf{A}\mathbf{z}^* - \mathbf{T}F(c^*) \equiv \mathbf{v} + \mathbf{e}^* \pmod{q}$$

and, similarly,

$$\mathbf{A}\mathbf{z}^\dagger - \mathbf{T}F(c^\dagger) \equiv \mathbf{v} + \mathbf{e}^\dagger \pmod{q}.$$

Subtracting gives

$$\mathbf{A}(\mathbf{z}^* - \mathbf{z}^\dagger) + (\mathbf{e}^\dagger - \mathbf{e}^*) \equiv \mathbf{T}(F(c^*) - F(c^\dagger)) = \mathbf{T}\mathbf{w} \equiv \mathbf{b} \pmod{q}.$$

Note that $\|\mathbf{z}^*\|_\infty, \|\mathbf{z}^\dagger\|_\infty \leq B$ and so $\|\mathbf{z}^* - \mathbf{z}^\dagger\|_\infty \leq 2B$. Similarly, $\|\mathbf{e}^*\|_\infty, \|\mathbf{e}^\dagger\|_\infty \leq 2^{d-1}$ and so $\|\mathbf{e}^\dagger - \mathbf{e}^*\|_\infty \leq 2^d$. Hence, by uniqueness of the LWE solution, $(\mathbf{z}^* - \mathbf{z}^\dagger, \mathbf{e}^\dagger - \mathbf{e}^*)$ is the desired solution to the LWE instance. □

This completes the proof of Theorem 3.