Algebraic Techniques for Short(er) Exact Lattice-Based Zero-Knowledge Proofs

Jonathan Bootle[1], Vadim Lyubashevsky[1], and Gregor Seiler[1,2]

[1] IBM Research – Zurich, Switzerland
[2] ETH Zurich, Switzerland

Abstract. A key component of many lattice-based protocols is a zero-knowledge proof of knowledge of a vector $\vec{s}$ with small coefficients satisfying $A\vec{s} = \vec{u} \bmod q$. While there exist fairly efficient proofs for a relaxed version of this equation which prove the knowledge of $\vec{s}'$ and c satisfying $A\vec{s}' = \vec{u}c$ where $\|\vec{s}'\| \gg \|\vec{s}\|$ and c is some small element in the ring over which the proof is performed, the proofs for the exact version of the equation are considerably less practical. The best such proof technique is an adaptation of Stern's protocol (Crypto '93), for proving knowledge of nearby codewords, to larger moduli. The scheme is a Σ-protocol, each of whose iterations has soundness error 2/3, and thus requires over 200 repetitions to obtain soundness error of 2^{-128}, which is the main culprit behind the large size of the proofs produced.

In this paper, we propose the first lattice-based proof system that significantly outperforms Stern-type proofs for proving knowledge of a short $\vec{s}$ satisfying $A\vec{s} = \vec{u} \bmod q$. Unlike Stern's proof, which is combinatorial in nature, our proof is more algebraic and uses various relaxed zero-knowledge proofs as sub-routines. The main savings in our proof system comes from the fact that each round has soundness error of $1/n$, where n is the number of columns of A. For typical applications, n is a few thousand, and therefore our proof needs to be repeated around 10 times to achieve a soundness error of 2^{-128}. For concrete parameters, it produces proofs that are around an order of magnitude smaller than those produced using Stern's approach.

Keywords. Lattices, Zero-Knowledge Proofs, Commitments

1 Introduction

Lattice based cryptography is viewed as one of the most promising post-quantum replacements for traditional public key cryptography because the most crucial cryptographic primitives, such as public key encryption and digital signatures, can be efficiently constructed from lattice assumptions. Furthermore, there exist cryptographic primitives (e.g. FHE [Gen09]) whose only current realization stems from lattice assumptions.

1.1 Lattice-based Zero-Knowledge Proofs.

A central part of many lattice protocols is a zero-knowledge proof of a vector $\vec{s}$ satisfying

$$A\vec{s} = \vec{u} \bmod q \tag{1}$$

for public $A \in \mathbb{Z}_q^{m\times n}$ and $\vec{u} \in \mathbb{Z}_q^m$. Current lattice-based zero-knowledge proofs for the above equation come in several varieties. The most direct approach proves exactly the knowledge of $\vec{s}$ satisfying (1) that the prover uses to generate the proof. This proof system [KTX08,LNSW13] is an adaptation of Stern's protocol [Ste93], which proves knowledge of nearby codewords, to larger moduli. Its main weakness is that each iteration of the proof has soundness error 2/3 and it therefore needs to be repeated 219 times to achieve soundness error 2^{-128}. For typical applications where the modulus is $q \approx 2^{30}$, the size of such a proof is several megabytes long.

There are other protocols that give "relaxed" proofs of (1), which may be useful in some situations. The Fiat-Shamir-with-Aborts [Lyu09,Gro10,Lyu12] approach proves knowledge of an $\vec{s}'$ and c satisfying $A\vec{s}' = c\vec{u} \bmod q$. Despite the fact that $\|\vec{s}'\| > \|\vec{s}\|$ and the presence of an extra factor c, this technique is useful for producing short lattice-based primitives, such as digital signatures, when performed over polynomial rings. The reason that these protocols are so efficienct is that each run of the protocol has negligible soundness error and so only needs to be performed once. Another approach [BCK$^+$14] proves the knowledge of $\vec{s}'$ satisfying $A\vec{s}' = 2\vec{u} \bmod q$ for $\|\vec{s}'\| > \|\vec{s}\|$. When performed over polynomial rings, the soundness error of this protocol $1/2d$, where d is the dimension of the ring. In the case where one has many equations as in (1) for the same A, but different $\vec{s}_i$ and $\vec{u}_i$, there are even sub-linear size proofs [BBC$^+$18] showing that $A\vec{s}_i' = \vec{u}_i$, where $\|\vec{s}_i'\| > \|\vec{s}_i\|$.

The main downside in all of the aforementioned efficient proofs is that even though they are more efficient than Stern-type proofs, they always prove knowledge of an $\vec{s}'$ which is larger than the $\vec{s}$ that the prover knows. Other than the Stern-type proofs mentioned above, the only other known proof system that exactly proves knowledge of the $\vec{s}$ in (1) is based on the hardness of the discrete logarithm problem [dPLS19] using the "Bulletproofs" [BCC$^+$16,BBB$^+$18] approach which results in short proofs, but long running times (in addition to requiring the discrete logarithm assumption).[3] The disadvantage of proofs that prove knowledge of a larger $\vec{s}'$ is that, for security reasons, they force the modulus q to be larger. When the zero-knowledge proof is the main part of the protocol one is building (e.g. group signatures [LNWX18,dPLS18]), this trade-off may be worthwhile. On the other hand, if one would like to use a zero-knowledge proof to prove something about a relation used in a different protocol (e.g. proving that the public key of a lattice based encryption scheme is well-formed), then

[3] Since the submission of this paper, independently achieved results (some using very different techniques) appeared for solving versions of this problem [Beu19,BN19,YAZ$^+$19].

one may not want to increase the parameters of the scheme just to make the zero-knowledge proof more efficient.

In this paper, we present a new proof technique for exactly proving (1) which is different from the "combinatorial" approach of Stern-like proofs. The proof crucially uses the connection between the coefficient and the NTT (i.e. FFT) representation in polynomial rings, invokes a "relaxed" lattice-based commitment scheme as a sub-routine, and uses some "tricks" present in certain discrete-log based proofs (e.g. [GK15]).

1.2 Our Approach

The basic building block of our zero-knowledge proof system is a proof of knowledge of an $\vec{s}$ with coefficients in $\{0, 1, 2\}$ satisfying (1). One can easily transform this into a proof system where $\vec{s}$ comes from the more typical space of $\{-1, 0, 1\}$ and also extend it into a proof system where $\vec{s}$ has coefficients in the set $\{-S \ldots, S\}$. The former is trivial, whereas the latter involves rewriting $\vec{s} = G\vec{s}'$, where $\vec{s}'$ has coefficients in $\{0, 1, 2\}$ for some public matrix G, and then proving knowledge of an $\vec{s}'$ such that $A'\vec{s}' = \vec{u}$ for $A' = AG$.

Notice that in the above transformation from a basic proof to a general one, the larger the size of the basic set is, the fewer columns A' will have – which is good for keeping the proof size small. On the other hand, as we'll see below, the larger the basic set is, the larger the proof will be for the fact that each coefficient of $\vec{s}'$ is in the basic set. When picking the size of the basic set, it is thus important to balance these two conditions to obtain the optimally minimal proof size. It turns out that choosing the basic set to be $\{0, 1, 2\}$ is very close to the optimal choice. Furthermore, $\{0, 1, 2\}$ (which is equivalent to $\{-1, 0, 1\}$) is often the actual set in which we want to prove that the coefficients of the solution lie. We therefore choose to work with this set for the remainder of the paper.

If $\vec{1}$ and $\vec{2}$ are n-dimensional vectors consisting of 1's and 2's, then proving that the coefficients are all in the set $\{0, 1, 2\}$ is equivalent to showing that

$$\vec{s} \circ (\vec{s} - \vec{1}) \circ (\vec{s} - \vec{2}) = \vec{0} \bmod q,$$

where $\circ$ denotes component-wise multiplication. Let us now consider a polynomial ring $\mathcal{R} = \mathbb{Z}_q[X]/(f(X))$ where $f(X)$ is a polynomial of degree n that splits into linear factors modulo q. For example, if n is a power of 2 and $q \equiv 1 \pmod{2n}$, then $f(X) = X^n + 1$ is a good polynomial to use. If $\mathbf{s} \in \mathcal{R}$ satisfies $\mathbf{s}(\mathbf{s} - \mathbf{1})(\mathbf{s} - \mathbf{2}) = \mathbf{0}$, then it also holds that $\hat{\mathbf{s}} \circ (\hat{\mathbf{s}} - \hat{\mathbf{1}}) \circ (\hat{\mathbf{s}} - \hat{\mathbf{2}}) = \hat{\mathbf{0}} \bmod q$, where $\hat{\mathbf{s}}$ is the NTT (or FFT) representation of $\mathbf{s}$. Since the we chose our ring so that $f(X)$ fully splits, $\hat{\mathbf{s}}$ is a vector of dimension n and so $\hat{\mathbf{1}}$ and $\hat{\mathbf{2}}$ are n-dimensional vectors consisting entirely of 1's and 2's. We can now rephrase what we're trying to prove in terms of polynomials and their NTT representations. Proving the knowledge of a polynomial $\mathbf{s} \in \mathcal{R}$ such that

$$\mathbf{s}(\mathbf{s} - \mathbf{1})(\mathbf{s} - \mathbf{2}) = \mathbf{0} \text{ and } A\hat{\mathbf{s}} = \vec{u} \bmod q \tag{2}$$

is equivalent to proving the knowledge of $\vec{s}$ with coefficients in $\{0, 1, 2\}$ satisfying (1).

Our proof of knowledge of (2) begins with the prover picking a random masking polynomial $\mathbf{y} \in R$ and producing a $\vec{w} = A\hat{\mathbf{y}} \bmod q$. For a challenge $c \in \mathbb{Z}_q \subset \mathcal{R}$, if the prover outputs $\mathbf{z} = \mathbf{y} + c\mathbf{s}$, then this can be rewritten as $\hat{\mathbf{z}} = \hat{\mathbf{y}} + c\hat{\mathbf{s}}$, and therefore the verifier can check the equation

$$A\hat{\mathbf{z}} = \vec{w} + c\vec{u}. \tag{3}$$

Notice that in order for $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ to imply (3), it is crucial that $c \in \mathbb{Z}_q$, since this is the only way that all the coefficients of $\hat{c}$ can be identical. By rewinding, we can obtain another equation $A\hat{\mathbf{z}}' = \vec{w} + c'\vec{u}$, for a $c' \neq c$ and combining the two we get

$$A(\hat{\mathbf{z}} - \hat{\mathbf{z}}') = (c - c')\vec{u}. \tag{4}$$

The above does not really prove (1) since we still do not know that the coefficients of $(\hat{\mathbf{z}} - \hat{\mathbf{z}})$ are in $\{0, 1, 2\}$ and there is the term $(c - c') \in \mathbb{Z}_q$ which is not necessarily 1. Let us first describe how the latter problem is handled. In the first step, the prover additionally makes commitments $\mathrm{Com}(\mathbf{y})$ and $\mathrm{Com}(\mathbf{s})$ to $\mathbf{y}$ and $\mathbf{s}$ using the commitment scheme from [BDL+18] which has the property that for any $c \in \mathcal{R}$

$$\mathrm{Com}(\mathbf{y}) + c \cdot \mathrm{Com}(\mathbf{s}) = \mathrm{Com}(\mathbf{y} + c\mathbf{s}).$$

After receiving the challenge c, the prover will prove that $\mathrm{Com}(\mathbf{y}) + c \cdot \mathrm{Com}(\mathbf{s})$ is a commitment to $\mathbf{z}$, which implies that $\mathbf{y} + c\mathbf{s} = \mathbf{z}$, and therefore after rewinding, $(c - c')\mathbf{s} = \mathbf{z} - \mathbf{z}'$. Plugging this into (4) implies that $A(c - c')\hat{\mathbf{s}} = (c - c')\vec{u}$, and since $c \neq c'$ and q is prime, we can divide out by $(c - c')$ to obtain

$$A\hat{\mathbf{s}} = \vec{u}. \tag{5}$$

What we still have not proved is that the coefficients of $\hat{\mathbf{s}}$ (or more precisely, the NTT coefficients of the message that was committed to in $\mathrm{Com}(\mathbf{s})$) are in $\{0, 1, 2\}$. For this proof, we make the observation that

$$\mathbf{z}(\mathbf{z} - c)(\mathbf{z} - 2c) = \mathbf{y}^3 + 3\mathbf{y}^2(\mathbf{s} - 1)c + \mathbf{y}(3\mathbf{s}^2 - 6\mathbf{s} + 2)c^2 + \mathbf{s}(\mathbf{s} - 1)(\mathbf{s} - 2)c^3. \tag{6}$$

In particular, the last coefficient of the above polynomial is exactly what we would like to prove equals to $\mathbf{0}$. In the first step of the protocol, the prover will also commit to

$$\begin{aligned}
\mathbf{t}_0 &= \mathrm{Com}(\mathbf{y}^3)\\
\mathbf{t}_1 &= \mathrm{Com}(3\mathbf{y}^2(\mathbf{s} - 1))\\
\mathbf{t}_2 &= \mathrm{Com}(\mathbf{y}(3\mathbf{s}^2 - 6\mathbf{s} + 2))
\end{aligned}$$

and after receiving the challenge c, he will again use the linearity of the commitment scheme to show that

$$\mathrm{Com}(\mathbf{z}(\mathbf{z} - c)(\mathbf{z} - 2c)) = \mathbf{t}_0 + c\mathbf{t}_1 + c^2\mathbf{t}_2 \tag{7}$$

Intuitively (by an argument similar to the Schwartz-Zippel Lemma) this implies that $\mathbf{z}(\mathbf{z}-c)(\mathbf{z}-2c)$ as written in (6) is indeed a polynomial that is quadratic in c and therefore the last term of (6) is $\mathbf{0}$ as we wanted.

The one thing that is still left to do is show that all the commitments that we made are valid. For this we use the proof from [BDL$^+$18], paying careful attention to the fact that the challenges have to come from a set whose differences are all invertible. To make this set large, [BDL$^+$18] proposed the use of a polynomial ring $\mathcal{R}$ such that the underlying polynomial $f(X)$ splits into a few high-degree irreducible terms. But in our proof, we crucially need $f(X)$ to fully split, and so the largest set that we can use is $\{0, \pm X^i\}$, for $0 \leq i < n$, which is of size $2n+1$. Thus the commitment validity proof needs to be repeated $128/\log 2n$ times.

Decreasing the soundness error. Looking at the number of repetitions, the challenge c comes from the set $\mathbb{Z}_q$ and therefore one would need $128/\log q$ such challenges for achieving 128-bit security. Since we mentioned above that the challenges for proving the commitments are valid come from a set of size $2n$, one may naïvely assume that $(128/\log q)\cdot(128/\log 2n)$ (parallel) repetitions of the protocol will be necessary – but this would be an overestimate. The prover does not have to convince the verifier that each commitment is correct with overwhelming probability. Intuitively, the probability of the verifier cheating is if he can guess the challenge c (which is $\frac{1}{q}$) or he can create an invalid commitment and not get caught. If the latter probability is ρ, then the probability of the verifier cheating is less than $\frac{1}{q}+\rho$. It is therefore not very useful to decrease ρ below $\frac{1}{q}$. Thus the commitment validity proof will need to be repeated (in parallel) a total of $128/\log 2n$ times (for the whole protocol) and the number proofs of (1) that will need to be made, conditioned on the commitment being valid, is $128/\log q$.

Observations. One interesting observation is that our proof crucially uses polynomial rings and the security of the commitment scheme in [BDL$^+$18] based on problems in polynomial rings, but the original problem instance in (1) is only viewed as a linear equation over $\mathbb{Z}_q$. One could, of course, have (1) represent an equation over some ring $\mathcal{R}'$ (which is not the same as the ring $\mathcal{R}$ that we did the proof over!) by having the matrix A be "structured". For example if A consists of concatenations of rotation matrices, then (1) is a polynomial equation over $\mathbb{Z}_q[X]/(X^m-1)$. So our proof can also be seen as a way to give a more efficient proof (of any kind, even relaxed) of a linear equation over $\mathbb{Z}_q$, though relying on the hardness of problems over polynomial rings.

While the beginning of our proof may have some similarity to proofs of a relaxed version of (1) (e.g. [Lyu09,Lyu12]), we believe that the resemblance is only superficial. Ignoring the NTT step (which is not present in other protocols), the extracted values of $\hat{\mathbf{z}}$ and c in (4) somewhat resemble what one obtains in the final step of the aforementioned protocols. In those protocols, the values of $\hat{\mathbf{z}}$ are constructed to be small (but still larger than $\hat{\mathbf{s}}$) by choosing the $\hat{\mathbf{y}}$ from a particular narrow distribution and using rejection sampling to keep the $\hat{\mathbf{z}}$ small. In the current proof, however, the $\hat{\mathbf{z}}$ are not small and everything about the size of the secret $\hat{\mathbf{s}}$ is proved elsewhere in the protocol by showing that the c^3 term of (6) is $\mathbf{0}$.

2 Preliminaries

2.1 Notation

The following table summarizes the notation and parameters that will appear in this paper.

Parameter	Explanation
$q \equiv 1 \pmod{l}$	Prime modulus that splits completely in $\mathcal{R}$
$\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$	The field over which the linear system is defined
$m \in \mathbb{Z}$	The number of rows in the linear system
$n \in \mathbb{Z}$	The number of columns in the linear system and the rank of $\mathcal{R}$
$\Phi_l \in \mathbb{Z}[X]$	The l-th cyclotomic polynomial of degree $n = \varphi(l)$
$\mathcal{R} = \mathbb{Z}[X]/(\Phi_l)$	The ring of integers in the l-th cyclotomic number field
$\mathcal{R}_q = \mathbb{Z}_q[X]/(\Phi_l)$	The ring of integers $\mathcal{R}$ modulo the fully splitting rational prime q
$\mathcal{C} \subset \mathcal{R}$	A set of low-weight challenge polynomials
$\bar{\mathcal{C}} = (\mathcal{C} - \mathcal{C}) \setminus \{\mathbf{0}\}$	The set of challenge differences excluding $\mathbf{0}$
T	Bound for honest prover's $\mathbf{f}\vec{\mathbf{r}}$ in embedding norm
$\sigma = 5T$	Standard deviation for sampling $\vec{\mathbf{y}}'$
$B = \sigma\sqrt{12n}$	Bound for honest prover's $\vec{\mathbf{z}}'$ in embedding norm
β^n	Error distribution on $\mathcal{R}$ in the RLWE problem
D^n_σ	Discrete Gaussian distribution on $\mathcal{R}$ with standard deviation σ

Table 1. Overview of Parameters and Notation

We use bold letters $\mathbf{f}$ for polynomials in $\mathcal{R}$, arrows as in $\vec{v}$ for integer vectors $\vec{v} \in \mathbb{Z}^k$, bold letters with arrows $\vec{\mathbf{v}}$ for vectors of polynomials $\vec{\mathbf{v}} \in \mathcal{R}^k$ and capital letters A and $\mathbf{A}$ for integer and polynomial matrices, respectively. We write $x \overset{\$}{\leftarrow} S$ when $x \in S$ is sampled uniformly at random from the set S and similarly $x \overset{\$}{\leftarrow} \rho$ when x is sampled according to the distribution ρ.

As is often the case in ring-based lattice cryptography, computation will be performed in the quotient ring $\mathcal{R}_q$ modulo q of the ring of integers $\mathcal{R}$ of the l-th cyclotomic number field. The geometry on $\mathcal{R}$ is inherited by embedding $\mathcal{R}$ into the Minkowski space, an n-dimensional real subspace of $\mathbb{C}^n$. Concretely, for $\mathbf{f}, \mathbf{g} \in \mathcal{R}$, we have the scalar product and its induced norm

$$\langle \mathbf{f}, \mathbf{g} \rangle = \sum_{j \in \mathbb{Z}_l^\times} f(\zeta^j)\overline{g(\zeta^j)} \quad \text{and}$$

$$\|\mathbf{f}\|_2 = \left(\sum_{j \in \mathbb{Z}_l^\times} |f(\zeta^j)|^2 \right)^{\frac{1}{2}},$$

where ζ is the primitive l-th complex root of unity $\zeta = e^{2\pi i/l}$. In the special case where $\mathcal{R}$ is a power-of-two cyclotomic ring, i.e. $l = 2^r$, the norm of $\mathbf{f} = f_0 + \cdots + f_{n-1}X^{n-1}$ is identical, up to a scaling factor, to the ℓ_2-norm of the coefficient vector $\vec{f} = (f_0, \dots, f_{n-1}) \in \mathbb{Z}^n$; that is,

$$\|\mathbf{f}\|_2 = \sqrt{n} \left(\sum_{i=1}^{n} |f_i|^2 \right)^{\frac{1}{2}} = \sqrt{n} \|\vec{f}\|_2.$$

The scalar product and norm are extended to vectors $\vec{\mathbf{v}} = (\mathbf{v}_1, \dots, \mathbf{v}_k), \vec{\mathbf{w}} = (\mathbf{w}_1, \dots, \mathbf{w}_k) \in \mathcal{R}^k$ of polynomials in the natural way,

$$\langle \vec{\mathbf{v}}, \vec{\mathbf{w}} \rangle = \sum_{i=1}^{k} \langle \mathbf{v}_i, \mathbf{w}_i \rangle,$$

$$\|\vec{\mathbf{v}}\|_2 = \left(\sum_{i=1}^{k} \|\mathbf{v}_i\|_2^2 \right)^{\frac{1}{2}}.$$

2.2 Fully Splitting Primes and the Number Theoretic Transform

Our prime modulus q needs to be such that q splits completely in $\mathcal{R}$; that is, the cyclotomic polynomial Φ_l needs to factor into linear polynomials modulo q. This is the case if and only if there exists a primitive l-th root of unity modulo q, which in turn is equivalent to the condition $q - 1 \equiv 0 \pmod{l}$.

Then, by the Chinese remainder theorem, we have that $\mathcal{R}_q = \mathbb{Z}_q[X]/\Phi_l$ is isomorphic $\mathbb{Z}_q^n$. Concretely,

$$\mathbb{Z}_q[X]/(\Phi_l) \cong \mathbb{Z}_q[X]/(X - \zeta_1) \times \cdots \times \mathbb{Z}_q[X]/(X - \zeta_n)$$

where $\zeta_1, \dots, \zeta_n$ are the primitive l-th roots of unity modulo q. The isomorphism is given by reduction modulo $X - \zeta_i$. We write $\hat{\mathbf{f}}$ for the image of $\mathbf{f} \in \mathcal{R}_q$ under the isomorphism and call it the *Number Theoretic Transform* (NTT) of $\mathbf{f}$.

When l is a product of powers of small primes then the NTT (and its inverse) can be computed very efficiently in a divide an conquer fashion. Especially popular is the optimal case where l is a power of two and indeed many schemes in lattice cryptography are instantiated over such a ring.

The existence of a fast NTT algorithm for computing the isomorphism speeds-up and simplifies computation but is not crucial for our results. Therefore we do not go into more details here.

2.3 Challenge Space

We define the challenge space $\mathcal{C} \subset \mathcal{R}$ as

$$\mathcal{C} = \{0, X^i \mid 0 \leq i < l\}.$$

The crucial property of this set is that the difference of any two members is invertible in $\mathcal{R}$ and the multiplication of any element in $\mathcal{R}$ by any member of the set does not increase the norm of the element.

In the area of lattice-based zero-knowledge proofs, when $\mathcal{R}$ is a power of two cyclotomic ring, there is a sometimes-used stronger result stating that $2(X^i - X^j)^{-1}$ exists and has ternary coefficients in $\{-1, 0, 1\}$; c.f. [BCK$^+$14, Lemma 3.1]. In our application we do not need any condition on the smallness of the inverse of a difference of challenges and hence we will not be concerned with this property.

Lemma 2.1. *The polynomials $X^i - X^j \in \mathcal{R}_q$ for $i \not\equiv j \pmod{l}$ are invertible.*

Proof. Let $\zeta \in \mathbb{Z}_q$ be one of the primitive l-th roots of unity. Then $X^i - X^j \mod X - \zeta = \zeta^i - \zeta^j$. The latter is zero in $\mathbb{Z}_q$ if and only if $i \equiv j \pmod{l}$. □

2.4 Error Distribution, Discrete Gaussians and Rejection Sampling

For sampling randomness in the commitment scheme that we use, and to define a variant of the Ring Learning with Errors problem, we need to define an error distribution β^n on $\mathcal{R}$. For general cyclotomic rings one has to be careful when sampling error polynomials as one has to do it over the Minkowski space [LPR13].

For power-of-two cyclotomics, however, it is secure and much easier to directly sample the polynomial coefficients. Moreover, we will need to bound the norm of error polynomials and these bounds turn out to be slightly better when sampling the coefficients using a uniform or binomial distribution on a small interval instead of a small one-dimensional (discrete) Gaussian. Also this is much easier to implement in practice. Therefore, in the power-of-two case we sample the coefficients of the random polynomials in the commitment scheme using the distribution β_2 on $\{-1, 0, 1\}$ where ±1 both have probability 5/16 and 0 has probability 6/16. This distribution is chosen (rather than the more "natural" uniform one) because it is easy to sample given a random bitstring by computing $a_1 + a_2 - b_1 - b_2 \bmod 3$ with uniformly random bits a_i, b_i. Now if $\vec{v} \xleftarrow{\$} \beta_2^n$ then we have the Chernov bound for $0 < \delta \leq 1$ given by

$$\Pr\left[\|\vec{v}\|_2 < \sqrt{(1+\delta)\frac{10}{16}n}\right] \geq 1 - \exp\left(-\frac{\delta^2}{3}\frac{10}{16}n\right). \tag{8}$$

In our zero-knowledge proof, the prover will want to output a vector $\vec{\mathbf{z}}$ whose distribution should be independent of a secret randomness vector $\vec{\mathbf{r}}$, so that $\vec{\mathbf{z}}$ cannot be used to gain any information on the prover's secret. During the protocol, the prover computes $\vec{\mathbf{z}} = \vec{\mathbf{y}} + \mathbf{f}\vec{\mathbf{r}}$ where $\vec{\mathbf{r}}$ is the randomness used to commit to the prover's secret, $\mathbf{f} \leftarrow \mathcal{C}$ is a challenge polynomial, and $\vec{\mathbf{y}}$ is a "masking" vector. To remove the dependency of $\vec{\mathbf{z}}$ on $\vec{\mathbf{r}}$, we use the rejection sampling technique by Lyubashevsky [Lyu09,Lyu12]. In the two variants of this technique the masking vector is either sampled uniformly from some bounded region or using a discrete Gaussian distribution. In the high dimensions we will encounter, the Gaussian variant is far superior as it gives acceptable rejection probabilities for much narrower distributions. We first define the discrete Gaussian distribution and then state the rejection sampling algorithm in Figure 1, which plays a central role in Lemma 2.4.

Definition 2.2. *The* discrete Gaussian distribution *on $\mathcal{R}^k$ centered around $\vec{\mathbf{v}} \in \mathcal{R}^k$ with standard deviation $\sigma > 0$ is given by*

$$D^{kn}_{\mathbf{v},\sigma}(\vec{\mathbf{z}}) = \frac{e^{-\|\vec{\mathbf{z}}-\vec{\mathbf{v}}\|_2^2/2\sigma^2}}{\sum_{\vec{\mathbf{z}}' \in \mathcal{R}^k} e^{-\|\vec{\mathbf{z}}'\|_2^2/2\sigma^2}}.$$

When it is centered around $\vec{\mathbf{0}} \in \mathcal{R}^k$ we write $D^{kn}_\sigma = D^{kn}_{\vec{\mathbf{0}},\sigma}$

We will use the following tail bound, which follows from [Ban93, Lemma 1.5(i)].

Lemma 2.3. *Let $\vec{\mathbf{z}} \overset{\$}{\leftarrow} D^{kn}_\sigma$. Then*

$$\Pr\left[\|\vec{\mathbf{z}}\|_2 \leq \sigma\sqrt{2kn}\right] \geq 1 - 2^{-\log(e/2)kn/4}.$$

Algorithm 1 Rej $(\vec{\mathbf{z}}, \vec{\mathbf{v}}, \sigma)$

```
$u \overset{\$}{\leftarrow} [0, 1)$
**if** $u < \frac{1}{12} \cdot \exp\left(\frac{-2\langle\vec{\mathbf{z}},\vec{\mathbf{v}}\rangle + \|\vec{\mathbf{v}}\|_2}{2\sigma^2}\right)$ **then**
    **return** 0
**else**
    **return** 1
**end if**
```

Lemma 2.4. *Let $\rho \colon \mathcal{R}^k \to [0, 1]$ be a probability distribution such that, for some $T > 0$, $\rho(\{\vec{\mathbf{v}} \in \mathcal{R}^k \mid \|\vec{\mathbf{v}}\|_2 \leq T\}) \geq 1 - 2^{-101}$ and let $\sigma \geq 5T$. Sample $\vec{\mathbf{v}} \overset{\$}{\leftarrow} \rho$ and $\vec{\mathbf{y}} \overset{\$}{\leftarrow} D^{kn}_\sigma$, set $\vec{\mathbf{z}} = \vec{\mathbf{y}} + \vec{\mathbf{v}}$, and run $b \leftarrow$ Rej $(\vec{\mathbf{z}}, \vec{\mathbf{v}}, \sigma)$. Then, the probability that $b = 0$ is at least $1/12 - 2^{-104}$ and the distribution of $(\vec{\mathbf{v}}, \vec{\mathbf{z}})$, conditioned on $b = 0$, is within statistical distance of 2^{-100} of the product distribution $\rho \times D^{kn}_\sigma$.*

The proof is essentially the same as in [Lyu12, Theorem 4.6], but we include it here for the sake of completeness since the statement in [Lyu12] is slightly different.

Proof. For every $\vec{\mathbf{v}}' \in \mathcal{R}^k$ let $S_{\vec{\mathbf{v}}'} \subset \mathcal{R}^k$ be the set of vectors $\vec{\mathbf{z}}'$ such that

$$\frac{D^{kn}_\sigma(\vec{\mathbf{z}}')}{D^{kn}_{\vec{\mathbf{v}}',\sigma}(\vec{\mathbf{z}}')} \leq 12.$$

By a simple variant of [Lyu12, Lemma 4.5] it follows that for all $\vec{\mathbf{v}}'$ such that $\|\vec{\mathbf{v}}'\|_2 \leq T$,

$$D_\sigma^{kn}(S_{\vec{\mathbf{v}}'}) \geq 1 - 2^{-102}.$$

Then,

$$\begin{aligned}
\Pr\left[b = 0\right] &= \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \in \mathcal{R}^k} D_{\vec{\mathbf{v}}',\sigma}^{kn}(\vec{\mathbf{z}}') \min\left(\frac{1}{12} \frac{D_\sigma^{kn}(\vec{\mathbf{z}}')}{D_{\vec{\mathbf{v}}',\sigma}^{kn}(\vec{\mathbf{z}}')}, 1\right) \\
&\geq \sum_{\|\vec{\mathbf{v}}'\|_2 \leq T} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \in S_{\vec{\mathbf{v}}'}} \frac{1}{12} D_\sigma^{kn}(\vec{\mathbf{z}}') \\
&\geq \frac{1}{12}(1 - 2^{-101})(1 - 2^{-102}) > \frac{1}{12} - 2^{-104}.
\end{aligned}$$

And on the other hand,

$$\begin{aligned}
\Pr\left[b = 0\right] &\leq \frac{1}{12} \sum_{\vec{\mathbf{v}} \in \mathcal{R}^k} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \in S_{\vec{\mathbf{v}}'}} D_\sigma^{kn}(\vec{\mathbf{z}}') + \sum_{\vec{\mathbf{v}} \in \mathcal{R}^k} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \notin S_{\vec{\mathbf{v}}'}} D_{\vec{\mathbf{v}}',\sigma}^{kn}(\vec{\mathbf{z}}') \\
&\leq \frac{1}{12} + \frac{1}{12} \sum_{\|\vec{\mathbf{v}}\|_2 \leq T} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \notin S_{\vec{\mathbf{v}}'}} D_\sigma^{kn}(\vec{\mathbf{z}}') \\
&\quad + \frac{1}{12} \sum_{\|\vec{\mathbf{v}}\|_2 > T} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \notin S_{\vec{\mathbf{v}}'}} D_\sigma^{kn}(\vec{\mathbf{z}}') \\
&\leq \frac{1 + 2^{-102} + 2^{-101}}{12} = \frac{1}{12} + 2^{-104}
\end{aligned}$$

Therefore, we find for the statistical distance between the conditional distribution of $(\vec{\mathbf{v}}, \vec{\mathbf{z}})$ and the product distribution $\rho \times D_\sigma$,

$$\begin{aligned}
&\frac{1}{2} \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \sum_{\vec{\mathbf{z}}' \in \mathcal{R}^k} \left|\Pr\left[\vec{\mathbf{v}} = \vec{\mathbf{v}}' \wedge \vec{\mathbf{z}} = \vec{\mathbf{z}}' \mid b = 0\right] - \rho(\vec{\mathbf{v}}') D_\sigma^{kn}(\vec{\mathbf{z}}')\right| \\
&= \frac{1}{2} \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \sum_{\vec{\mathbf{z}}' \in S_{\vec{\mathbf{v}}'}} \left|\frac{\rho(\vec{\mathbf{v}}') D_\sigma^{kn}(\vec{\mathbf{z}}')}{12 \Pr\left[b = 0\right]} - \rho(\vec{\mathbf{v}}') D_\sigma^{kn}(\vec{\mathbf{z}}')\right| \\
&\quad + \frac{1}{2} \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \sum_{\vec{\mathbf{z}}' \notin S_{\vec{\mathbf{v}}'}} \left|\frac{\rho(\vec{\mathbf{v}}') D_{\vec{\mathbf{v}}',\sigma}^{kn}(\vec{\mathbf{z}}')}{\Pr\left[b = 0\right]} - \rho(\vec{\mathbf{v}}') D_\sigma^{kn}(\vec{\mathbf{z}}')\right| \\
&\leq \frac{1}{2} \left|\frac{1}{12 \Pr\left[b = 0\right]} - 1\right| \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \in \mathcal{R}^k} D_\sigma^{kn}(\vec{\mathbf{z}}') \\
&\quad + \frac{1}{2} \sum_{\vec{\mathbf{v}}' \in \mathcal{R}^k} \rho(\vec{\mathbf{v}}') \sum_{\vec{\mathbf{z}}' \notin S_{\vec{\mathbf{v}}'}} D_\sigma^{kn}(\vec{\mathbf{z}}') \\
&\leq \frac{1}{2} \left|\frac{1}{12 \Pr\left[b = 0\right]} - 1\right| + 3 \cdot 2^{-103} \leq 2^{-100}
\end{aligned}$$

□

2.5 Lattice Problems

We will use the commitment scheme from [BDL+18] whose security can be based on variants of the following two standard lattice problems.

Definition 2.5. *The* Ring Short Integer Solution *problem* $\mathrm{RSIS}_{k,B}$ *with parameters* $k \geq 1$ *and* $B > 0$ *is solved by finding a short, non-zero vector* $\vec{\mathbf{s}} \in \mathcal{R}^{k+1}$ *such that* $(\mathbf{1}, \vec{\mathbf{a}}^T) \cdot \vec{\mathbf{s}} = \mathbf{0}$ *over* $\mathcal{R}_q$*. We say that an algorithm* $\mathcal{A}$ *has advantage* ε *in solving the* $\mathrm{RSIS}_{k,B}$ *problem if*

$$\Pr\left[\|\vec{\mathbf{s}}\|_2 \leq B \ \wedge\ (\mathbf{1}, \vec{\mathbf{a}}^T) \cdot \vec{\mathbf{s}} = \mathbf{0} \ \wedge\ \vec{\mathbf{s}} \neq \vec{\mathbf{0}}^{k+1} \,\middle|\, \vec{\mathbf{a}} \xleftarrow{\$} \mathcal{R}_q^k;\ \vec{\mathbf{s}} \leftarrow \mathcal{A}(\vec{\mathbf{a}})\right] \geq \varepsilon$$

Definition 2.6. *In the* Ring Learning with Errors *problem* RLWE_m *with parameter* $m \geq 1$*, an adversary* $\mathcal{A}$ *tries to distinguish* $(\vec{\mathbf{a}}, \vec{\mathbf{b}}) \xleftarrow{\$} \mathcal{R}_q^m \times \mathcal{R}_q^m$ *from* $(\vec{\mathbf{a}}, \vec{\mathbf{a}}\mathbf{s} + \vec{\mathbf{e}})$ *with* $\vec{\mathbf{a}} \xleftarrow{\$} \mathcal{R}_q^m$ *and secret short* $\mathbf{s} \xleftarrow{\$} \beta^n$*,* $\vec{\mathbf{e}} \xleftarrow{\$} \beta^{mn}$*. We say that an algorithm* $\mathcal{A}$ *has advantage* ε *in solving the* RLWE_m *problem if*

$$\begin{aligned}\Bigg| \Pr\left[b = 1 \,\middle|\, \vec{\mathbf{a}} \xleftarrow{\$} \mathcal{R}_q^m \ ;\ \mathbf{s} \xleftarrow{\$} \beta^n \ ;\ \vec{\mathbf{e}} \xleftarrow{\$} \beta^{mn} \ ;\ b \leftarrow \mathcal{A}(\vec{\mathbf{a}}, \vec{\mathbf{a}}\mathbf{s} + \vec{\mathbf{e}})\right] \\ - \Pr\left[b = 1 \,\middle|\, \vec{\mathbf{a}} \xleftarrow{\$} \mathcal{R}_q^m \ ;\ \vec{\mathbf{b}} \xleftarrow{\$} \mathcal{R}_q^m \ ;\ b \leftarrow \mathcal{A}(\vec{\mathbf{a}}, \vec{\mathbf{b}})\right]\Bigg| \geq \varepsilon\end{aligned}$$

2.6 Commitment Scheme

A commitment scheme consists of a triple of algorithms (KeyGen, Com, Open) which work as follows.

$\mathrm{KeyGen}(1^\lambda) \rightarrow \mathrm{pp}$ is a probabilistic polynomial-time algorithm that produces the public parameters pp for the commitment scheme, defines the message space M, randomness space R and commitment space C, and implicitly includes the security parameter λ.

$\mathrm{Com}(\mathrm{pp}, m) \rightarrow (c, r)$ is a probabilistic polynomial-time algorithm that takes a message m and the public parameters pp as input and produces a commitment $c \in C$ and some randomness $r \in R$ used to compute and open c.

$\mathrm{Open}(\mathrm{pp}, m, c, r) \rightarrow b$ is a deterministic polynomial-time algorithm that takes the public parameters pp, a message m, randomness r, and a commitment c as input, and produces a bit $b \in \{0, 1\}$ as output.

A commitment scheme should be *hiding* and *binding*.

Definition 2.7 (Hiding). *A commitment scheme is* ε*-hiding if for all algorithms* $\mathcal{A}$

$$\Pr\left[b = b' \,\middle|\, \begin{array}{c} \mathrm{pp} \leftarrow \mathrm{KeyGen}(1^\lambda),\ (m_0, m_1) \leftarrow \mathcal{A}(\mathrm{pp}) \\ b \leftarrow \{0,1\},\ (c, r) \leftarrow \mathrm{Com}(\mathrm{pp}, m_b) \\ b' \leftarrow \mathcal{A}(\mathrm{pp}, c) \end{array}\right] < \varepsilon$$

where the probability is taken over the randomness of KeyGen*,* Com *and* $\mathcal{A}$*.*

Definition 2.8 (Binding). *A commitment scheme is ε-binding if for all algorithms $\mathcal{A}$*

$$\Pr\left[\begin{array}{c} m \neq m' \textit{ and} \\ \mathrm{Open}(\mathrm{pp}, m, c, r) = \mathrm{Open}(\mathrm{pp}, m', c, r) \end{array} \middle| \begin{array}{c} \mathrm{pp} \leftarrow \mathrm{KeyGen}(1^\lambda), \\ (m, m', r, r', c) \leftarrow \mathcal{A}(\mathrm{pp}) \end{array}\right] < \varepsilon$$

where the probability is taken over the randomness of KeyGen *and $\mathcal{A}$.*

If we restrict the algorithms $\mathcal{A}$ to probabilistic polynomial time algorithms in the definition of the hiding or binding properties, then we say that the property is computational. If we allow for arbitrarily powerful algorithms, then the property is statistical.

In our protocol, we use a variant of the commitment scheme from [BDL$^+$18] which splits the message space into different components. This allows useful manipulations on the different components as part of our zero-knowledge proof.

We will create public parameters that can be used to commit to messages $\vec{\mathbf{m}} \in \mathcal{R}_q^4$. Define the matrix $\mathbf{B} \in \mathcal{R}_q^{5\times 6}$ (with row vectors $\vec{\mathbf{b}}_1^T, \ldots, \vec{\mathbf{b}}_5^T$).

$$\mathbf{B} = \begin{pmatrix} \vec{\mathbf{b}}_1^T \\ \vec{\mathbf{b}}_2^T \\ \vec{\mathbf{b}}_3^T \\ \vec{\mathbf{b}}_4^T \\ \vec{\mathbf{b}}_5^T \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{b}_{1,2} & \mathbf{b}_{1,3} & \mathbf{b}_{1,4} & \mathbf{b}_{1,5} & \mathbf{b}_{1,6} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{b}_{2,6} \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{b}_{3,6} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{b}_{4,6} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{b}_{5,6} \end{pmatrix}$$

where the polynomials $\mathbf{b}_{i,j} \in \mathcal{R}_q$ are chosen uniformly at random.

To commit to $\vec{\mathbf{m}} = (\mathbf{m}_2, \mathbf{m}_3, \mathbf{m}_4, \mathbf{m}_5)^T \in \mathcal{R}_q^4$, we choose a random polynomial vector $\vec{\mathbf{r}} \xleftarrow{\$} \beta^{6n}$ from the error distribution and output the commitment

$$\mathrm{Com}(\vec{\mathbf{m}}; \vec{\mathbf{r}}) = \vec{\mathbf{t}} = \begin{pmatrix} \mathbf{t}_1 \\ \mathbf{t}_2 \\ \mathbf{t}_3 \\ \mathbf{t}_4 \\ \mathbf{t}_5 \end{pmatrix} = \mathbf{B} \cdot \vec{\mathbf{r}} + \begin{pmatrix} \mathbf{0} \\ \mathbf{m}_2 \\ \mathbf{m}_3 \\ \mathbf{m}_4 \\ \mathbf{m}_5 \end{pmatrix}$$

A valid (relaxed) opening of such a commitment $\vec{\mathbf{t}}$ consists of a message vector $\vec{\mathbf{m}} \in \mathcal{R}_q^4$, a short vector $\vec{\mathbf{r}} \in \mathcal{R}_q^6$ with $\|\vec{\mathbf{r}}\|_2 \leq 2B$ and a challenge difference $\bar{\mathbf{f}} \in \bar{\mathcal{C}}$ or $\bar{\mathbf{f}} = \mathbf{1}$. The verifier checks that

$$\bar{\mathbf{f}}\vec{\mathbf{t}} = \mathbf{B}\vec{\mathbf{r}} + \bar{\mathbf{f}} \begin{pmatrix} \mathbf{0} \\ \vec{\mathbf{m}} \end{pmatrix}$$

and that $\|\vec{\mathbf{r}}\|_2 \leq 2B$.

Remark. Although the commitment scheme allows relaxed openings to a multiple of the original commitment, it is used incidentally in our protocol. We would like to stress that our zero-knowledge proof shows that the prover knows an *exact* solution to a system of linear equations.

Lemma 2.9. *C.f. [BDL$^+$18, Lemma 6] For every algorithm $\mathcal{A}$ that has advantage ε in breaking the* hiding *property of the commitment scheme, there exists another algorithm $\mathcal{A}'$ that runs in the same time and has advantage ε in distinguishing* RLWE_5*.*

Lemma 2.10. *C.f. [BDL$^+$18, Lemma 7] For every algorithm $\mathcal{A}$ that succeeds with probability ε in breaking the* binding *property of the commitment scheme, there exists another algorithm $\mathcal{A}'$ that runs in the same time and solves* $\mathrm{RSIS}_{5,8B}$ *with probability ε.*

3 The Main Protocol

We want to prove knowledge of a short integer vector $\vec{s}$ that is a solution to a linear equation $A\vec{s} = \vec{u}$ over $\mathbb{Z}_q$ with public matrix A and vector $\vec{u}$. We now describe our protocol for this task. In the introduction, we made various simplifications to make the key ideas easier to understand. We now give more precise details.

Concretely, let A be an $m \times n$ matrix over $\mathbb{Z}_q$ and $\vec{s}$ have coefficients in $\{-S, \ldots, S\}$. First we transform the equation to an equation with vector $\vec{s}'$ having coefficients in $\{0, 1, 2\}$. The easiest way to achieve this is simply to write the coefficients $s_1, \ldots, s_n$ of $\vec{s}$ as

$$s_i = s'_{i,0} + 3s'_{i,1} + \cdots + 3^{r-1}s'_{i,r-1} - 3^r s'_{i,r} = \vec{g}^T \vec{s}'_i$$

where $r = \lfloor \log_3 S + 1 \rfloor$, the coefficients $s'_{i,j}$ of $\vec{s}'_i$ are in $\{0, 1, 2\}$, and $\vec{g}^T$ is the gadget row vector $\vec{g}^T = (1, 3, \ldots, 3^{r-1}, -3^r)^T$. Then we have $\vec{s} = (I_n \otimes \vec{g}^T)\vec{s}'$ and hence

$$A'\vec{s}' = A(I_n \otimes \vec{g}^T)\vec{s}' = A\vec{s} = \vec{u}$$

when we write $A' = A(I_n \otimes \vec{g}^T) \in \mathbb{Z}_q^{m \times rn}$.

As discussed in the introduction, our high level strategy is for the prover to send the masked opening $\mathbf{z} = \mathbf{y} + \mathbf{cs}$, and use commitments to $\mathbf{y}$ and $\mathbf{s}$ to show that $\mathbf{z}$ was correctly formed. Then, the prover and verifier can use this masked opening $\mathbf{z}$, along with some extra commitments, to prove that the following polynomial expression is a polynomial of degree 2, and not degree 3.

$$\mathbf{z}(\mathbf{z}-\mathbf{c})(\mathbf{z}-2\mathbf{c}) = \mathbf{y}^3 + 3\mathbf{y}^2(\mathbf{s}-1)\mathbf{c} + \mathbf{y}(3\mathbf{s}^2 - 6\mathbf{s} + 2)\mathbf{c}^2 + \mathbf{s}(\mathbf{s}-1)(\mathbf{s}-2)\mathbf{c}^3. \quad (9)$$

The simplest way to do this would be to make 5 commitments, and check equation 9 in committed form, i.e.

$$\begin{aligned} \mathbf{t}_1 &= \mathrm{Com}(\mathbf{y}) & \mathbf{t}_2 &= \mathrm{Com}(\mathbf{s}) \\ \mathbf{t}_3 &= \mathrm{Com}(\mathbf{y}^3) & \mathbf{t}_4 &= \mathrm{Com}(3\mathbf{y}^2(\mathbf{s}-1)) \\ \mathbf{t}_5 &= \mathrm{Com}(\mathbf{y}(3\mathbf{s}^2 - 6\mathbf{s} + 2)) \end{aligned}$$

$$\mathrm{Com}(\mathbf{z}) \stackrel{?}{=} \mathbf{t}_1 + \mathbf{c}\mathbf{t}_2 \qquad \mathrm{Com}(\mathbf{z}(\mathbf{z}-\mathbf{c})(\mathbf{z}-2\mathbf{c})) \stackrel{?}{=} \mathbf{t}_3 + \mathbf{t}_4\mathbf{c} + \mathbf{t}_5\mathbf{c}^2$$

In our protocol, we optimize the procedure so that the prover needs only commit to 4 polynomials, instead of 5. Given that we will send $\mathbf{z}$, and check that it was correctly computed as $\mathbf{y}+\mathbf{c}\mathbf{s}$, we can use $\mathbf{z}$ to help evaluate the left-hand side of equation 9, and use the commitment to $\mathbf{s}$ to simplify the right hand side. After these optimisations, our proof uses the following alternative expression.

$$(\mathbf{z}-\mathbf{c})(\mathbf{z}-2\mathbf{c})\mathbf{s} = \mathbf{z}\mathbf{y}(2\mathbf{s}-3) - \mathbf{y}^2(\mathbf{s}-3) + \mathbf{s}(\mathbf{s}-1)(\mathbf{s}-2)\mathbf{c}^2. \tag{10}$$

Our protocol is based around making 4 commitments and checking (10) in committed form.

$$\begin{aligned} \mathbf{t}_2 &= \mathrm{Com}(\mathbf{y}) & \mathbf{t}_3 &= \mathrm{Com}(\mathbf{s}) \\ \mathbf{t}_4 &= \mathrm{Com}(\mathbf{y}(2\mathbf{s}-3) & \mathbf{t}_5 &= \mathrm{Com}(\mathbf{y}^2(\mathbf{s}-3)) \end{aligned}$$

$$\mathrm{Com}(\mathbf{z}) \overset{?}{=} \mathbf{t}_1 + \mathbf{c}\mathbf{t}_2 \qquad \mathrm{Com}(\mathbf{0}) \overset{?}{=} (\mathbf{z}-\mathbf{c})(\mathbf{z}-2\mathbf{c})\mathbf{t}_3 - \mathbf{z}\mathbf{t}_4 + \mathbf{t}_5$$

This is still a slight simplification of what takes place in the protocol, as the commitment scheme given in Section 2.6 will actually commit to all four messages at the same time, but fortunately, the commitment scheme nevertheless allows us to manipulate the individual components and check the given equations.

The protocol uses two challenges, $c \in \mathbb{Z}_q$, and $\mathbf{f} \in \mathcal{C}$. The first challenge c is used to embed $\mathbf{s}$ into a masked value, which is used to show that $A\hat{\mathbf{s}} = \vec{u}$. The second challenge $\mathbf{f}$ is used to embed the commitment randomness into a masked value, which will help us to check important equations in committed form. Of course, we cannot allow the verifier to see the commitment randomness without any random masking, as this would leak information about the committed secret $\mathbf{s}$. This leads to two more extra terms when checking the equations, which are given by $\mathbf{x}_1$ and $\mathbf{x}_2$ in the protocol.

We now discuss each step of the protocol, describing the actions of the prover and verifier.

In the first move, the prover samples a random masking value $\mathbf{y}$, which will later be used to hide the secret $\mathbf{s}$. Then they sample the randomness $\vec{\mathbf{r}}$ used to commit to the four ring elements given in the polynomial equations above. They also fix the random masking value $\hat{\mathbf{y}}$ by computing $\vec{w} = A\hat{\mathbf{y}}$. The value $\hat{\mathbf{y}}$ will be used to help verify that $A\hat{\mathbf{s}} = \vec{u}$ later on. The prover sends these to the verifier.

Next, the verifier sends a random challenge $c \in \mathbb{Z}_q$ to the prover. As we have seen, the challenge c is used to embed the secret $\mathbf{s}$ into $\mathbf{z}$. The prover samples a new masking value $\vec{\mathbf{y}}'$ which will be used to hide the commitment randomness $\vec{\mathbf{r}}$. They also compute the values $\mathbf{x}_1$ and $\mathbf{x}_2$, which will later allow the verifier to check equation (10) in committed form using a masked version of $\vec{\mathbf{r}}$. They send these values to the verifier.

Next, the verifier sends a random challenge $\mathbf{f} \in \mathcal{C}$ to the prover. The prover computes $\vec{\mathbf{z}}'$, a masked version of the commitment randomness, and applies a rejection sampling algorithm to make sure that this value does not leak any information about $\mathbf{r}$.

Finally, the verifier checks a blinded version of the equation $A\hat{\mathbf{s}} = \vec{u}$, and blinded versions of the equation (10) and $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ written in committed form, using the extra terms which the prover sent earlier.

The full protocol is given in Figure 1.

3.1 Security Analysis

Theorem 3.1. *The protocol in Figure 1 is complete, computational honest verifier zero-knowledge if* RLWE_5 *is hard and generalized special sound if* $\mathrm{RSIS}_{5,8B}$ *is hard.*

More precisely, the honest prover $\mathcal{P}$ convinces the honest verifier $\mathcal{V}$ with probability $\varepsilon \approx 1/12$.

For zero-knowledge, there exists a simulator $\mathcal{S}$, that, without access to secret information, outputs a simulation of a non-aborting transcript of the protocol between $\mathcal{P}$ and $\mathcal{V}$. Then for every algorithm $\mathcal{A}$ that has advantage ε in distinguishing the simulated transcript from an actual transcript, there is an algorithm $\mathcal{A}'$ with the same running time that has advantage $\varepsilon - 2^{-100}$ in distinguishing RLWE_5.

For knowledge-soundness, there is an extractor $\mathcal{E}$ with the following properties. When given rewindable black-box access to a deterministic prover $\mathcal{P}^*$ that convinces $\mathcal{V}$ with probability $\varepsilon > 2/q + 2/l$, $\mathcal{E}$ either outputs a solution $\vec{s}^* \in \{0,1,2\}^n$ to $A\vec{s}^* = \vec{u}$, or a $\mathrm{RSIS}_{5,8B}$ *solution for $\vec{\mathbf{b}}_1^T$ in expected time at most $144/(\varepsilon - 2/q - 2/l)$ when running $\mathcal{P}^*$ once is assumed to take unit time.*

Notice that we only require the simulator $\mathcal{S}$ to simulate non-aborting transcripts, i.e. the interaction between $\mathcal{P}$ and $\mathcal{V}$ conditioned on the prover not aborting. The rationale behind this is that in the non-interactive version that is relevant in practice one never gets to see the aborting proofs. In any case, there is a standard technique which makes it possible to simulate the aborting transcripts too, whereby the prover commits to the binding quantities $\mathbf{w}'$, $\mathbf{x}_1$ and $\mathbf{x}_2$ and only opens the commitments if he does not abort.

Proof. Completeness. It follows directly from Lemma 2.4 that the honest prover $\mathcal{P}$ does not abort with probability at least $1/12 - 2^{-100}$. Moreover, in this case the distribution of the vector $\vec{\mathbf{z}}'$ sent by $\mathcal{P}$ has statistical distance at most 2^{-100} from D_σ^{6n}, and Lemma 2.3 implies that the bound $\|\vec{\mathbf{z}}'\|_2 \leq B = \sigma\sqrt{12n}$ is true with probability at least $1 - 2^{-0.66n} - 2^{-100}$. It is easy to see that all of the other verification equations are always true for the messages sent by $\mathcal{P}$. Therefore, the honest prover convinces the honest verifier with probability $\varepsilon \approx 1/12$.

Soundness. The extractor $\mathcal{E}$ needs to obtain accepting transcripts from $\mathcal{P}^*$ for 3 different first challenges $c_1, c_2, c_3 \in \mathbb{Z}_q$. Moreover, for each of the 3 c_i, $\mathcal{E}$ needs 2 accepting transcripts with first challenge c_i and two different second challenges $\mathbf{f}_{i,1} \neq \mathbf{f}_{i,2}$. So in total $\mathcal{E}$ needs 6 transcripts. To this end, he runs $\mathcal{P}^*$, sends uniformly random challenges c_1 and $\mathbf{f}_{1,1}$ and repeats until he obtains a first accepting transcript. This takes expected time $1/\varepsilon$. Then, by a standard heavy rows argument, with probability at least 1/2, the probability of obtaining an

Prover $\mathcal{P}$		Verifier $\mathcal{V}$
Inputs:		
$A \in \mathbb{Z}_q^{m\times n}, \hat{\mathbf{s}} \in \{0,1,2\}^n$		$A, \vec{u}, \vec{\mathbf{b}}_1, \ldots, \vec{\mathbf{b}}_5$
$\vec{u} = A\hat{\mathbf{s}} \in \mathbb{Z}_q^m$		
$\vec{\mathbf{b}}_1, \ldots, \vec{\mathbf{b}}_5 \in \mathcal{R}_q^6$		
$\mathbf{y} \xleftarrow{\$} \mathcal{R}_q$		
$\vec{\mathbf{r}} \xleftarrow{\$} \beta^{6n}$		
$\vec{\mathbf{t}} = \begin{pmatrix} \vec{\mathbf{b}}_1^T \\ \vec{\mathbf{b}}_2^T \\ \vec{\mathbf{b}}_3^T \\ \vec{\mathbf{b}}_4^T \\ \vec{\mathbf{b}}_5^T \end{pmatrix} \vec{\mathbf{r}} + \begin{pmatrix} \mathbf{0} \\ \mathbf{y} \\ \mathbf{s} \\ \mathbf{y}(2\mathbf{s}-3) \\ \mathbf{y}^2(\mathbf{s}-3) \end{pmatrix}$		
$\vec{w} = A\hat{\mathbf{y}}$	$\xrightarrow{\vec{\mathbf{t}}, \vec{w}}$	
	$\xleftarrow{c}$	$c \xleftarrow{\$} \mathbb{Z}_q$
$\mathbf{z} = \mathbf{y} + c\mathbf{s}$		
$\vec{\mathbf{y}}' \xleftarrow{\$} D_\sigma^{6n}$		
$\mathbf{w}' = \vec{\mathbf{b}}_1^T \vec{\mathbf{y}}'$		
$\mathbf{x}_1 = \left(\vec{\mathbf{b}}_2^T + c\vec{\mathbf{b}}_3^T\right)\vec{\mathbf{y}}'$		
$\mathbf{x}_2 = \left((\mathbf{z}-c)(\mathbf{z}-2c)\vec{\mathbf{b}}_3^T - \mathbf{z}\vec{\mathbf{b}}_4^T + \vec{\mathbf{b}}_5^T\right)\vec{\mathbf{y}}'$	$\xrightarrow{\mathbf{z}, \mathbf{w}', \mathbf{x}_1, \mathbf{x}_2}$	
	$\xleftarrow{\mathbf{f}}$	$\mathbf{f} \xleftarrow{\$} \mathcal{C}$
$\vec{\mathbf{z}}' = \vec{\mathbf{y}}' + \mathbf{f}\vec{\mathbf{r}}$		
If Rej $\left(\vec{\mathbf{z}}', \mathbf{f}\vec{\mathbf{r}}, \sigma\right) = 1$, abort	$\xrightarrow{\vec{\mathbf{z}}'}$	
		$\left\Vert\vec{\mathbf{z}}'\right\Vert \leq B = \sigma\sqrt{12n}$
		$A\hat{\mathbf{z}} \overset{?}{=} \vec{w} + c\vec{u}$
		$\vec{\mathbf{b}}_1^T \vec{\mathbf{z}}' \overset{?}{=} \mathbf{w}' + \mathbf{f}\mathbf{t}_1$
		$\left(\vec{\mathbf{b}}_2^T + c\vec{\mathbf{b}}_3^T\right)\vec{\mathbf{z}}' + \mathbf{f}\mathbf{z}$
		$\overset{?}{=} \mathbf{x}_1 + \mathbf{f}(\mathbf{t}_2 + c\mathbf{t}_3)$
		$\left((\mathbf{z}-c)(\mathbf{z}-2c)\vec{\mathbf{b}}_3^T - \mathbf{z}\vec{\mathbf{b}}_4^T + \vec{\mathbf{b}}_5^T\right)\vec{\mathbf{z}}'$
		$\overset{?}{=} \mathbf{x}_2 + \mathbf{f}((\mathbf{z}-c)(\mathbf{z}-2c)\mathbf{t}_3 - \mathbf{z}\mathbf{t}_4 + \mathbf{t}_5)$

Fig. 1. Lattice-based proof of knowledge of a ternary solution to a linear equation over $\mathbb{Z}_q$.

accepting transcript conditioned fixing the first challenge c_1, but with uniformly random second challenge, is at least $\varepsilon/2$. So conditioned on c_1, the extractor obtains a second accepting transcript with challenges c_1 and $\mathbf{f}_{1,2} \neq \mathbf{f}_{1,1}$ with probability at least $\varepsilon/2 - 1/l$, and he succeeds in getting such a transcript in expected time at most $(\varepsilon/2 - 1/l)^{-1}$. For the third transcript he sends uniformly random $c_2 \neq c_1$ and $\mathbf{f}_{2,1}$ and succeeds in expected time at most $(\varepsilon - 1/q)^{-1}$. Then, using the heavy rows argument again, we can assume that the acceptance probability for fixed first challenge c_2 is at least $\varepsilon/2 - 1/(2q)$, which is true with probability at least $1/2$. Therefore, in conditioned expected time $(\varepsilon/2 - 1/(2q) - 1/l)^{-1}$, the extractor receives the 4-th transcript with challenges c_2 and $\mathbf{f}_{2,2} \neq \mathbf{f}_{2,1}$. Continuing in the same way, the last two transcripts are obtained in conditioned expected time at most $3/(\varepsilon - 2/q - 2/l)$.

In summary, with probability $1/8$, the total expected time needed to obtain the 6 transcripts is less than

$$T = \frac{9}{\varepsilon - 2/q - 2/l}.$$

With probability $7/8$ the extractor is not so lucky and might run for a long time or not terminate at all. We cope with this by limiting the runtime of $\mathcal{E}$ to $2T$. Then, by Markov's inequality, the extractor gets hold of the 6 accepting transcripts in time at most $2T$ with probability at least $1/16$. By restarting in case of failure we finally conclude that in expected time $16T$ the extractor indeed has the 6 accepting transcripts needed.

Let us now see how to use the transcripts. Let $\vec{\mathbf{z}}'_{i,j}$, $i = 1, 2, 3$, $j = 1, 2$, be the last messages from $\mathcal{P}^*$. Write $\bar{\vec{\mathbf{z}}}'_i = \vec{\mathbf{z}}'_{i,1} - \vec{\mathbf{z}}'_{i,2}$ and $\bar{\mathbf{f}}_i = \mathbf{f}_{i,1} - \mathbf{f}_{i,2}$ for the difference of these messages in the transcripts with the same first challenge and the difference of the corresponding second challenges, respectively. The verification equation $\vec{\mathbf{b}}_1^T \vec{\mathbf{z}}'_{i,j} = \mathbf{w}' + \mathbf{f}_{i,j}\mathbf{t}_1$ yields approximate solutions to the first equation of the commitment $\vec{\mathbf{t}}$ by subtracting,

$$\vec{\mathbf{b}}_1^T \bar{\vec{\mathbf{z}}}'_i = \bar{\mathbf{f}}_i \mathbf{t}_1.$$

Then we can compute openings $\mathbf{m}_2 = \mathbf{y}^*$, $\mathbf{m}_3 = \mathbf{s}^*$, $\mathbf{m}_4$ and $\mathbf{m}_5$ of $\vec{\mathbf{t}}$. For instance,

$$\mathbf{m}_k = \mathbf{t}_k - \vec{\mathbf{b}}_k^T \frac{\bar{\vec{\mathbf{z}}}'_1}{\bar{\mathbf{f}}_1}.$$

Note these openings are valid relaxed openings of our commitment scheme with $\|\bar{\vec{\mathbf{z}}}'_1\|_2 \leq 2B$. Therefore, when using $\bar{\vec{\mathbf{z}}}'_2$ and $\bar{\mathbf{f}}_2$ or $\bar{\vec{\mathbf{z}}}'_3$ and $\bar{\mathbf{f}}_3$ to compute openings we either get the same $\mathbf{m}_k$ or break the binding property of the commitment scheme. The latter would translate to a $\mathrm{RSIS}_{5,8B}$ solution; c.f. Lemma 2.10. Concretely, if

$$\mathbf{m}_k \neq \mathbf{m}'_k = \mathbf{t}_k - \vec{\mathbf{b}}_2^T \frac{\bar{\vec{\mathbf{z}}}'_2}{\bar{\mathbf{f}}_2},$$

then $\bar{\mathbf{f}}_2 \bar{\vec{\mathbf{z}}}'_1 - \bar{\mathbf{f}}_1 \bar{\vec{\mathbf{z}}}'_2 \neq 0$ and we get the $\mathrm{RSIS}_{5,8B}$ solution

$$\vec{\mathbf{b}}_1^T \left(\bar{\mathbf{f}}_2 \bar{\vec{\mathbf{z}}}'_1 - \bar{\mathbf{f}}_1 \bar{\vec{\mathbf{z}}}'_2 \right) = \mathbf{0}.$$

Here we have used the fact that $\|\vec{\mathbf{z}}'_i\|_2 \leq 2B$, which implies $\left\|\bar{\mathbf{f}}_i \vec{\mathbf{z}}'_i\right\|_2 \leq 2\,\|\vec{\mathbf{z}}'_i\|_2 \leq 4B$ and $\left\|\bar{\mathbf{f}}_2 \vec{\mathbf{z}}'_1 - \bar{\mathbf{f}}_1 \vec{\mathbf{z}}'_2\right\|_2 \leq 8B$.

Assume we did not break the commitment scheme and write $\mathbf{z}_i$ for the message $\mathbf{z}$ sent by the prover in the $(2i-1)$-th and $(2i)$-th transcript, which is equal in the two transcripts. Consider the verification equations

$$\left(\vec{\mathbf{b}}_2^T + c_i \vec{\mathbf{b}}_3^T\right) \vec{\mathbf{z}}'_{i,j} + \mathbf{f}_{i,j} \mathbf{z}_i = \mathbf{x}_{1,i} + \mathbf{f}_{i,j} \left(\mathbf{t}_2 + c_i \mathbf{t}_3\right).$$

Subtract the equations with $j = 1, 2$ and the same i to obtain

$$\left(\vec{\mathbf{b}}_2^T + c_i \vec{\mathbf{b}}_3^T\right) \frac{\vec{\mathbf{z}}'_i}{\bar{\mathbf{f}}_i} + \mathbf{z}_i = \mathbf{t}_2 + c_i \mathbf{t}_3.$$

Now substitute the opening

$$\mathbf{y}^* + c_i \mathbf{s}^* = \mathbf{m}_2 + c_i \mathbf{m}_3 = \mathbf{t}_2 + c_i \mathbf{t}_3 - \left(\vec{\mathbf{b}}_2^T + c_i \vec{\mathbf{b}}_3^T\right) \frac{\vec{\mathbf{z}}'_i}{\bar{\mathbf{f}}_i}$$

corresponding to $\mathbf{t}_2 + c_i \mathbf{t}_3$. This gives

$$\mathbf{z}_i = \mathbf{y}^* + c_i \mathbf{s}^*. \tag{11}$$

So we see that the messages $\mathbf{z}_i$ are of the expected form with constant polynomials $\mathbf{y}^*$ and $\mathbf{s}^*$ that are independent of the challenges c_i. Next from the verification equations

$$\begin{aligned}&\left((\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\vec{\mathbf{b}}_3^T - \mathbf{z}_i \vec{\mathbf{b}}_4^T + \vec{\mathbf{b}}_5^T\right) \vec{\mathbf{z}}'_{i,j} \\ &= \mathbf{x}_{2,i} + \mathbf{f}_{i,j} \left((\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\mathbf{t}_3 - \mathbf{z}_i \mathbf{t}_4 + \mathbf{t}_5\right)\end{aligned}$$

we find by using the opening $(\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\mathbf{s}^* - \mathbf{z}_i \mathbf{m}_4 + \mathbf{m}_5$ corresponding to $(\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\mathbf{t}_2 - \mathbf{z}_i \mathbf{t}_4 + \mathbf{t}_5$ and Equation (11),

$$\begin{aligned}&(\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\mathbf{s}^* - \mathbf{z}_i \mathbf{m}_4 + \mathbf{m}_5 \\ &= \left(\mathbf{y}^* + c_i(\mathbf{s}^* - 1)\right)\left(\mathbf{y}^* + c_i(\mathbf{s}^* - 2)\right)\mathbf{s}^* - \mathbf{y}^* \mathbf{m}_4 - c_i \mathbf{s}^* \mathbf{m}_4 + \mathbf{m}_5 \\ &= \left((\mathbf{y}^*)^2 \mathbf{s}^* - \mathbf{y}^* \mathbf{m}_4 + \mathbf{m}_5\right) + \left(\mathbf{y}^*(2\mathbf{s}^* - 3) - \mathbf{m}_4\right)\mathbf{s}^* c_i + (\mathbf{s}^* - 1)(\mathbf{s}^* - 2)\mathbf{s}^* c_i^2 \\ &= \mathbf{0}.\end{aligned}$$

So we have a polynomial of degree 2 over $\mathcal{R}_q$ that evaluates to zero at the 3 points c_1, c_2 and c_3. We can write this as a matrix-vector equation over $\mathcal{R}_q$,

$$\begin{pmatrix} 1 & c_1 & c_1^2 \\ 1 & c_2 & c_2^2 \\ 1 & c_3 & c_3^2 \end{pmatrix} \begin{pmatrix} (\mathbf{y}^*)^2 \mathbf{s}^* - \mathbf{y}^* \mathbf{m}_4 + \mathbf{m}_5 \\ \left(\mathbf{y}^*(2\mathbf{s}^* - 3) - \mathbf{m}_4\right)\mathbf{s}^* \\ (\mathbf{s}^* - 1)(\mathbf{s}^* - 2)\mathbf{s}^* \end{pmatrix} = \begin{pmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}.$$

The determinant of this well-known Vandermonde matrix is equal to $(c_1 - c_2)(c_1 - c_3)(c_2 - c_3) \in \mathbb{Z}_q^\times \subset \mathcal{R}_q^\times$ and hence the matrix is invertible over $\mathcal{R}_q$. Therefore, $\mathbf{s}^*(\mathbf{s}^* - 1)(\mathbf{s}^* - 2) = \mathbf{0}$. Applying the NTT to this last equation implies

$$\hat{\mathbf{s}}^* \circ (\hat{\mathbf{s}}^* - \vec{1}) \circ (\hat{\mathbf{s}}^* - \vec{2}) = \vec{0}$$

in $\mathbb{Z}_q^n$. So the coefficients of $\hat{\mathbf{s}}^*$ are all in $\{0,1,2\}$. Finally, by subtracting copies of the second verification equation from on another, we get $A(\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2) = (c_1 - c_2)\vec{u}$. But we know that

$$\frac{\hat{\mathbf{z}}_1 - \hat{\mathbf{z}}_2}{c_1 - c_2} = \hat{\mathbf{s}}^*$$

has only coefficients in $\{0,1,2\}$ and is the solution to the linear equation with matrix A that we wanted to find.

Zero-Knowledge. We can simulate a non-aborting transcript between the honest prover and the honest verifier in the following way. First, note that in such a transcript $\mathbf{z} = \mathbf{y} + c\mathbf{s}$ is uniformly random because the honest prover samples $\mathbf{y}$ uniformly at random. Moreover, $\vec{\mathbf{z}}'$ is statistically close to D_σ^{6n} by Lemma 2.4. So the simulator can just pick $\mathbf{z} \xleftarrow{\$} \mathcal{R}_q$ and $\vec{\mathbf{z}}' \leftarrow D_\sigma^{6n}$. Next, by Lemma 2.4 again, we know that $\mathbf{f}\vec{\mathbf{r}}$ is independent of $\vec{\mathbf{z}}'$, and hence that $\mathbf{f}$ is independent of $\vec{\mathbf{z}}'$. The two challenges c and $\mathbf{f}$ are uniformly random since the honest verifier chooses them in this way. Therefore, the simulator picks $c \xleftarrow{\$} \mathbb{Z}_q$ and $\mathbf{f} \xleftarrow{\$} \mathcal{C}$. The commitment $\vec{\mathbf{t}}$ is computationally indistinguishable from a dummy commitment if RLWE$_5$ is hard (c.f. Lemma 2.9. In fact, the construction of the commitment scheme is such that $\vec{\mathbf{t}}$ contains an additive term that is precisely a RLWE$_5$ sample. So the simulator can just take a uniformly random $\vec{\mathbf{t}} \xleftarrow{\$} \mathcal{R}_q^5$. Now, in an honest transcript, the remaining messages $\vec{w}$, $\mathbf{w}'$, $\mathbf{x}_1$ and $\mathbf{x}_2$ are all uniquely determined by the verification equations because of completeness. We see that if the simulator computes these messages so that the verification equations become true, then the resulting transcript is indistinguishable from the honest transcript. More precisely, a simulated transcript has statistical distance at most 2^{-100} from a distribution which differs from the actual transcripts only in that $\vec{\mathbf{t}}$ is distributed differently. Therefore, if there is an algorithm $\mathcal{A}$ that has advantage ε in distinguishing a simulated transcript from an actual transcript, then this algorithm must be able to distinguish RLWE$_5$ samples from random with advantage $\varepsilon - 2^{-100}$.

3.2 Repeating the Proof

For the moduli q and the dimensions $n = \varphi(l)$ that occur when proving equations from lattice-based cryptographic schemes, our proof does not have sufficiently low soundness error as $\varepsilon_0 = 2/q + 2/l$ will be much larger than 2^{-128}. Therefore the proof needs to be repeated multiple times. If for t repetitions, every single repetition succeeds with probability $\varepsilon > (\varepsilon_0)^t$, then it cannot be that each of them has success probability less than ε_0. Otherwise we would have $\varepsilon < (\varepsilon_0)^t$. So one of the proofs will be extractable. Since $l < q$, the number of repetitions necessary is determined by l. If l is considerably smaller than q then it is worth repeating the lower half of the proof with challenge $\mathbf{f}$ a couple of times for a single execution of the upper half with challenge c. The lower part of the proof demonstrates knowledge of an opening of the commitment $\vec{\mathbf{t}}$. Recall that the extractor needs successful transcripts of the full protocol with the same challenge c and two different challenges $\mathbf{f}_1$ and $\mathbf{f}_2$. He gets these by obtaining a first successful transcript and then running the prover with fixed challenge c and random $\mathbf{f}_2 \neq \mathbf{f}_1$. So if the lower part is repeated twice then the first successful execution will have challenges c and $(\mathbf{f}_1, \mathbf{f}_1')$ and the extractor can choose fresh challenges $(\mathbf{f}_2, \mathbf{f}_2')$ from the set $\mathcal{C}^2$ of size l^2 and only needs that one of the two $\mathbf{f}_2$ and $\mathbf{f}_2'$ is different from the corresponding $\mathbf{f}_1$ or $\mathbf{f}_1'$, i.e. that $(\mathbf{f}_1, \mathbf{f}_1') \neq (\mathbf{f}_2, \mathbf{f}_2')$. Hence we see that the proof with two repetitions of the lower part will have soundness error $\varepsilon_0 = 2/q + 2/l^2$.

3.3 Non-Interactive Proof

In practice, the interactive protocol given in Figure 1 is usually converted to a non-interactive protocol by using the Fiat-Shamir heuristic. So the two challenges $c \in \mathbb{Z}_q$ and $\mathbf{f} \in \mathcal{C}$ are computed by the prover from a hash of public information and all previous messages, where the hash function is modelled as a random oracle. Since our protocol does not have sufficiently low soundness error, it needs to be repeated multiple times to be secure. In the non-interactive version this is done by computing multiple proofs in parallel where all messages of all parallel proofs are put into the hash function to derive the various challenges for the parallel executions of the protocol. Here we allow for the lower half to be repeated multiple times for each repetition of the upper half, as explained in Section 3.2. Concretely, we repeat the upper half t times with challenges c_i, $i = 1, \ldots, t$, and for each of the repetitions we perform the lower part t' times with challenges $\mathbf{f}_j$, $j = t'(i-1)+1, \ldots, t'(i-1)+t'$.

If the rejection sampling on the vectors $\vec{\mathbf{z}}_j'$ in the parallel proofs was performed individually, the whole proof would need to be restarted if only one of the $\vec{\mathbf{z}}_j'$ was rejected, which would happen with probability about $1 - (1/12)^{tt'}$. Therefore, the runtime of the non-interactive prover would be very long. Instead, we mask the concatenation of the t secret vectors $\mathbf{f}_j \vec{\mathbf{r}}_i$ for $i = 1, \ldots, t$ and $j = (i-1)t' + k$ with the same k by sampling the masking vectors $\vec{\mathbf{y}}_j'$ with a standard deviation that is equal to $5T'$, where T' is a bound on the concatenation of those secret vectors. Then we do rejection sampling on all of the corresponding $\vec{\mathbf{z}}_j'$ at once. The downside of this is that the bound T' and hence the standard deviation of the discrete Gaussian distribution for the $\vec{\mathbf{y}}_j'$ increases by a factor of $\sqrt{t}$. We could also perform rejection sampling on all of the tt' masking vectors at once with another increase of the standard deviation by a factor of $\sqrt{t'}$. But usually t' is at most two or three and hence the increase in prover time from not including this optimisation this does not pose a problem.

As another improvement we make use of the fact that it is not necessary to recommit to the constant secret polynomial $\mathbf{s}$ in all of the parallel executions of the proof. Instead it is actually enough to recommit to all of the other messages by sampling fresh RLWE errors $\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_4, \mathbf{r}_5$ for them and the first row of the commitment. In this way, we actually give out $1 + 4t$ RLWE samples and require RLWE_{1+4t} to be a hard problem. The complete non-interactive prover algorithm is given in Figure 2 and the corresponding verifier in Figure 3.

Apart from the improvements described, the non-interactive algorithms also make use of the standard technique of sending the challenges c_i and $\mathbf{f}_j$ instead of the large binding quantities $\vec{w}_i$, $\mathbf{w}'_j$, $\mathbf{x}_{1,j}$ and $\mathbf{x}_{2,j}$. The verifier computes these as the only missing terms in the verification equations, which allows him to check the challenges. The functions $\mathrm{G}(\cdot)$ and $\mathrm{H}(\cdot)$ are the two hash functions, modelled as random oracles, for sampling the challenges.

Algorithm 2 Non-Interactive Prover

```
 1: Input: $A, \vec{u}, \hat{\mathbf{s}}$
 2: Output: $(\vec{\mathbf{t}}_i)_{i\in[t]}, (c_i)_{i\in[t]}, (\mathbf{z}_i)_{i\in[t]}, (\mathbf{f}_j)_{j\in[tt']}, (\vec{\mathbf{z}}'_j)_{j\in[tt']}$
 3: $\mathbf{r}_3, \mathbf{r}_6 \xleftarrow{\$} \beta^n$
 4: $\mathbf{t}_3 = \mathbf{b}_{3,6}\mathbf{r}_6 + \mathbf{r}_3 + \mathbf{s}$
 5: for $i = 1, \ldots, t$ do
 6:     $\mathbf{y}_i \xleftarrow{\$} \mathcal{R}_q$
 7:     $\vec{w}_i = A\hat{\mathbf{y}}_i$
 8:     $\mathbf{r}_{i,1}, \mathbf{r}_{i,2}, \mathbf{r}_{i,4}, \mathbf{r}_{i,5} \xleftarrow{\$} \beta^n$
 9:     $\vec{\mathbf{r}}_i = (\mathbf{r}_{i,1}, \mathbf{r}_{i,2}, \mathbf{r}_3, \mathbf{r}_{i,4}, \mathbf{r}_{i,5}, \mathbf{r}_6)^T$
10:     $\mathbf{t}_{i,1} = \vec{\mathbf{b}}_1^T\vec{\mathbf{r}}_i$
11:     $\mathbf{t}_{i,2} = \vec{\mathbf{b}}_2^T\vec{\mathbf{r}}_i + \mathbf{y}_i$
12:     $\mathbf{t}_{i,4} = \vec{\mathbf{b}}_4^T\vec{\mathbf{r}}_i + \mathbf{y}_i(2\mathbf{s} - 3)$
13:     $\mathbf{t}_{i,5} = \vec{\mathbf{b}}_5^T\vec{\mathbf{r}}_i + \mathbf{y}_i^2(\mathbf{s} - 3)$
14:     $\vec{\mathbf{t}}_i = (\mathbf{t}_{i,1}, \mathbf{t}_{i,2}, \mathbf{t}_3, \mathbf{t}_{i,4}, \mathbf{t}_{i,5})^T$
15: end for
16: $(c_i)_{i\in[t]} = \mathrm{H}(A, \vec{u}, (\vec{\mathbf{t}}_i)_i, (\vec{w}_i)_i)$
17: for $i = 1, \ldots, t$ do
18:     $\mathbf{z}_i = \mathbf{y}_i + c_i\mathbf{s}$
19:     for $j = (i-1)t' + 1, \ldots, (i-1)t' + t'$ do
20:         $\vec{\mathbf{y}}'_j \xleftarrow{\$} D_\sigma^{6n}$
21:         $\mathbf{w}'_j = \vec{\mathbf{b}}_1^T\vec{\mathbf{y}}'_j$
22:         $\mathbf{x}_{1,j} = (\vec{\mathbf{b}}_2^T + c_i\vec{\mathbf{b}}_3^T)\vec{\mathbf{y}}'_j$
23:         $\mathbf{x}_{2,j} = ((\mathbf{z}_i - \mathbf{c}_i)(\mathbf{z}_i - 2\mathbf{c}_i)\vec{\mathbf{b}}_3^T - \mathbf{z}_i\vec{\mathbf{b}}_4^T + \vec{\mathbf{b}}_5^T)\vec{\mathbf{y}}'_j$
24:     end for
25: end for
26: $(\mathbf{f}_j)_{j\in[tt']} = \mathrm{G}((c_i)_i, (\mathbf{z}_i)_i, (\mathbf{w}'_j)_j, (\mathbf{x}_{1,j})_j, (\mathbf{x}_{2,j})_j)$
27: for $j = 1, \ldots, tt'$ do
28:     $\vec{\mathbf{z}}'_j = \vec{\mathbf{y}}'_j + \mathbf{f}_j\vec{\mathbf{r}}_{\lceil j/t' \rceil}$
29: end for
30: for $k = 1, \ldots, t'$ do
31:     if Rej $\big((\vec{\mathbf{z}}'_j)_{j=k,\ldots,(t-1)t'+k}, (\mathbf{f}_{(i-1)t'+k}\vec{\mathbf{r}}_i)_i, \sigma\big) = 1$ or $\big\|(\vec{\mathbf{z}}'_j)_j\big\|_\infty > 6\sigma$ then
32:         goto 3
33:     end if
34: end for
```

Algorithm 3 Non-Interactive Verifier

```
**Input:** $A, \vec{u}, (\vec{\mathbf{t}}_i)_{i\in[t]}, (c_i)_{i\in[t]}, (\mathbf{z}_i)_{i\in[t]}, (\mathbf{f}_j)_{j\in[tt']}, (\vec{\mathbf{z}}'_j)_{j\in[tt']}$
**Output:** $b \in \{0,1\}$
**for** $i = 1, \dots, t$ **do**
    $\vec{w}_i = A\hat{\mathbf{z}}_i - c_i\vec{u}$
    **for** $j = (i-1)t' + 1, \dots, (i-1)t' + t'$ **do**
        $\mathbf{w}'_j = \vec{\mathbf{b}}_1^T\vec{\mathbf{z}}'_j - \mathbf{f}_j\mathbf{t}_{i,1}$
        $\mathbf{x}_{1,j} = (\vec{\mathbf{b}}_2^T + c_i\vec{\mathbf{b}}_3^T)\vec{\mathbf{z}}'_j + \mathbf{f}_j\mathbf{z}_i - \mathbf{f}_j(\mathbf{t}_{i,2} + c_i\mathbf{t}_{i,3})$
        $\mathbf{x}_{2,j} = ((\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\vec{\mathbf{b}}_3^T - \mathbf{z}_i\vec{\mathbf{b}}_4^T + \vec{\mathbf{b}}_5^T)\vec{\mathbf{z}}'_j$
                $- \mathbf{f}_j((\mathbf{z}_i - c_i)(\mathbf{z}_i - 2c_i)\mathbf{t}_{i,3} - \mathbf{z}_i\mathbf{t}_{i,4} + \mathbf{t}_{i,5})$
    **end for**
**end for**
$(c'_i)_{i\in[t]} = \mathrm{H}(A, \vec{u}, (\vec{\mathbf{t}}_i)_i, (\vec{w}_i)_i)$
$(\mathbf{f}'_j)_{j\in[tt']} = \mathrm{G}((c_i)_i, (\mathbf{z}_i)_i, (\mathbf{w}'_j)_j, (\mathbf{x}_{1,j})_j, (\mathbf{x}_{2,j})_j)$
**if** $\left\|\vec{\mathbf{z}}'_j\right\|_2 \le B \wedge c'_i = c_i \wedge \mathbf{f}'_j = \mathbf{f}_j$ **then**
    **return** 1
**else**
    **return** 0
**end if**
```

3.4 Proof Size

We want to compute the size of the non-interactive proofs that are produced by Algorithm 2. Each of the t polynomial vectors $\vec{\mathbf{t}}_i$ consists of 5 uniformly random polynomials $\mathbf{t}_{i,\nu} \in \mathcal{R}_q$ of which one of them, namely $\mathbf{t}_{i,3}$, is the same in all $\vec{\mathbf{t}}_i$ and only needs to be transmitted once. The polynomials $\mathbf{z}_i$ are also uniformly random. So we need

$$(1 + 5t)n\lceil\log q\rceil/8$$

bytes for $(\vec{\mathbf{t}}_i)_i$ and $(\mathbf{z}_i)_i$. By contrast the vectors $\vec{\mathbf{z}}'_j$ sampled from discrete Gaussian distributions with standard deviation $\sigma = 5T'$ where T' is a bound on the vector $(\mathbf{f}_{(i-1)t'+k}\vec{\mathbf{r}}_i)_i$ in dimension $6tn$. Note that the embedding norm is invariant under multiplication by the monomials $\mathbf{f}_j$ and we do not need to take them into account. The coefficients of a discrete Gaussian are smaller in absolute value than 6σ with probability at least $1 - 2^{-24}$, c.f. [Lyu12, Lemma 4.4], and our non-interactive prover enforces this. So the transmission of $(\vec{\mathbf{z}}'_j)_j$ requires

$$6tt'n\lceil\log(12\sigma)\rceil/8$$

bytes; that is, $\lceil\log(6\sigma)\rceil$ bits per coefficient for the absolute value and one sign bit per coefficient. This does not make use of the fact that the coefficients of $\vec{\mathbf{z}}_j$ are distributed according to a (truncated) discrete Gaussian with known standard deviation, which has less entropy than the uniform distribution. So to get slightly smaller proof sizes one can encode the $\vec{\mathbf{z}}_j$ using a Huffman code, for instance.

Finally, the challenge polynomials $(c_i)_i$ and $(\mathbf{f}_j)_j$ together require

$$t(\lceil\log q\rceil + t'\lceil\log l\rceil)/8$$

bytes.

4 Efficiency Comparison

The simplest application of our proof protocol is to prove knowledge of the secret in LWE samples over $\mathbb{Z}_q$. Let's consider the case of ternary error, modulus q below 2^{32} such that $4096 \mid q-1$, and dimension $d = 1024$. Moduli of around this size are used in FHE schemes and group signature schemes following the hash-and-sign paradigm with Boyen [Boy10] or Ducas-Micciancio standard-model signatures [DM14], for example. Now m such LWE samples $\vec{u}$ are of the form

$$\vec{u} = A'\vec{s}' + \vec{e}$$

with $A' \in \mathbb{Z}_q^{m \times d}$ public and chosen uniformly at random, and $\vec{s} \in \{-1,0,1\}^d$ and $\vec{e} \in \{-1,0,1\}^m$ secret. We can write $A = (A', I_m)$ and then the above LWE equation as $\vec{u} = A(\vec{s}' \parallel \vec{e})$, which is of the form suitable for our proof system. So let $\mathcal{R} = \mathbb{Z}[X]/(X^n+1)$ with $n = 2d = 2048$ and define $\mathbf{s} \in \mathcal{R}_q$ to be the polynomial whose NTT $\hat{\mathbf{s}}$ is given by the concatenation of the vectors $\vec{s}'$ and $\vec{e}$.

4.1 Our Proof System.

With $t' = 3$ lower repetitions, the protocol has soundness error $2/q + 2/l^3$ which is approximately 2^{-31}. So we make $t = 4$ upper repetitions of the protocol to reach the 128 bit security level.

As we are in the power-of-two case recall that we are sampling the randomness vectors $\vec{\mathbf{r}}_i \in \mathcal{R}^6$ using the binomial distribution β_2 independently for each coefficient. By the Chernov bound (8) it follows that $T' = \sqrt{1.1 \cdot 0.625 \cdot 6tn} = 183.83$ with probability at least $1 - 2^{-102}$. So we sample the vectors $\vec{\mathbf{y}}'_j$ with $\sigma = 5T' = 919.13$.

Hardness. The security of the above instantiation of our protocol is based on the hardness of the $\mathrm{RSIS}_{5,8B}$ and RLWE_{1+4t} problems over the ring $\mathbb{Z}_q[X]/(X^{2048}+1)$ with $q \approx 2^{32}$. We analyze known attacks against these problems and start with the Ring SIS problem. Here one needs to find a short vector in the $k = 6n$-dimensional lattice

$$\Lambda^{\perp}(\vec{\mathbf{b}}_1^T) = \left\{ \vec{\mathbf{x}} \in \mathcal{R}^6 \mid \vec{\mathbf{b}}_1^T \vec{\mathbf{x}} \equiv 0 \pmod{q\mathcal{R}} \right\}.$$

It has volume q^n, as one can easily see by writing down a basis. By applying a reduction algorithm to the basis that achieves a root Hermite factor δ_0 we find a short vector of length $q^{n/k}\delta_0^k$. But it is actually sufficient to find a short vector in a sublattice by omitting some of the columns of the matrix corresponding to $\vec{\mathbf{b}}_1^T$.

It turns out that the optimal dimension is $k = 6509$. There a root Hermite factor of 1.0011 would be needed to find a vector of length $8B$. This is out of reach for current reduction algorithms.

For the Ring LWE problem RLWE_{1+4t} we study the primal attack and use the fact that the lattice

$$\Lambda^{\perp} = \left\{ \vec{\mathbf{x}} \in \mathcal{R}^{3+4t} \mid \left(\vec{\mathbf{a}}\, \mathbf{I}_{1+4t}\, \vec{\mathbf{t}} \right) \vec{\mathbf{x}} \equiv \vec{\mathbf{0}} \pmod{q\mathcal{R}} \right\}$$

of volume $q^{(1+4t)n}$ contains the unusually short vector $(\mathbf{s}, \vec{\mathbf{e}}^T, -\mathbf{1})^T$ of expected length about $\sqrt{(2+4t)n10/16}$. Here we do not need to use all of the LWE samples and can instead search for an unusually short vector in a lattice of dimension $k = (i+1)n + 1$ that we get from considering i samples, $1 \leq i \leq 1 + 4t$. Lattice reduction will succeed in finding the unusually short vector if $\lambda_2/\lambda_1 \geq 0.3\delta_0^k$ where we assume that there is no other very short vector and hence, by the Gaussian heuristic,

$$\lambda_2 = \sqrt{\frac{k}{2\pi e}} q^{in/k}.$$

It follows that in our case we would need to achieve a root Hermite factor of about $\delta_0 = 1.0027$, which is impossible.

Size. It follows from the formulas in Section 3.4 that the total size of the proof is 384.03 KB. Here we have not used the trivial encoding for the vectors $\vec{\mathbf{z}}_j$ using $\lceil \log(12\sigma) \rceil = 14$ bits per coefficient. Instead we have computed the entropy of the discrete Gaussian distribution with $\sigma = 919.13$, truncated to $\{-6\sigma, \ldots, 6\sigma\}$, which is below 12 bits. So using a Huffman code the vectors $\vec{\mathbf{z}}_j$ can be encoded using 12 bits per coefficient on average.

4.2 Stern-like Proofs.

We compare our result against the Stern-type protocol presented in [KTX08,LNSW13], as it is the only other lattice-based zero-knowledge protocol capable of proving that a prover knows an exact solution to a system of linear equations, rather than a solution to a related system.

The protocol of [LNSW13] proves knowledge of $\vec{s} \in \mathbb{Z}^n$ with coefficients in $\{-S, \ldots, S\}$, such that $A\vec{s} = \vec{u}$, where A is an $m \times n$ matrix over $\mathbb{Z}_q$. For the analysis, we closely follow [LNSW13, Figure 1].

Set $k = \lfloor \log S \rfloor + 1$. The protocol decomposes $\vec{s}$ into k vectors $\vec{s}_j$ with coefficients in $\{-1, 0, 1\}$ using a type of binary decomposition, and extends each $\vec{s}_j$ to a longer vector $\tilde{\vec{s}}_j$ with a constant number of 0, 1 and -1 entries. The protocol also uses a matrix A' which is derived from A. Then, the protocol proves that that all of the coefficients of the extended vectors do indeed lie in $\{-1, 0, 1\}$ and that $A' \sum_{j=0}^{k-1} \tilde{\vec{s}}_j$ is equal to $\vec{u}$, which implies a short solution to the original system using A.

The protocol involves choosing a random permutation, choosing random masking vectors $\vec{r}_j$ for the $\tilde{\vec{s}}_j$, and making three commitments.

- The first commitment is a commitment to a random permutation and a matrix-vector product involving A' and the $\vec{r}_j$. The random permutation can be generated from a short seed of using a PRG. The result of the matrix-vector product is $n \log q$ elements, but this value itself is never revealed.
- The second commitment is a commitment to permutations of the $\vec{r}_j$. The $\vec{r}_j$ can also be generated using a PRG.

- The third commitment is a commitment to permutations of the vectors $\tilde{\vec{s}}_j + \vec{r}_j$.

The prover sends these three commitments to the verifier. Using a simple commitment scheme which commits by hashing and is secure in the random oracle model, each of these commitments has size 256 bits.

The protocol uses challenges from the set $\{1, 2, 3\}$. As for the prover's responses to challenges, when the challenge is equal to 1, the prover sends the permuted $\vec{r}_j$ and $\tilde{\vec{s}}_j$ to the verifier. This means sending $6kn + 3kn\lceil\log q\rceil$ bits. When the challenge is equal to 2, the prover sends the permutation seed and the permuted $\tilde{\vec{s}}_j + \vec{r}_j$ to the verifier. This means sending $256 + 3kn\lceil\log q\rceil$ bits. When the challenge is equal to 3, the prover sends the permutation seed and the permuted $\vec{r}_j$ to the verifier. This means sending 512 bits, as it is sufficient to send the random seed for the $\vec{r}_j$ alongside the permutation seed.

Summing up the challenge responses, dividing by 3, and adding the sizes of the 3 commitments, a single execution of the protocol has an expected proof size of $1024{+}kn(2\lceil\log q\rceil{+}1)$ bits. A single execution has a soundness error of $2/3$. For a soundness error of λ bits, this means repeating the protocol $t = \lceil\lambda/(\log 3 - 1)\rceil$ times, for a total proof size of $1024t + knt(2\lceil\log q\rceil + 2)$ bits.

For the application above, with $\lambda = 124$ and $k = 1$, the result is a proof size of 3.44 MB.

References

Ban93. Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296:625–635, 1993.

BBB+18. Benedikt Bünz, Jonathan Bootle, Dan Boneh, Andrew Poelstra, Pieter Wuille, and Gregory Maxwell. Bulletproofs: Short proofs for confidential transactions and more. In *IEEE Symposium on Security and Privacy*, pages 315–334, 2018.

BBC+18. Carsten Baum, Jonathan Bootle, Andrea Cerulli, Rafaël del Pino, Jens Groth, and Vadim Lyubashevsky. Sub-linear lattice-based zero-knowledge arguments for arithmetic circuits. In *CRYPTO*, pages 669–699, 2018.

BCC+16. Jonathan Bootle, Andrea Cerulli, Pyrros Chaidos, Jens Groth, and Christophe Petit. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting. In *EUROCRYPT*, pages 327–357, 2016.

BCK+14. Fabrice Benhamouda, Jan Camenisch, Stephan Krenn, Vadim Lyubashevsky, and Gregory Neven. Better zero-knowledge proofs for lattice encryption and their application to group signatures. In *ASIACRYPT*, pages 551–572, 2014.

BDL+18. Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In *SCN*, pages 368–385, 2018.

Beu19. Ward Beullens. On sigma protocols with helper for mq and pkp, fishy signature schemes and more. Cryptology ePrint Archive, Report 2019/490, 2019. https://eprint.iacr.org/2019/490.

BN19. Carsten Baum and Ariel Nof. Concretely-efficient zero-knowledge arguments for arithmetic circuits and their application to lattice-based cryptography. Cryptology ePrint Archive, Report 2019/532, 2019. `https://eprint.iacr.org/2019/532`.

Boy10. Xavier Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *Public Key Cryptography*, pages 499–517, 2010.

DM14. Léo Ducas and Daniele Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO*, pages 335–352, 2014.

dPLS18. Rafaël del Pino, Vadim Lyubashevsky, and Gregor Seiler. Lattice-based group signatures and zero-knowledge proofs of automorphism stability. In *CCS*, pages 574–591, 2018.

dPLS19. Rafael del Pino, Vadim Lyubashevsky, and Gregor Seiler. Short discrete log proofs for fhe and ring-lwe ciphertexts. In *PKC*, 2019.

Gen09. Craig Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178, 2009.

GK15. Jens Groth and Markulf Kohlweiss. One-out-of-many proofs: Or how to leak a secret and spend a coin. In *EUROCRYPT*, pages 253–280, 2015.

Gro10. Jens Groth. A verifiable secret shuffle of homomorphic encryptions. *J. Cryptology*, 23(4):546–579, 2010.

KTX08. Akinori Kawachi, Keisuke Tanaka, and Keita Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *ASIACRYPT*, pages 372–389, 2008.

LNSW13. San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *PKC*, pages 107–124, 2013.

LNWX18. San Ling, Khoa Nguyen, Huaxiong Wang, and Yanhong Xu. Constant-size group signatures from lattices. In *PKC*, pages 58–88, 2018.

LPR13. Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-lwe cryptography. In *EUROCRYPT*, pages 35–54, 2013.

Lyu09. Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In *ASIACRYPT*, pages 598–616, 2009.

Lyu12. Vadim Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, pages 738–755, 2012.

Ste93. Jacques Stern. A new identification scheme based on syndrome decoding. In *CRYPTO*, pages 13–21, 1993.

YAZ$^+$19. Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In *Crypto*, 2019. To appear.