Simple Amortized Proofs of Shortness for Linear Relations over Polynomial Rings

Carsten Baum[1⋆] and Vadim Lyubashevsky[2⋆⋆]

[1] Department of Computer Science, Bar-Ilan University, Israel
carsten.baum@biu.ac.il
[2] IBM Research – Zurich, Switzerland
vad@zurich.ibm.com

Abstract. For a public value y and a linear function f, giving a zero-knowledge proof of knowledge of a secret value x that satisfies $f(x) = y$ is a key ingredient in many cryptographic protocols. Lattice-based constructions, in addition, require proofs of "shortness" of x. Of particular interest are constructions where f is a function over polynomial rings, since these are the ones that result in efficient schemes with short keys and outputs.

All known approaches for such lattice-based zero-knowledge proofs are not very practical because they involve a basic protocol that needs to be repeated many times in order to achieve negligible soundness error. In the amortized setting, where one needs to give zero-knowledge proofs for many equations for the same function f, the situation is more promising, though still not yet fully satisfactory. Current techniques either result in proofs of knowledge of x's that are exponentially larger than the x's actually used for the proof (i.e. the *slack* is exponential), or they have polynomial slack but require the number of proofs to be in the several thousands before the amortization advantages "kick in".

In this work, we give a new approach for constructing amortized zero-knowledge proofs of knowledge of short solutions over polynomial rings. Our proof has small polynomial slack and is practical even when the number of relations is as small as the security parameter.

1 Introduction

The security of lattice-based cryptographic primitives is based on the hardness of recovering a short vector $\mathbf{s}$ when given a matrix $\mathbf{A}$ and $\mathbf{t} = \mathbf{A}\mathbf{s}$ as inputs. The operations are performed over some ring R, which most commonly is either $\mathbb{Z}_p$ or a polynomial ring $\mathbb{Z}_p[X]/(X^n + 1)$.[1] Depending on the primitive, $\mathbf{s}$ can represent the secret key, the randomness used during encryption, the signature of a message, or anything else that one should not be able to obtain just by knowing $\mathbf{A}$ and $\mathbf{t}$. In many cryptographic protocols, someone announcing the value $\mathbf{t}$ would also need to be able to prove the knowledge of a short pre-image $\bar{\mathbf{s}}$ (the pre-image may not be unique) of $\mathbf{t}$ satisfying

$$\mathbf{A} \cdot \bar{\mathbf{s}} = \mathbf{t} \tag{1}$$

The first approach developed for constructing such a zero-knowledge proof is using Stern-type proofs for codes [Ste93] adapted to the lattice setting [KTX08,LNSW13]. The main downside of this technique is that it has rather long proofs. Each run of the protocol has soundness error 2/3,

⋆ Supported by the BIU Center for Research in Applied Cryptography and Cyber Security in conjunction with the Israel National Cyber Bureau in the Prime Minister's Office.
⋆⋆ Supported by the SNSF ERC Transfer Grant CRETP2-166734-FELICITY

[1] We will use this polynomial ring throughout our paper as it is the one that is almost exclusively used in practice. Instantiations with other monic polynomials are also possible.

thus requiring over 200 repetitions for 128 bits of security and over 400 repetitions if one would like to have 128 bits of *quantum* security in the resulting NIZK proof constructed via the Fiat-Shamir technique. Even the most basic application, such as proving the knowledge of a solution to a hash function (e.g. the one in [LMPR08]), would require around 1KB for every round, thus making the total proof size approximately 400 KB. More complicated applications, as well as those in which the inputs are not taken from $\{0,1\}^k$, quickly push such proofs to the order of Megabytes. In applications where hundreds of such proofs need to be given (which is the scenario that we consider in this paper), this technique is therefore rather impractical.

Another technique for creating zero-knowledge proofs is the "Fiat-Shamir with Aborts" approach which allows to create a proof of knowledge of a vector $\bar{\mathbf{s}}$ with small coefficients (though larger than those in $\mathbf{s}$) and a ring element $\bar{\mathbf{c}}$ with very small coefficients satisfying $\mathbf{A}\bar{\mathbf{s}} = \bar{\mathbf{c}}\mathbf{t}$ [Lyu09,Lyu12]. As long as the ring R has many elements with small coefficients, such proofs are very efficient, producing soundness of $1 - 2^{-128}$ with just one iteration. While these proofs are good enough for constructing practical digital signatures (e.g. [GLP12,DDLL13,BG14]), commitment schemes with proofs of knowledge [BKLP15,BDOP16], and certain variants of verifiable encryption schemes [LN17], they prove less than what the honest prover knows. In many applications where zero-knowledge proofs are used, in particular those that need to take advantage of additive homomorphisms, the presence of the element $\bar{\mathbf{c}}$ makes these kinds of "approximate" proofs too weak to be useful. As of today, we do not have any truly practical zero-knowledge proof systems that give a proof of (1).

1.1 Amortized Proofs

When one considers *amortized* proofs, the situation is considerably more promising. Before describing the results in this area, we recall the concepts of *slack* and *overhead* that determine the efficiency and utility of such proofs. The ratio between the norm of the vector $\bar{\mathbf{s}}$ that can be extracted from the proof and the vector $\mathbf{s}$ that the honest prover uses to create the zero-knowledge proof is referred to as the slack. A slack of d implies that the coefficients of $\bar{\mathbf{s}}$ will have (approximately) $\log d$ more bits than those of $\mathbf{s}$. Therefore having a small polynomial slack is usually fine for practical purposes. The *overhead* refers to the number of vectors (of the same dimension as $\mathbf{s}$, and coefficient sizes which are essentially determined by the slack) that need to be generated for a proof having negligible soundness error. Because the overhead affects the size of the total proof linearly, it is crucial to keep it to a very small constant – preferably one or two.

One idea to prove the relation in (1) is to use the above "Fiat-Shamir with Aborts" protocol, but with challenges that come from the set $\{0,1\}$. This indeed gives a proof of (1) with a small slack, but the soundness error would be $1/2$, which would require the overhead to be the security parameter λ, which makes the scheme impractical.[2] On the other hand, the amortized proof given in [DPSZ12] using the protocol from [CD09] showed that one only needs to generate $\lambda + k$ vectors to prove the knowledge of k equations, which gives an overhead of $1 + \lambda/k$. The main downside of this protocol is that the slack is exponential in k.

The works of [BDLN16] and [CDXY17] gave a novel protocol, still using the Fiat-Shamir with Aborts with 0/1 challenges idea, in which the overhead was constant (down to 2), while the slack

[2] In fact, this scheme would be even less practical than Stern proofs because Stern proofs have slack 1. But as we discuss later, the advantage of such proofs is that they can be efficient in the amortized setting, whereas we do not currently know of a way in which Stern-type proofs can exploit amortization.

could be bounded by a small polynomial in the security parameter.[3] The main downside of these protocols is that they require the number of equations to be fairly large before amortization kicks in. In particular, if the security parameter is λ, then one needs to have more than $k = 4\lambda^2$ equations.[4] Thus scenarios where one does not have too many (around λ as in [DPSZ12]) equations to prove would not benefit from the protocol.

The work of [DL17] showed that one could decrease the number of equations in the above protocol by a factor of α by increasing the running time of the proof by a factor of 2^α. They also gave a protocol requiring even fewer equations when the functions are over polynomial rings of dimension n by using an idea from [BCK+14]. Instead of running the Fiat-Shamir protocol with 0/1 challenges, one could do it with with challenges of the form X^i. This further reduces the required number of equations by a factor approximately $\log n$. Nevertheless, one still needs at least a few thousand of them in order to be able to use amortization in a practical manner.[5]

1.2 Our Contribution

In this paper, we give a novel protocol that works over polynomial rings in which the slack is polynomial and amortization kicks in as soon as the number of equations is essentially the security parameter.

Suppose that $a_1, \ldots, a_m$ are polynomials in the ring $R_p = \mathbb{Z}_p[X]/(X^n + 1)$ and for $j \in \{1, \ldots, k\}$, $s_1^{(j)}, \ldots, s_m^{(j)}$ are polynomials with small coefficients satisfying

$$\sum_{i=1}^{m} a_i \cdot s_i^{(j)} = t^{(j)}. \tag{2}$$

In this section, we sketch the framework for the proof of (2) for k equations over polynomial rings of degree n which can be done with overhead $1 + \lambda/k$ and slack approximately $O(\sqrt{nmk\lambda})$ where λ is the security parameter.

Our protocol is most naturally explained as working in two steps, both of which use the approximate proof of knowledge protocols from [Lyu09,Lyu12]. While each part is an "approximate" proof, the combination of the two proofs yields a pre-image of the exact $t^{(j)}$.

In the first step, we prove the existence of small polynomials $\bar{s}_i^{(j)}$ and a polynomial $\bar{c}$ satisfying

$$\sum_{i=1}^{m} a_i \cdot \bar{s}_i^{(j)} = \bar{c} \cdot t^{(j)}. \tag{3}$$

This is done by simply running the protocol from [Lyu09,Lyu12] over the ring R_p simultaneously for all k equations. The second part of our protocol no longer works over the ring R_p, but rather over $\mathbb{Z}_p$. For convenience of notation, we will therefore rewrite (2) and (3) as equations over $\mathbb{Z}_p$ rather than R_p.

[3] To be precise, only by a super-polynomial factor in the first work.

[4] Asymptotically, one could decrease k to $O(\lambda^{1+\epsilon})$, but it works only for rather large λ that would not be used in practice.

[5] We give more details about this in Section 4.3.

The polynomials $a_1, \ldots, a_m$ can be combined into a matrix $\mathbf{A} \in \mathbb{Z}_p^{n \times nm}$ that already incorporates the polynomial multiplication in the ring R_p (we define this matrix $\mathbf{A}$ explicitly in (12)). Then $\sum_{i=1}^{m} a_i \cdot s_i = t$ over R_p can be expressed as

$$\mathbf{As} = \mathbf{t}, \qquad \text{with } \mathbf{s} = \begin{bmatrix} s_1 \\ \vdots \\ s_m \end{bmatrix},$$

where the polynomials $s_i \in R_p$ are interpreted as vectors in $\mathbb{Z}_p^n$ in the natural way via their coefficients. Combining all k equations allows us to rewrite (2) and (3) as

$$\mathbf{AS} = \mathbf{T} \tag{4}$$

and

$$\mathbf{A}\bar{\mathbf{S}} = \mathsf{Rot}(\bar{c}) \cdot \mathbf{T}, \tag{5}$$

where $\mathsf{Rot}(\bar{c})$ is a polynomial multiplication matrix (which we again formally introduce later – see (11)).

The second part of our protocol then uses the approximate zero-knowledge proof from [Lyu12] over the ring $\mathbb{Z}_p$ to prove the knowledge of an $\bar{\mathbf{S}}'$ with small coefficients and a matrix $\bar{\mathbf{C}}' \in \{-1, 0, 1\}^{k \times \ell}$ such that

$$\mathbf{A}\bar{\mathbf{S}}' = \mathbf{T} \cdot \bar{\mathbf{C}}'. \tag{6}$$

It can then be shown that (5) and (6) taken together imply that the prover either knows an exact solution to (4) or is able to solve an instance of the Ring-SIS problem: because of the commutativity of multiplication in the ring R_p, (5) can be rewritten as

$$\mathbf{A}\bar{\mathbf{S}} = \mathsf{Rot}(\bar{c}) \cdot \mathbf{T} \quad \Longrightarrow \quad \mathsf{Rot}(\bar{c}^{-1}) \cdot \mathbf{A}\bar{\mathbf{S}} = \mathbf{T} \quad \Longrightarrow \quad \mathbf{A} \cdot (\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}} = \mathbf{T}, \tag{7}$$

where $\mathbf{I}_m$ is an m-dimensional identity matrix. Plugging the above equation into (6), we get

$$\mathbf{A}\bar{\mathbf{S}}' = \mathbf{A} \cdot (\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}'. \tag{8}$$

If $\bar{\mathbf{S}}' \neq (\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}'$, then $(\mathsf{Rot}(\bar{c}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}' \neq \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}'$ and (8), which can be rewritten as

$$\mathbf{A} \cdot (\mathsf{Rot}(\bar{c}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}' = \mathbf{A} \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}',$$

imply a solution to Ring-SIS due to the fact that the coefficients of $(\mathsf{Rot}(\bar{c}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}'$ and $\bar{\mathbf{S}} \cdot \bar{\mathbf{C}}'$ are all small. If we assume that Ring-SIS with those coefficient sizes is a hard problem, then it must be that

$$\bar{\mathbf{S}}' = (\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}'. \tag{9}$$

We would like to then conclude that this equality implies that all the coefficients of $(\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}$ are small. Since we know from (7) that $(\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}$ is a solution for (4), this will complete the proof.

To see why the coefficients of $(\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}$ are small, we need to examine how the extracted matrix $\bar{\mathbf{C}}'$ is created. This matrix is the difference of two challenges $\mathbf{C}_1'$ and $\mathbf{C}_2' \in \{0, 1\}^{k \times \ell}$ from the second step of our Σ-protocol. The crucial part is that the random challenge $\mathbf{C}_2'$ is chosen *after*

all the other variables have been fixed.[6] We then show that if some matrix $\mathbf{M}$ has large coefficients, then with high probability (which depends on the parameter ℓ) so will the product $\mathbf{M} \cdot (\mathbf{C}'_1 - \mathbf{C}'_2)$ when $\mathbf{C}'_2$ is chosen uniformly from $\{0,1\}^{k \times \ell}$. But the matrix $\bar{\mathbf{S}}'$ in (9) has small coefficients, and therefore all the coefficients of $(\mathsf{Rot}(\bar{c}^{-1}) \otimes \mathbf{I}_m) \cdot \bar{\mathbf{S}}$ must be small as well.

1.3 Beyond Polynomial Rings

The key to our construction is the fact there is a large challenge set of matrices (with small coefficients) that "commutes" with $\mathbf{A}$ – a fact which is used in (7). Above, we sketched our result when the matrix $\mathbf{A}$ was as in (12) and we used the commutativity from (13). The soundness of this scheme was based on the hardness of the Ring-SIS problem. More generally, we can take the matrix $\mathbf{A}$ to be as in (14) and use the commutativity in (15), which will base the soundness of the scheme on the Module-SIS problem (the presentation of the paper follows this more general approach).

If we keep the dimension of the matrix $\mathbf{A}$ in (14) fixed and increase d, then the dimension of the ring R_p (in which the challenge c lies) decreases by a factor of d. Since the number of possible challenges needs to remain constant, a smaller ring dimension implies that the norm of the challenges must be larger, and this has a direct consequence on the size of the parameters and the size of the proof. In practice, if we would like the number of challenges to be 2^{256}, then using rings of dimension at least 256 and the challenge set as in (21) allows for challenges to have ℓ_2-norm less than 8. If we reduce the dimension of the ring R_p all the way down to 1, thus proving linear relations over $\mathbb{Z}_p$, the norm of the challenge would would be 2^{256}, and this would result in completely impractical parameters.

An attempt for working over small rings. Instead of creating the proof in (5), one could instead prove knowledge of $\bar{\mathbf{S}}$ and a square, invertible $\bar{\mathbf{C}}$ such that

$$\mathbf{A} \cdot \bar{\mathbf{S}} = \mathbf{T} \cdot \bar{\mathbf{C}}, \qquad \text{or equivalently,} \qquad \mathbf{A} \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}^{-1} = \mathbf{T}$$

Plugging the value for $\mathbf{T}$ into (6), we would obtain

$$\mathbf{A} \cdot \bar{\mathbf{S}}' = \mathbf{A} \cdot \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}^{-1} \cdot \bar{\mathbf{C}}'. \tag{10}$$

If the equality $\bar{\mathbf{S}}' = \bar{\mathbf{S}} \cdot \bar{\mathbf{C}}^{-1} \cdot \bar{\mathbf{C}}'$ holds, then we would be able to conclude, via the same argument as at the end of Section 1.2, that the coefficients of $\bar{\mathbf{S}} \cdot \bar{\mathbf{C}}^{-1}$ are small. The problem with the proof is that we do not have an argument as to why the equality must be true. In Section 1.2, we were able to use the commutativity to show that if this equality does not hold, then we have a solution to the Ring-SIS problem. In (10), we have no way of moving the matrix $\bar{\mathbf{C}}^{-1}$ to the other side, and therefore cannot argue that we would have a solution to SIS. We should mention that if the modulus p is set large enough and the coefficients of $\bar{\mathbf{S}}, \bar{\mathbf{S}}'$ are small enough, then one could prove that for all but a negligible fraction of randomly-generated $\mathbf{A}$, there do not exist matrices $\bar{\mathbf{S}}, \bar{\mathbf{S}}', \bar{\mathbf{C}}, \bar{\mathbf{C}}'$ with small coefficients that satisfy (10). A rough calculation, however, shows that the parameters that would be required for the preceding to hold would be completely outside of anything that one would want to use in practice. We thus leave it as an open problem to reduce the number of amortized samples (as compared to [BDLN16,CDXY17,DL17]) required for efficiently proving linear relations over $\mathbb{Z}_p$.

[6] This is enforced in the scheme by choosing the challenge for the second part of the protocol as the hash of the output of Step 1 of the protocol, where the hash function is modeled as a random oracle.

2 Preliminaries

Throughout this work, lower-case bold variables such as $\mathbf{a}$ will denote vectors while their counterparts written in upper-case such as $\mathbf{A}$ are matrices. We will denote the set $\{1, \ldots, k\}$ as $[k]$.

2.1 Polynomials

Let R be the ring $\mathbb{Z}[X]/(X^n+1)$, and for a prime p, R_p be the ring $\mathbb{Z}_p[X]/(X^n+1)$. For an element $c = c_0 + c_1 X + \ldots + c_{n-1}X^{n-1} \in R$, we define the matrix

$$\mathsf{Rot}(c) \triangleq \begin{bmatrix} c_0 & -c_{n-1} & -c_{n-2} & \ldots & -c_1 \\ c_1 & c_0 & -c_{n-1} & \ldots & -c_2 \\ c_2 & c_1 & c_0 & \ldots & -c_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ c_{n-1} & c_{n-2} & c_{n-3} & \ldots & c_0 \end{bmatrix}. \tag{11}$$

The coefficients of column i (for $0 \leq i < n$) of $\mathsf{Rot}(c)$ are exactly the coefficients of the polynomial cX^i in R. It is easy to check that for any two polynomials $b, c \in R$, we have

$$\mathsf{Rot}(b+c) = \mathsf{Rot}(c+b) \text{ and } \mathsf{Rot}(b \cdot c) = \mathsf{Rot}(b) \cdot \mathsf{Rot}(c).$$

In particular, this implies that the multiplication of such matrices is commutative.

We define $\mathbf{I}_m$ to be an m-dimensional integer identity matrix. For elements $a_1, \ldots, a_m \in R$, let $\mathbf{A}$ be a matrix

$$\mathbf{A} = [\mathsf{Rot}(a_1) \mid \ldots \mid \mathsf{Rot}(a_m)] \in \mathbb{Z}^{n \times nm}. \tag{12}$$

Then for any $c \in R$, we have

$$\mathsf{Rot}(c) \cdot \mathbf{A} = \mathbf{A} \cdot (\mathsf{Rot}(c) \otimes \mathbf{I}_m) \tag{13}$$

where $\otimes$ denotes the tensor product of matrices. More generally, if

$$\mathbf{A} = \begin{bmatrix} \mathsf{Rot}(a_{1,1}) & \mid & \ldots & \mid & \mathsf{Rot}(a_{1,m}) \\ \vdots & & \ddots & & \vdots \\ \mathsf{Rot}(a_{d,1}) & \mid & \ldots & \mid & \mathsf{Rot}(a_{d,m}) \end{bmatrix} \in \mathbb{Z}^{dn \times nm}, \tag{14}$$

then we have

$$(\mathsf{Rot}(c) \otimes \mathbf{I}_d) \cdot \mathbf{A} = \mathbf{A} \cdot (\mathsf{Rot}(c) \otimes \mathbf{I}_m). \tag{15}$$

To simplify notation, for any polynomial $c \in R$ and positive integer x, we define

$$\mathsf{Rot}_x(c) \triangleq (\mathsf{Rot}(c) \otimes \mathbf{I}_x).$$

2.2 Vectors and Matrices

Let $\mathbf{a} = (a_1, \ldots, a_n) \in \mathbb{Z}^n$, then the ℓ_2, ℓ_∞-norms of $\mathbf{a}$ are

$$||\mathbf{a}|| = \sqrt{\sum_i a_i^2} \text{ and } ||\mathbf{a}||_\infty = \max_i |a_i|.$$

If $\mathbf{a}$ instead comes from $\mathbb{Z}_p^n$ then we consider the natural embedding of each coefficient from $\mathbb{Z}_p$ into the interval $[-(p-1)/2, (p-1)/2] \subset \mathbb{Z}$, and then use the definition of the norm as above. For $\mathbf{a} \in R$ (or R_p), the norm is defined to be the norm of the vector of its coefficients.

For matrices $\mathbf{A}, \mathbf{B} \in \mathbb{Z}^{n \times m}$ comprised of elements $a_{i,j}$ and $b_{i,j}$, we define their norms and inner products by viewing the matrices as nm-dimensional vectors:

$$\|\mathbf{A}\|_\infty \triangleq \max_{i,j} \|a_{i,j}\|_\infty$$
$$\|\mathbf{A}\| \triangleq \sqrt{\sum_{i,j} a_{i,j}^2}$$
$$\langle \mathbf{A}, \mathbf{B} \rangle \triangleq \sum_{i,j} a_{i,j} \cdot b_{i,j}$$

For a matrix $\mathbf{X} \in \mathbb{Z}^{n \times m}$, the largest singular value $s_1(\mathbf{X})$ has the property that for any $\mathbf{v} \in \mathbb{Z}^m$, $\|\mathbf{X} \cdot \mathbf{v}\| \leq s_1(\mathbf{X}) \cdot \|\mathbf{v}\|$. Because the singular values are invariant under transpose, we also have that for any $\mathbf{v} \in \mathbb{Z}^n$, $\|\mathbf{v}^T \cdot \mathbf{X}\| \leq s_1(\mathbf{X}) \cdot \|\mathbf{v}\|$.

2.3 The Ring-SIS and Module-SIS Problems

We give the definitions of Ring-SIS (using the notation from Section 2.1) from [LM06] and its generalization, Module-SIS, from [LS15] in the ℓ_∞-norm.

Definition 2.1. *In the Ring-SIS$_{n,m,\beta}$ problem, we are given random polynomials $a_1, \ldots, a_m \in R_p$ and are asked to find a non-zero integer vector $\mathbf{s} \in \mathbb{Z}^{nm}$ such that $\|\mathbf{s}\|_\infty \leq \beta$ and $\mathbf{As} = \mathbf{0} \bmod p$, where $\mathbf{A} \triangleq [\mathsf{Rot}(a_1) \mid \ldots \mid \mathsf{Rot}(a_m)]$.*

Definition 2.2. *In the d-dimensional Module-SIS$_{n,m,\beta}$ problem, we are given random polynomials $a_{i,j} \in R_p$ for $i \in [d]$ and $j \in [m]$ and are asked to find a non-zero integer vector $\mathbf{s} \in \mathbb{Z}^{nm}$ such that $\|\mathbf{s}\|_\infty \leq \beta$ and $\mathbf{As} = \mathbf{0} \bmod p$, where*

$$\mathbf{A} \triangleq \begin{bmatrix} \mathsf{Rot}(a_{1,1}) & \mid & \ldots & \mid & \mathsf{Rot}(a_{1,m}) \\ \vdots & & \ddots & & \vdots \\ \mathsf{Rot}(a_{d,1}) & \mid & \ldots & \mid & \mathsf{Rot}(a_{d,m}) \end{bmatrix}.$$

Notice that the Ring-SIS problem is exactly the Module-SIS problem with $d = 1$.

2.4 Probability

The below lemma and the corollary that follows it state (in a slightly generalized way) that if $\mathbf{s} \cdot \mathbf{C} \bmod p$ only has "small" coefficients when $\mathbf{C}$ is a matrix with random 0/1 coefficients, then $\mathbf{s}$ must only have small coefficients as well.

Lemma 2.3. *For all $\mathbf{s} \in \mathbb{Z}_p^k$ and $\mathbf{c}' \in \{0,1\}^k$ it holds that*

$$\Pr_{\mathbf{c} \leftarrow \{0,1\}^k} \left[\|\langle \mathbf{s}, \mathbf{c} - \mathbf{c}' \rangle \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \right] \leq \frac{1}{2}.$$

Proof. Let s_i be the coefficient of $\mathbf{s}$ such that $\|s_i\|_\infty = \|\mathbf{s}\|_\infty$. Then we can write

$$\langle \mathbf{s}, \mathbf{c} - \mathbf{c}' \rangle = s_i c_i + \sum_{j \neq i} s_j c_j - \langle \mathbf{s}, \mathbf{c}' \rangle \triangleq s_i c_i + r \bmod p.$$

If $\|r\|_\infty \geq \frac{1}{2} \cdot \|\mathbf{s}\|_\infty = \frac{1}{2} \cdot \|s_i\|_\infty$, then c_i would need to be 1 for any chance of $s_i c_i + r$ to be less than $\frac{1}{2} \cdot \|\mathbf{s}\|_\infty$. So we have

$$\Pr_{\mathbf{c} \leftarrow \{0,1\}^k} \left[\|\langle \mathbf{s}, \mathbf{c} - \mathbf{c}' \rangle \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \;\middle|\; \|r\|_\infty \geq \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \right] \leq \Pr_{c_i \leftarrow \{0,1\}} [c_i = 1] = \frac{1}{2}.$$

If $\|r\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty = \|s_i\|_\infty$, then we will show that

$$\|r + s_i\|_\infty \geq \frac{1}{2} \cdot \|s_i\|_\infty, \tag{16}$$

which implies that

$$\Pr_{\mathbf{c} \leftarrow \{0,1\}^k} \left[\|\langle \mathbf{s}, \mathbf{c} - \mathbf{c}' \rangle \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \;\middle|\; \|r\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \right] \leq \Pr_{c_i \leftarrow \{0,1\}} [c_i = 0] = \frac{1}{2},$$

which will complete the proof of the lemma.

To prove (16), note that because we can assume that $|s_i| \leq p/2$ and $|r| < |s_i|/2$, we know that $\|r + s_i\|_\infty$ is either equal to $|r + s_i|$ or $|r + s_i \pm p|$. It's clear that in the former case, we have $|r + s_i| > |s_i|/2$. For the latter case, assume for the sake of contradiction that $u = r + s_i \pm p$ with $|u| < |s_i|/2$. This would imply that

$$p = |\pm p| = |r + s_i - u| \leq |r| + |s_i| + |u| < |s_i|/2 + |s_i| + |s_i|/2 = 2|s_i| \leq p.$$

□

Corollary 2.4. *Let* $\mathbf{M}$ *be any matrix in* $\mathbb{Z}_p^{r \times k}$ *and* $\mathbf{C}'$ *any matrix in* $\{0,1\}^{k \times \ell}$*. Then*

$$\Pr_{\mathbf{C} \leftarrow \{0,1\}^{k \times \ell}} \left[\|\mathbf{M} \cdot (\mathbf{C} - \mathbf{C}') \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{M}\|_\infty \right] \leq 2^{-\ell}.$$

Proof. Let $\mathbf{s}$ be the row of $\mathbf{M}$ such that $\|\mathbf{s}\|_\infty = \|\mathbf{M}\|_\infty$. Then

$$\begin{aligned}
&\Pr_{\mathbf{C} \leftarrow \{0,1\}^{k \times \ell}} \left[\|\mathbf{M} \cdot (\mathbf{C} - \mathbf{C}') \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{M}\|_\infty \right] \\
&\quad = \Pr_{\mathbf{C} \leftarrow \{0,1\}^{k \times \ell}} \left[\|\mathbf{M} \cdot (\mathbf{C} - \mathbf{C}') \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \right] \\
&\quad \leq \Pr_{\mathbf{C} \leftarrow \{0,1\}^{k \times \ell}} \left[\|\mathbf{s} \cdot (\mathbf{C} - \mathbf{C}') \bmod p\|_\infty < \frac{1}{2} \cdot \|\mathbf{s}\|_\infty \right] \\
&\quad \leq 2^{-\ell},
\end{aligned}$$

with the last inequality following directly from Lemma 2.3 because the columns of $\mathbf{C}$ are independently chosen. □

2.5 Discrete Gaussians over $\mathbb{Z}^n$

Define the function $\rho_\sigma(x) = \exp\left(\frac{-x^2}{2\sigma^2}\right)$ and the discrete Gaussian distribution over the integers, D_σ, as

$$D_\sigma(x) = \frac{\rho(x)}{\rho(\mathbb{Z})} \text{ where } \rho(\mathbb{Z}) = \sum_{v\in\mathbb{Z}} \rho(v).$$

We will write $\mathbf{X} \leftarrow D_\sigma^{n\times m}$ to mean that every coefficient of the matrix $\mathbf{X}$ is distributed according to D_σ.

Using the tail bounds for the 0-centered discrete Gaussian distribution (cf. [Ban93]), we can show that for any $\sigma > 0$, $x \leftarrow D_\sigma$ is likely to be close to σ. Namely, for any $k > 0$ it holds that

$$\Pr_{x\leftarrow D_\sigma}[|x| > k\sigma] \leq 2e^{-k^2/2}, \tag{17}$$

and when $\mathbf{x}$ is drawn from D_σ^n, we have

$$\Pr_{\mathbf{x}\leftarrow D_\sigma^n}[\|\mathbf{x}\| > \sqrt{2n}\cdot\sigma] < 2^{-n/4}. \tag{18}$$

3 Proving Knowledge

Let $\mathbf{A} \in \mathbb{Z}_p^{dn\times nm}$ be a matrix shaped as in (14), $\mathbf{S} \in \mathbb{Z}^{nm\times k}$ be an arbitrary matrix and $\mathbf{T} = \mathbf{A}\cdot\mathbf{S}$. Our goal is to give a zero-knowledge proof of knowledge of an $\bar{\mathbf{S}}$ such that

$$\mathbf{A}\cdot\bar{\mathbf{S}} = \mathbf{T}.$$

Since the soundness of our proof of knowledge relies on the fact that it is hard to find a short solution $\mathbf{s}$ such that $\mathbf{As} = \mathbf{0}$, it should be assumed that the prover is not the one who generates $\mathbf{A}$. Instead, one should think of $\mathbf{A}$ as being chosen such that each polynomial $a_{i,j} \in R_p$ in (14) is random. In practice, this can be enforced by, for example, defining $\mathbf{A} = \mathrm{XOF}(0)$ for some standardized extendable output function XOF.

3.1 Setting the Stage

We will now introduce the parameters and subroutines of our construction.

Singular Values. Let us write

$$\mathbf{S} = \begin{bmatrix}\mathbf{S}_1\\ \vdots \\ \mathbf{S}_m\end{bmatrix}$$

where each $\mathbf{S}_i$ is an $n\times k$ matrix, then we define the parameters s and $\mathsf{s}_{\mathsf{part}}$ as any reals that satisfy

$$\mathsf{s} \geq s_1(\mathbf{S}) \tag{19}$$

$$\mathsf{s}_{\mathsf{part}} \geq \max_i\{s_1(\mathbf{S}_i)\}. \tag{20}$$

Since the parameters of our proof are most naturally derived from s and $\mathsf{s}_{\mathsf{part}}$, we will assume that these values are public information.

For example, if $\mathbf{S}$ is a matrix in which the maximal coefficient has absolute value at most γ, then we can set

$$\mathsf{s} = \gamma \cdot \sqrt{nmk} \text{ and } \mathsf{s}_{\mathsf{part}} = \gamma \cdot \sqrt{nk},$$

Or if each coefficient of $\mathbf{S}$ is chosen according to D_σ, then one can use the bound in (32) and set

$$\mathsf{s} = \sigma \cdot (\sqrt{nm} + \sqrt{k} + 5) \text{ and } \mathsf{s}_{\mathsf{part}} = \sigma \cdot (\sqrt{n} + \sqrt{k} + 5).$$

The Challenge Sets. As outlined in Section 1.2, our proof uses two different challenges

$$c_1 \in \mathcal{C} \subset R_p \text{ and } \mathbf{C}_2 \in \{0,1\}^{k \times \ell}.$$

The challenge set $\mathcal{C}$ consists of polynomials in R whose coefficients are 0 except for α ± 1's:

$$\mathcal{C} \triangleq \{c \in R : \|c\|_\infty = 1, \|c\| = \sqrt{\alpha}\}, \tag{21}$$

A property that we will need the set $\mathcal{C}$ to have is that all the non-zero elements of the difference set $\mathcal{C} - \mathcal{C}$ are invertible in the ring R_p. This property is achieved if one sets the modulus p properly so that the polynomial $X^n + 1$ does not split into too many irreducible elements modulo p [LS17, Theorem 1.1].

The definition of $\mathcal{C}$ implies that the ℓ_2-norm of each row of $\mathsf{Rot}(c_1)$ and $\mathsf{Rot}_m(c_1)$ is exactly $\sqrt{\alpha}$. Therefore, each row of

$$\mathsf{Rot}_m(c_1) \cdot \mathbf{S} = \begin{bmatrix} \mathsf{Rot}(c_1) \cdot \mathbf{S}_1 \\ \vdots \\ \mathsf{Rot}(c_1) \cdot \mathbf{S}_m \end{bmatrix}$$

has ℓ_2-norm at most $\sqrt{\alpha} \cdot s_1(\mathbf{S}_i) \leq \sqrt{\alpha} \cdot \mathsf{s}_{\mathsf{part}}$. Since there are nm rows in $\mathbf{S}$, it implies that

$$\|\mathsf{Rot}_m(c_1) \cdot \mathbf{S}\| \leq \mathsf{s}_{\mathsf{part}} \cdot \sqrt{\alpha nm}. \tag{22}$$

Similarly, because each column of $\mathbf{C}_2$ contains only 0/1 coefficients, we know that its maximum ℓ_2-norm is $\sqrt{k}$. Because $\mathbf{C}_2$ has ℓ columns, we obtain

$$\|\mathbf{S} \cdot \mathbf{C}_2\| \leq \mathsf{s} \cdot \sqrt{k\ell}. \tag{23}$$

Equations (22) and (23) will be useful later when we are bounding the size of the pre-image that can be extracted from the Prover.

Rejection Sampling. In our zero-knowledge proof, the prover will want to output matrices $\mathbf{Z}_1, \mathbf{Z}_2$ whose distributions should be independent of the secret matrix $\mathbf{S}$. During the protocol, the prover obtains $\mathbf{Z}'_i = \mathbf{B}_i + \mathbf{Y}_i$ where $\mathbf{B}_i$ depends on the secret $\mathbf{S}$ and $\mathbf{Y}_i$ is a "masking" matrix each of whose coefficients is a discrete Gaussian with standard deviation σ. To remove the dependency of $\mathbf{Z}'_i$ on $\mathbf{B}_i$, we use the rejection sampling procedure from [Lyu12] in Algorithm 1, which has the properties described in Lemma 3.1.

Lemma 3.1 ([Lyu12]). *Let $\mathbf{B} \in \mathbb{Z}^{n \times m}$ be any matrix. Consider a procedure that samples a $\mathbf{Y} \leftarrow D_\sigma^{n \times m}$ and then returns the output of $\mathsf{RejectionSample}(\mathbf{Z} := \mathbf{Y} + \mathbf{B}, \mathbf{B}, \sigma, \rho)$ where $\sigma \geq \frac{12}{\ln \rho} \cdot \|\mathbf{B}\|$. The probability that this procedure outputs 1 is within 2^{-100} of $1/\rho$. The distribution of $\mathbf{Z}$, conditioned on the output being 1, is within statistical distance of 2^{-100} of $D_\sigma^{n \times m}$.*

Algorithm 1 $\mathsf{RejectionSample}(\mathbf{Z}, \mathbf{B}, \sigma, \rho)$

$u \leftarrow [0, 1)$
if $u > \frac{1}{\rho} \cdot \exp\left(\frac{-2\langle \mathbf{Z}, \mathbf{B}\rangle + \|\mathbf{B}\|^2}{2\sigma^2}\right)$ **then**
  return 0
else
  return 1
end if

Checking Bounds. In the protocol, the verifier will accept if and only if the values $\mathbf{Z}_1, \mathbf{Z}_2$, sent by the prover in the last round, are bounded in their norm as defined in Algorithm 2. The reader is encouraged, at this point, to skip to Section 3.2 and come back to Algorithm 2 and Proposition 3.2 after becoming familiar with the main protocol in Figure 1.

Algorithm 2 $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$

Output 1 if and only if the following 3 conditions are satisfied:

1. The ℓ_2-norm of every row of $\mathbf{Z}_1$ is at most $\sqrt{2k} \cdot \sigma_1$.
2. Every entry of $\mathbf{Z}_2$ has magnitude at most $7 \cdot \sigma_2$.
3. If $\mathbf{Z}_2 \in \mathbb{Z}^{nm \times \ell}$ is written as $\begin{bmatrix} \mathbf{Z}_2^{(1)} \\ \vdots \\ \mathbf{Z}_2^{(m)} \end{bmatrix}$ for $\mathbf{Z}_2^{(i)} \in \mathbb{Z}^{n \times \ell}$, then every column of every $\mathbf{Z}_2^{(i)}$ has ℓ_2-norm at most $\sqrt{2n} \cdot \sigma_2$.

We now show that, if $\mathbf{Z}_1, \mathbf{Z}_2$ are generated as in the protocol, then $\mathsf{IsSmall}$ will output 1 with high probability when the parameters have bounds that occur in practical applications.

Proposition 3.2. *Let $\mathbf{Z}_1 \in \mathbb{Z}^{nm \times k}, \mathbf{Z}_2 \in \mathbb{Z}^{nm \times \ell}$ have entries that are distributed according to D_{σ_1} and D_{σ_2} respectively. If $\ell < 1000, m < 100, n \geq 128, mn < 2^{14}$ and $k > 100$, then $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$ will return 1 with probability $> 1 - 2^{-8}$.*

Proof. Each row of $\mathbf{Z}_1$ has dimension k, so by (18) the probability that its ℓ_2-norm is greater than $\sqrt{2k} \cdot \sigma_1$ is less than $2^{-k/4}$. By a union bound, the probability that any of the nm rows has norm greater than $\sqrt{2k} \cdot \sigma_1$ is $nm \cdot 2^{-k/4}$.

By (17), a coefficient of $\mathbf{Z}_2$ will have magnitude greater than $7 \cdot \sigma_2$ with probability less than $2 \cdot e^{-49/2} \approx 2^{-34}$. Thus the probability that at least one of the $nm\ell$ coefficients will be be greater than $7 \cdot \sigma_2$ is less than $nm\ell \cdot 2^{-34}$.

Every column of $\mathbf{Z}_2^{(i)}$ is of dimension n, and so by (18), we have that the probability that a column will have norm greater than than $\sqrt{2n} \cdot \sigma_2$ is less than $2^{-n/4}$. Each $\mathbf{Z}_2^{(i)}$ has ℓ columns and there are m such matrices, thus the probability that at least one column will have norm larger than $2^{-n/4}$ is less than $\ell m \cdot 2^{-n/4}$.

The probability that $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$ will return 1 is therefore greater than

$$1 - (nm \cdot 2^{-k/4} + nm\ell \cdot 2^{-34} + \ell m \cdot 2^{-n/4}).$$

With the parameter bounds in the statement of the lemma, the above probability is greater than $1 - (2^{-9} + 2^{-10} + 2^{-15}) > 1 - 2^{-8}$. □

A list of all the important parameters of the protocol, as introduced above, can be found in Table 1.

Parameter	Use
n	Degree of the polynomial that defines R
m	Number of columns in $\mathbf{A}$
d	Number of rows in $\mathbf{A}$
k	Number of equations being simultaneously proven
ℓ	Number of columns in the second challenge
α	Number of non-zero entries in the first challenge $\left(\mathrm{argmin}_x(\binom{n}{x} \cdot 2^x > 2^{256})\right)$
$\mathcal{C}$	The set of first challenges (as in (21))
s	Spectral norm of the secret (as in (19))
$\mathsf{s}_{\mathsf{part}}$	Maximum spectral norm of sub-matrices of the secret (as in (20))
ρ_1, ρ_2	Inverses of the accepting probabilities of the rejection sampling
σ_1	Standard deviation of coefficients of $\mathbf{Y}_1$ (and of the transmitted $\mathbf{Z}_1$): $\left(\frac{12}{\ln \rho_1} \cdot \mathsf{s}_{\mathsf{part}} \cdot \sqrt{nm\alpha}\right)$
σ_2	Standard deviation of coefficients of $\mathbf{Y}_2$ (and of the transmitted $\mathbf{Z}_2$): $\left(\frac{12}{\ln \rho_2} \cdot \mathsf{s} \cdot \sqrt{k\ell}\right)$

Table 1. Parameters of the proof.

3.2 The Zero-Knowledge Proof

The complete protocol $\mathcal{P}_{\mathsf{3Rounds}}$ can be found in Figure 1.

Correctness. If the prover follows the protocol, then

$$\begin{aligned}\mathbf{A} \cdot \mathbf{Z}_1 &= \mathbf{A} \cdot \mathsf{Rot}_m(c_1) \cdot \mathbf{S} + \mathbf{A} \cdot \mathbf{Y}_1 \\ &= \mathsf{Rot}_d(c_1) \cdot \mathbf{A} \cdot \mathbf{S} + \mathbf{A} \cdot \mathbf{Y}_1 \\ &= \mathsf{Rot}_d(c_1) \cdot \mathbf{T} + \mathbf{W}_1 \bmod p,\end{aligned}$$

where the second equality follows from (15). We also have

$$\mathbf{A} \cdot \mathbf{Z}_2 = \mathbf{A} \cdot \mathbf{S} \cdot \mathbf{C}_2 + \mathbf{A} \cdot \mathbf{Y}_2 = \mathbf{T} \cdot \mathbf{C}_2 + \mathbf{W}_2 \bmod p.$$

Thus an honest verifier will always accept a non-aborting transcript.

To compute the probability that the prover does not abort, observe that (22) and the definition of σ_1 allow us to use Lemma 3.1 to conclude that $u_1 = 1$ with probability $\approx 1/\rho_1$. Similarly, (23) and the definition of σ_2 allow us to use Lemma 3.1 to conclude that $u_2 = 1$ with probability $\approx 1/\rho_2$. If $u_1 = u_2 = 1$, this implies that $\mathbf{Z}_1, \mathbf{Z}_2$ are distributed (within statistical distance 2^{-100}) as $D_{\sigma_1}^{nm \times k}$ and $D_{\sigma_2}^{nm \times \ell}$, respectively. Then Proposition 3.2 says that $u_3 = 1$ with probability at least $1 - 2^{-8}$. Therefore the probability that the prover does not abort is at least

$$\frac{1}{\rho_1 \cdot \rho_2} \cdot (1 - 2^{-8}).$$

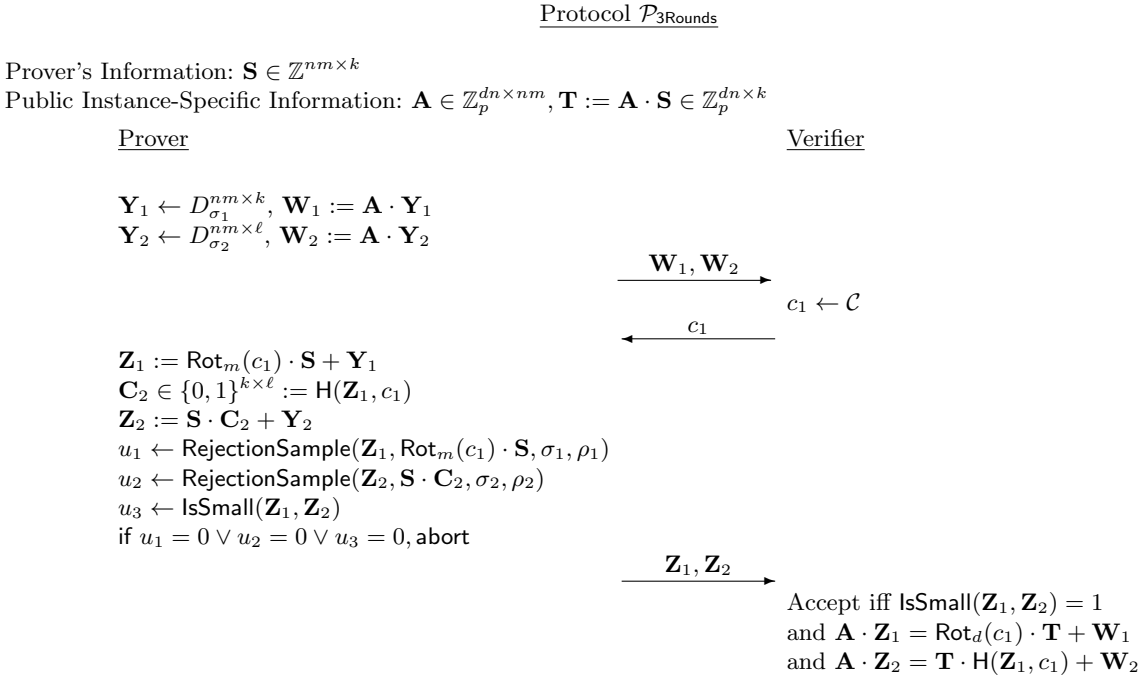

Fig. 1. 3-Round Proof. $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{k \times \ell}$ is a cryptographic hash function modeled as a random oracle.

Zero Knowledge. We will now prove that our protocol is honest-verifier zero-knowledge. More concretely, we only show that the protocol is zero-knowledge when the prover does not abort prior to sending $\mathbf{Z}_1, \mathbf{Z}_2$. The reason that this is enough for practical purposes is that HVZK Σ-protocols are first converted into non-interactive proofs via the Fiat-Shamir transform. The non-interactive protocol (Algorithm 3 in Section A.1) generates challenges c_1 as the hash of $\mathbf{W}_i$ and $\mathbf{T}$, and otherwise repeats the prover's part of the protocol until a non-abort occurs and the Prover outputs the transcript $\mathbf{Z}_1, \mathbf{Z}_2, c_1$. Therefore only the non-aborting transcripts will be seen, and thus only they need to be simulated. One can also, using the standard technique of sending commitments of the $\mathbf{W}_1, \mathbf{W}_2$ and only opening them in case a non-abort occurs, make the interactive protocol zero-knowledge. We give more details in Appendix A.

Lemma 3.3. *There exists an algorithm $\mathcal{S}$ that, on input $(\mathbf{A}, \mathbf{T})$, generates transcripts*

$$(\mathbf{W}_1, \mathbf{W}_2, c_1, \mathbf{C}_2, \mathbf{Z}_1, \mathbf{Z}_2)$$

which are within statistical distance 2^{-99} away from the non-aborting transcripts of the protocol in Figure 1.

Proof. Consider the following algorithm $\mathcal{S}$:

1. Sample $\mathbf{Z}_1 \leftarrow D_{\sigma_1}^{nm \times k}$, $c_1 \leftarrow \mathcal{C}$, and set $\mathbf{W}_1 := \mathbf{A} \cdot \mathbf{Z}_1 - \mathsf{Rot}_d(c_1) \cdot \mathbf{T}$.
2. Sample $\mathbf{Z}_2 \leftarrow D_{\sigma_2}^{nm \times \ell}$, set $\mathbf{C}_2 := \mathsf{H}(\mathbf{Z}_1, c_1)$ and $\mathbf{W}_2 := \mathbf{A} \cdot \mathbf{Z}_2 - \mathbf{T} \cdot \mathbf{C}_2$.
3. Output $(\mathbf{W}_1, \mathbf{W}_2, c_1, \mathbf{C}_2, \mathbf{Z}_1, \mathbf{Z}_2)$ if $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2) = 1$, otherwise go to Step 1.

We already showed in the section on "Correctness" that in the real protocol when no abort occurs, the distributions $\mathbf{Z}_1$ and $\mathbf{Z}_2$ are distributed within statistical distance 2^{-100} of $D_{\sigma_1}^{nm\times k}$ and $D_{\sigma_2}^{nm\times \ell}$, respectively (and independent of $\mathbf{S}_1, c_1, \mathbf{S}_2, \mathbf{C}_2$). Since $\mathbf{W}_1$ is completely determined by $\mathbf{A}, \mathbf{T}, \mathbf{Z}_1$, and c_1, the distribution $(\mathbf{Z}_1, c_1, \mathbf{W}_1)$ on Line 1 of $\mathcal{S}$ is within 2^{-100} of the distribution of these variables in the actual protocol.

The variable $\mathbf{C}_2$ is determined by $\mathbf{Z}_1$ and c_1 and $\mathbf{W}_2$ is again completely determined by $\mathbf{A}, \mathbf{T}, \mathbf{Z}_2$, and $\mathbf{C}_2$. Therefore the distribution of $(\mathbf{Z}_2, \mathbf{C}_2, \mathbf{W}_2)$ on Line 2 of $\mathcal{S}$ is within 2^{-100} of the distribution of these variables in the actual protocol.

The deterministic procedure $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$ is the same in the actual protocol and in $\mathcal{S}$ and therefore running it does not increase the statistical distance of the output. Therefore the statistical distance of the output of $\mathcal{S}$ is within $2 \cdot 2^{-100}$ of that in the non-aborting run of the real protocol. □

Soundness. We now move to proving the soundness of the protocol. The proof will be done in three steps. Lemma 3.4 is a standard argument following [Dam10] that shows that if a Prover runs in time t and succeeds with probability ϵ, then there is an extractor who, in time $O(t/\epsilon)$ and constant probability, can extract some linear equations relating the public key parts $\mathbf{A}$ and $\mathbf{T}$. In Theorem 3.5, we then show that if there is an algorithm who can extract such information, then he can either recover a pre-image of $\mathbf{T}$ or solve the Module-SIS problem for $\mathbf{A}$. The proof of Theorem 3.5 uses the information-theoretic Lemma 3.6 that proves that if H is a purely-random function to which the Prover only has black-box access to, then the recovered pre-image in Theorem 3.5 has small coefficients.

Lemma 3.4. *If there is a Prover $\mathcal{P}$ that succeeds in protocol $\mathcal{P}_{\mathsf{3Rounds}}$ in time t with probability $\epsilon > 2^{-254}$, then there exists an extractor $\mathcal{E}^{\mathcal{P}}$ that outputs $\mathbf{Z}_1, \mathbf{Z}_1', c_1 \neq c_1', \mathbf{Z}_2, \mathbf{Z}_2'$ with probability $\epsilon' \geq 1/8$ such that*

$$\mathbf{A} \cdot (\mathbf{Z}_1 - \mathbf{Z}_1') = \mathsf{Rot}_d(c_1 - c_1') \cdot \mathbf{T} \tag{24}$$

$$\mathbf{A} \cdot (\mathbf{Z}_2 - \mathbf{Z}_2') = \mathbf{T} \cdot (\mathsf{H}(\mathbf{Z}_1, c_1) - \mathsf{H}(\mathbf{Z}_1', c_1')). \tag{25}$$

Moreover, for each $s \in \mathbb{N}$ the extractor runs in time at most $t' = 12st/\epsilon$ except with probability 2^{-s+1}.

Proof. Consider the following algorithm $\mathcal{E}^{\mathcal{P}}$:

(1) Sample randomness $r \leftarrow \{0,1\}^{O(\lambda)}$ uniformly at random for the Prover $\mathcal{P}$. Then run $\mathcal{P}$ on r, until it outputs $\mathbf{W}_1, \mathbf{W}_2$.
(2) Send a random challenge $c_1 \leftarrow \mathcal{C}$ to the Prover $\mathcal{P}$. If $\mathcal{P}$ does not output $\mathbf{Z}_1, \mathbf{Z}_2$ such that $(\mathbf{W}_1, \mathbf{W}_2, c_1, \mathbf{Z}_1, \mathbf{Z}_2)$ is an accepted transcript, then go to step (1).
  (3.1) Rewind $\mathcal{P}$ for the same r until the challenge was sent. Send a new challenge $c_1' \leftarrow C \setminus \{c_1\}$ to $\mathcal{P}$.
  (3.2) If $\mathcal{P}$ outputs $\mathbf{Z}_1', \mathbf{Z}_2'$ such that $(\mathbf{W}_1, \mathbf{W}_2, c_1', \mathbf{Z}_1', \mathbf{Z}_2')$ is an accepted transcript then output $\mathbf{Z}_1, \mathbf{Z}_2, c_1, c_1', \mathbf{Z}_1', \mathbf{Z}_2'$ and terminate.
  (3.3) Sample $c \leftarrow \mathcal{B}_{\epsilon/16}$. If $c = 1$ then abort, otherwise go to step (3.1).[7]

[7] $\mathcal{B}_{\epsilon/16}$ is the Bernoulli probability distribution that outputs 1 with probability $\epsilon/16$ and 0 otherwise. The purpose of this step is for the algorithm to abort in case he selected a challenge c_1 for which it is very unlikely that the Prover will be able to answer a second challenge.

Damgård showed in [Dam10, Theorem 1] that $\mathcal{E}^{\mathcal{P}}$ outputs two transcripts

$$(\mathbf{W}_1, \mathbf{W}_2, c_1, \mathbf{Z}_1, \mathbf{Z}_2), (\mathbf{W}_1, \mathbf{W}_2, c_1', \mathbf{Z}_1', \mathbf{Z}_2')$$

that satisfy the verification equations with probability at least $1/8$. Equations (24), (25) are then obtained from the output of $\mathcal{E}^{\mathcal{P}}$ by writing down the verification equations and eliminating $\mathbf{W}_1, \mathbf{W}_2$.

We will now show that $\mathcal{E}^{\mathcal{P}}$ runs in expected time $O(t/\epsilon)$. First, it is straightforward to see that the extractor finds $\mathbf{Z}_1, \mathbf{Z}_2, c_1$ in expected time t/ϵ since the event of obtaining a first valid response follows a geometric distribution. The probability that $\mathcal{E}^{\mathcal{P}}$ proceeds to Step (3.1) in more than $k_1 = \ln(2)s/\epsilon$ steps is less than

$$(1-\epsilon)^{k_1} < \exp(-\epsilon k_1) = 2^{-s}.$$

Similarly, it might happen that the prover $\mathcal{P}$ will never respond with a second valid response on a fixed random tape r and $\mathcal{E}^{\mathcal{P}}$ never terminates in step (3.2). By the same argument as before, the event of abort in step (3.3) follows a geometric distribution. Then after at most $k_2 \approx 11s/\epsilon$ iterations the extractor will have terminated except with probability 2^{-s}. By a union bound, $\mathcal{E}^{\mathcal{P}}$ will terminate in time at most $\approx 12st/\epsilon$ except with probability 2^{-s+1}. □

Theorem 3.5. *Suppose that there exists an extractor $\mathcal{E}^{\mathcal{P},\mathsf{H}}$ making q queries to $\mathsf{H}(\cdot)$ that succeeds in producing the outputs from the statement of Lemma 3.4 in time t' with probability ϵ'. Then there exists an algorithm which, in time approximately t' and with probability at least $\epsilon' - q^2 \cdot 2^{-\ell-1} - 2^{-\ell}$, can output one of two things: either a short solution $\bar{\mathbf{S}}$ that satisfies $\mathbf{A} \cdot \bar{\mathbf{S}} = \mathbf{T} \bmod p$ with*

$$\|\bar{\mathbf{S}}\|_\infty \leq 2 \cdot \|\mathbf{Z}_2 - \mathbf{Z}_2'\|_\infty \leq 28 \cdot \sigma_2,$$

or a solution to ℓ_∞-norm Module-SIS$_{nd,nm,\beta}$ problem for

$$\beta = 2^{3/2} \cdot k \cdot \sigma_1 + 2^{5/2} \cdot \sqrt{\alpha n} \cdot \sigma_2.$$

Proof. We now describe the algorithm. Whenever $\mathcal{E}^{\mathcal{P},\mathsf{H}}$ queries the function H, our algorithm checks whether it was previously outputted, and if not, assigns a random value in $\{0,1\}^{k \times \ell}$ to the query. Once $\mathcal{E}^{\mathcal{P},\mathsf{H}}$ returns the values from Lemma 3.4, the algorithm outputs

$$\bar{\mathbf{S}} = \mathsf{Rot}_m^{-1}(c_1 - c_1') \cdot (\mathbf{Z}_1 - \mathbf{Z}_1'). \tag{26}$$

First, we see that

$$\mathbf{A} \cdot \bar{\mathbf{S}} = \mathsf{Rot}_d^{-1}(c_1 - c_1') \cdot \mathbf{A} \cdot (\mathbf{Z}_1 - \mathbf{Z}_1') = \mathbf{T},$$

where the first equality follows from (15) and the second from (24). Thus $\bar{\mathbf{S}}$ is indeed a pre-image of $\mathbf{T}$ and all that we have left to prove is that the coefficients of $\bar{\mathbf{S}}$ have small magnitudes.

For convenience, we define $\mathbf{C}_2 = \mathsf{H}(\mathbf{Z}_1, c_1)$ and $\mathbf{C}_2' = \mathsf{H}(\mathbf{Z}_1', c_1')$. Then plugging in $\mathbf{A} \cdot \bar{\mathbf{S}}$ into (25), we obtain

$$\mathbf{A} \cdot (\mathbf{Z}_2 - \mathbf{Z}_2') = \mathbf{A} \cdot \bar{\mathbf{S}} \cdot (\mathbf{C}_2 - \mathbf{C}_2'), \tag{27}$$

If $(\mathbf{Z}_2 - \mathbf{Z}_2') \neq \bar{\mathbf{S}} \cdot (\mathbf{C}_2 - \mathbf{C}_2')$, then we will show that we obtain a solution to Module-SIS. In particular, if $(\mathbf{Z}_2 - \mathbf{Z}_2') \neq \bar{\mathbf{S}} \cdot (\mathbf{C}_2 - \mathbf{C}_2')$, then due to the invertibility of $\mathsf{Rot}_m(c_1 - c_1')$, we also have that

$$\mathsf{Rot}_m(c_1 - c_1') \cdot (\mathbf{Z}_2 - \mathbf{Z}_2') \neq \mathsf{Rot}_m(c_1 - c_1') \cdot \bar{\mathbf{S}} \cdot (\mathbf{C}_2 - \mathbf{C}_2') = (\mathbf{Z}_1 - \mathbf{Z}_1') \cdot (\mathbf{C}_2 - \mathbf{C}_2'), \tag{28}$$

where the equality comes from the definition of $\bar{\mathbf{S}}$ in (26). If we similarly multiply (27) by $\mathsf{Rot}_m(c_1 - c_1')$ and apply the identity from (15), we obtain

$$\mathbf{A} \cdot \left((\mathsf{Rot}_m(c_1 - c_1')) \cdot (\mathbf{Z}_2 - \mathbf{Z}_2') - (\mathbf{Z}_1 - \mathbf{Z}_1') \cdot (\mathbf{C}_2 - \mathbf{C}_2') \right) = \mathbf{0}. \tag{29}$$

Equation (28) implies that some column of the above pre-image of $\mathbf{0}$ is non-zero, and we therefore have a solution to Module-SIS$_{nd,nm,\beta}$ for some value β.

To compute β, we need to compute the maximum coefficient of the pre-image of $\mathbf{0}$. Every row of $\mathbf{C}_2 - \mathbf{C}_2'$ has $-1/0/1$ coefficients, and therefore has norm at most $\sqrt{k}$. Since each row of $\mathbf{Z}_1$ and $\mathbf{Z}_1'$ has norm at most $\sqrt{2k} \cdot \sigma_1$, the Cauchy-Schwartz and triangular inequalities imply that each coefficient of the product $(\mathbf{Z}_1 - \mathbf{Z}_1') \cdot (\mathbf{C}_2 - \mathbf{C}_2')$ has magnitude at most $2^{3/2} \cdot k \cdot \sigma_1$.

We now move on to computing a bound on the coefficient size of $\mathsf{Rot}_m(c_1 - c_1') \cdot (\mathbf{Z}_2 - \mathbf{Z}_2')$. Using the same notation as in Algorithm 2, we rewrite the preceding product as

$$\mathsf{Rot}_m(c_1 - c_1') \cdot (\mathbf{Z}_2 - \mathbf{Z}_2') = \begin{bmatrix} \mathsf{Rot}(c_1 - c_1') \cdot \left(\mathbf{Z}_2^{(1)} - \mathbf{Z}_2^{\prime(1)} \right) \\ \vdots \\ \mathsf{Rot}(c_1 - c_1') \cdot \left(\mathbf{Z}_2^{(m)} - \mathbf{Z}_2^{\prime(m)} \right) \end{bmatrix}.$$

By the triangular inequality, for all i, every column of $\left(\mathbf{Z}_2^{(i)} - \mathbf{Z}_2^{\prime(i)} \right)$ has norm at most $2^{3/2} \cdot \sqrt{n} \cdot \sigma_2$ and each row of $\mathsf{Rot}(c_1 - c_1')$ has norm at most $2\sqrt{\alpha}$. By the Cauchy-Schwartz inequality, it then follows that every coefficient of $\mathsf{Rot}_m(c_1 - c_1') \cdot (\mathbf{Z}_2 - \mathbf{Z}_2')$ has magnitude at most $2^{5/2} \cdot \sqrt{\alpha n} \cdot \sigma_2$.

Equations (28) and (29) therefore imply that we can extract a solution to Module-SIS$_{nd,nm,\beta}$ for

$$\beta = 2^{3/2} \cdot k \cdot \sigma_1 + 2^{5/2} \cdot \sqrt{\alpha n} \cdot \sigma_2.$$

If we assume that such a Module-SIS problem instance is hard, it must therefore be that

$$(\mathbf{Z}_2 - \mathbf{Z}_2') = \bar{\mathbf{S}} \cdot (\mathbf{C}_2 - \mathbf{C}_2'). \tag{30}$$

In Lemma 3.6, we will show that because $\mathbf{C}_2$ and $\mathbf{C}_2'$ are derived via a random oracle from $\bar{\mathbf{S}}$, it implies that the extractor $\mathcal{E}^{\mathcal{P},\mathsf{H}}$ making q queries to $\mathsf{H}(\cdot)$ (yet otherwise having possibly unlimited running time) has probability less than $q^2 \cdot 2^{-\ell-1} + 2^{-\ell}$ of outputting a $\bar{\mathbf{S}}$ with some coefficient larger than $2 \cdot \|(\mathbf{Z}_2 - \mathbf{Z}_2')\|_\infty$ that still satisfies (30).

This will complete the proof of the Theorem. □

Lemma 3.6. *Suppose that an algorithm $\mathcal{A}^{\mathsf{H}}$ makes q queries to $\mathsf{H}(\cdot)$. Then the probability, over the randomness of $\mathsf{H}(\cdot)$, that $\mathcal{A}^{\mathsf{H}}$ succeeds in producing $\mathbf{Z}_1, \mathbf{Z}_1', c_1 \neq c_1'$ defining an*

$$\mathbf{M} \triangleq \mathsf{Rot}_m^{-1}(c_1 - c_1') \cdot (\mathbf{Z}_1 - \mathbf{Z}_1')$$

that satisfies

$$\|\mathbf{M}\|_\infty > 2 \cdot \|\mathbf{M} \cdot (\mathsf{H}(\mathbf{Z}_1, c_1) - \mathsf{H}(\mathbf{Z}_1', c_1'))\|_\infty \tag{31}$$

is at most $q^2 \cdot 2^{-\ell-1} + 2^{-\ell}$.

Proof. We first observe that an $\mathcal{A}^{\mathsf{H}}$ that never makes the queries $\mathsf{H}(\mathbf{Z}_1, c_1)$ and $\mathsf{H}(\mathbf{Z}'_1, c'_1)$ has probability at most $2^{-\ell}$ of satisfying (31). This follows directly from Corollary 2.4.

An algorithm $\mathcal{A}^{\mathsf{H}}$ making q queries to H who succeeds (with probability larger than $2^{-\ell}$) must therefore make q queries

$$\mathbf{C}_2^{(i)} = \mathsf{H}\left(\mathbf{Z}_1^{(i)}, c_1^{(i)}\right)$$

for $i \in [q]$ and then output $(\mathbf{Z}_1, c_1) = \left(\mathbf{Z}_1^{(i)}, c_1^{(i)}\right)$ and $(\mathbf{Z}'_1, c'_1) = \left(\mathbf{Z}_1^{(j)}, c_1^{(j)}\right)$ that he queried that satisfy (31). Without loss of generality, we can assume that $i < j$. We will now bound the probability with which $\mathcal{A}^{\mathsf{H}}$ can succeed after making query j.

After $j-1$ queries of the form $\mathsf{H}\left(\mathbf{Z}_1^{(i)}, c_1^{(i)}\right)$ for $i \in [j-1]$, $\mathcal{A}^{\mathsf{H}}$ has $j-1$ equations of the form $\mathbf{C}_2^{(i)} = \mathsf{H}\left(\mathbf{Z}_1^{(i)}, c_1^{(i)}\right)$. The algorithm can "succeed" after making query j, if for some $i < j$, we have

$$\left\|\mathbf{M}^{(i,j)}\right\|_\infty > 2 \cdot \left\|\mathbf{M}^{(i,j)} \cdot \left(\mathbf{C}_2^{(j)} - \mathbf{C}_2^{(i)}\right)\right\|_\infty$$

where

$$\mathbf{M}^{(i,j)} \triangleq \mathsf{Rot}_m^{-1}\left(c_1^{(j)} - c_1^{(i)}\right) \cdot \left(\mathbf{Z}_1^{(j)} - \mathbf{Z}_1^{(i)}\right).$$

Observe that the value for $\mathbf{C}_2^{(j)}$ will be chosen uniformly at random from $\{0,1\}^{k\times\ell}$ on this j^{th} query. Corollary 2.4 then tells us that

$$\Pr_{\mathbf{C}_2^{(j)} \leftarrow \{0,1\}^{k\times\ell}}\left[\left\|\mathbf{M}^{(i,j)} \cdot \left(\mathbf{C}_2^{(j)} - \mathbf{C}_2^{(i)}\right)\right\|_\infty < \frac{1}{2} \cdot \left\|\mathbf{M}^{(i,j)}\right\|_\infty\right] \leq 2^{-\ell}.$$

By the union bound, we obtain that

$$\Pr_{\mathbf{C}_2^{(j)} \leftarrow \{0,1\}^{k\times\ell}}\left[\exists\, i < j\,, s.t.\, \left\|\mathbf{M}^{(i,j)} \cdot \left(\mathbf{C}_2^{(j)} - \mathbf{C}_2^{(i)}\right)\right\|_\infty < \frac{1}{2} \cdot \left\|\mathbf{M}^{(i,j)}\right\|_\infty\right] \leq (j-1) \cdot 2^{-\ell}.$$

Because there are a total of q queries to H, and so j ranges from 1 to q, another union bound states that

$$\begin{aligned}\Pr_{\substack{\mathbf{C}_2^{(t)} \leftarrow \{0,1\}^{k\times\ell} \\ t\in[q]}}&\left[\exists i < j\,, s.t.\, \left\|\mathbf{M}^{(i,j)} \cdot \left(\mathbf{C}_2^{(j)} - \mathbf{C}_2^{(i)}\right)\right\|_\infty < \frac{1}{2} \cdot \left\|\mathbf{M}^{(i,j)}\right\|_\infty\right] \\ &\leq \sum_{j=1}^{q} (j-1) \cdot 2^{-\ell} < q^2 \cdot 2^{-\ell-1},\end{aligned}$$

and this completes the proof. □

4 Practical Considerations and Sample Instantiations

If we want to simultaneously prove k equations using the challenge matrix $\mathbf{C}_2 \in \{0,1\}^{k\times\ell}$, then the entire proof will consist of $k+\ell$ vectors. It can be seen from Theorem 3.5 and Table 1 that, the larger k and ℓ are, the larger the "soundness slack" of the solution will be. It is therefore always advantageous to set ℓ to be as small as possible. The combination of Theorem 3.5 with Lemma 3.4

implies that if there is a Prover who runs in time t and succeeds with probability ϵ while making q queries to the random oracle H, then we can extract either a solution or solve the related Module-SIS instance in time $\approx 12t/\epsilon$ with probability $\approx \frac{1}{8} - q^2 \cdot 2^{-\ell-1}$. Setting $\ell = 2\log q + 5$ would make the preceding probability constant.

If one assumes that the Prover is classical, then q is at most 2^{128}, and therefore one could set $\ell > 260$. It is difficult to say anything precise about quantum provers which are allowed to make queries to H in superposition. Also, since Lemma 3.4 implicitly uses rewinding, an extractor cannot actually use it to retrieve the pre-image $\bar{\mathbf{S}}$. In this case, the common approach in the literature on lattice-based zero-knowledge proofs of knowledge is to simply double the length of the hash function range to protect against Grover's attack and then assume that this makes the classically-secure protocol also secure against quantum attacks. There have not yet been any natural counterexamples to such constructions. In the same spirit, it is then reasonable to double the allowed number of queries that the Prover makes. Thus, for 128-bits of security, we should set $q = 2^{256}$ and $\ell > 516$.

When converting our Σ-protocol into a non-interactive version via the Fiat-Shamir transform (obtaining Algorithms 3 and 4), one loses an additional factor of q in the success probability of the extraction algorithm (cf. [AABN08]). But because the Fiat-Shamir transformation has been widely used for several decades, and we still do not have examples where this loss in the proof actually translates into practice, one usually ignores this factor q when choosing practical parameter. One could also make an argument that the q^2 factor in Theorem 3.5 is an artifact of the proof and is due to the fact that Lemma 3.6 is allowing for an algorithm which can query H q times and then consider all the $\binom{q}{2}$ differences of these queries (which is what happens in the extraction), whereas the real Prover must answer one of the q responses of H. It may therefore be reasonable to only set $\ell > 256$ rather than doubling it. In particular, if we did the proof directly on the 5-round protocol in Figure 2, then we would only have a linear factor of q. Our guess is that, just like the loss of the factor of q when transforming a Σ-protocol to a NIZK, the loss of the factor of q when converting our 5-round protocol into a Σ-protocol doesn't have any practical implications. But since our protocol is new, it may be wise to err on the side of caution for now. In the next section, we will set parameters for both $\ell = 261$ and $\ell = 517$ to illustrate the slight difference that this choice causes.

4.1 Sample Parameters

There are many different applications where one may want to apply our result, so we will just give a simple instantiation to give a sense for the size of the parameters and for comparison with previous works. Suppose that we have k equations of the form $\mathbf{A} \cdot \mathbf{s} = \mathbf{t}$ where the coefficients of $\mathbf{s}$ are chosen from D_σ. The linear equations can be written in matrix form as $\mathbf{A} \cdot \mathbf{S} = \mathbf{T}$.

From [Ver10] (also see [MP12, Lemma 2.9]), we can put a sharp bound on the spectral norm of matrices that are chosen like $\mathbf{S}$ as

$$\Pr_{\mathbf{S} \leftarrow D_\sigma^{x \times k}}[s_1(\mathbf{S}) > \sigma \cdot (\sqrt{x} + \sqrt{k} + r)] < \exp\left(-\pi r^2\right). \tag{32}$$

Plugging in $x = nm$ and $x = n$ into the above equation with $r = 5$ yields a bound on s and $\mathsf{s}_{\mathsf{part}}$ that holds with probability greater than $1 - 2^{-110}$. In Table 2 we give instantiations of our scheme such that recovering $\mathbf{S}$ from $\mathbf{A}$ and $\mathbf{T}$, as well as fulfilling either of the two consequences of Theorem 3.5 is conjectured to be approximately 2^{-128}-hard for quantum algorithms using the cryptanalysis from [DLL+17]. The only distinction in the first four sets of parameters are the values

of k and ℓ. In set 5, we change ρ_1, ρ_2 in a way that makes the algorithm run twice as slow (because it requires twice as many repetitions), but allows one to reduce the proof size versus parameter set 4 in case we have more equations to prove simultaneously.

	Set 1	Set 2	Set 3	Set 4	Set 5
n	256	256	256	256	256
d	7	7	7	7	7
m	14	14	14	14	14
$\log_2 p$	36	36	36	36	36
σ	3	3	3	3	3
k	250	500	250	500	1000
ℓ	261	261	517	517	517
α	60	60	60	60	60
$\mathsf{s}, \mathsf{s}_{\mathsf{part}}$	$(242, 110)$	$(262, 130)$	$(242, 110)$	$(262, 130)$	$(289, 157)$
ρ_1, ρ_2	$(\sqrt{3}, \sqrt{3})$	$(\sqrt{3}, \sqrt{3})$	$(\sqrt{3}, \sqrt{3})$	$(\sqrt{3}, \sqrt{3})$	$(\sqrt{6}, \sqrt{6})$
proof size / equation	21KB	16KB	32KB	22KB	16KB
commit size / equation	8KB	8KB	8KB	8KB	8KB
Hermite factor original proof	1.003	1.003	1.003	1.003	1.003
Hermite factor SIS	1.0036	1.0038	1.0037	1.0038	1.0039
slack	$2^{23.5}$	$2^{24.2}$	$2^{24.1}$	$2^{24.7}$	$2^{24.6}$

Table 2. Sample Parameters

4.2 Efficiency

We will now give a rough estimate of the efficiency of our protocol, using parameter set 4 in Table 2. We did not implement our protocol, but the basic operations that it uses are described in various other papers and allow us to obtain a ballpark estimate of the running time.

Computing each column of the matrix $\mathbf{W}_1$ and $\mathbf{W}_2$ requires the multiplication of m polynomials in R_p, thus requiring a total of $mk + m\ell$ polynomial multiplications. Computing $\mathbf{Z}_1$ then requires a further mk multiplications in R_p, for a total of $2mk+ml$ multiplications. If the degree of the ring R_p is $n = 256$, then we know from [LS17] that each operation requires about $30K$ cycles when $p \approx 2^{29}$. Since our prime p is larger and may require more than 64 bits for multiplication, we can double the number of cycles to $60K$. For the fourth set of parameters in Table 2, this implies that the total number of cycles spent on polynomial multiplication will be around one billion cycles. Since our proof will need to be repeated approximately $\rho_1 \cdot \rho_2 = 3$ times, we can roughly approximate that we will spend 1.5 seconds doing polynomial multiplications on a $2.5GHz$ machine.

We also need to generate $nmk + nm\ell \approx 3.7 \cdot 10^6$ discrete Gaussians for every iteration. Extrapolating from [MW17, Figure 1], one can generate approximately $6 \cdot 10^6$ discrete Gaussians per second. We will need to generate $3 \cdot 3.7 \cdot 10^6$ discrete Gaussians for the five expected runs of our algorithm, which would require approximately 1.85 seconds.

Another operation is the multiplication of $\mathbf{S} \cdot \mathbf{C}_2$ in the creation of $\mathbf{Z}_2$. Asymptotically, this is the most expensive operation requiring $\Theta(nmk\ell)$ operations. But matrix multiplications of this form are extremely efficient in practice – on a 2.5 GHz machine, the linear algebra package Scilab can perform about 15 multiplications of $\mathbf{S} \cdot \mathbf{C}_2$ per second. This means that five such multiplications take $\frac{1}{3}$ of a second. If we then generously assume that all the other overhead in our protocol, such

as vector additions and hash function evaluations, take 0.75 seconds, then the total time for the protocol will require less than 4.5 seconds. Since we are amortizing over 500 proofs, this comes out to approximately 9 milliseconds per proof.

4.3 Comparison

We will now compare our protocol with the parameters given in [DL17] for the protocol that works over polynomial rings: if one is simultaneously proving approximately 8000 equations, then one requires $16m$ multiplications over R_p per proof and $16nm$ discrete Gaussians (see [DL17, Table 1]). Using the same numbers as in Section 4.2 and ignoring any overhead cost, we obtain approximately 11 milliseconds per sample. Due to the fact that the slack is smaller (see [DL17, Table 2]), it is reasonable to assume that for some applications one would be able to use a modulus that is less than 32 bits and so polynomial multiplications can be implemented more efficiently. [DL17] also has no requirement to choose a modulus such that X^n+1 does not split too much (which is necessary in our case, because we need all non-zero elements in $\mathcal{C}-\mathcal{C}$ to be invertible). It is therefore conceivable that the multiplication operation could take as little as 10000 cycles (i.e. 6 times faster), which would make the running time approximately 10 milliseconds per proof (with most of the time being used for the Gaussian sampling). Thus when one has enough samples available, the proof techniques in [BDLN16,CDXY17,DL17] are roughly as efficient as the one in this paper.

Decreasing the required number of samples, however, has a significant effect on the running time of the protocol in [DL17]. For example, decreasing the required number of proofs to 2200 ends up increasing the running time by a factor of 8, and decreasing to 1300 slows down the protocol by a factor of 128. For this latter parameter, the protocol in this paper is therefore over two orders of magnitude more efficient.

One could of course overcome the minimum number of samples requirement by simply adding "dummy" equations. This would, however, increase the overhead of the proof. Our proof sizes are a little longer than those in [DL17], but the difference is not large enough for it to be beneficial to use the latter scheme with an artificially-inflated number of secrets. The proofs from [BDLN16,CDXY17,DL17] have a smaller slack by around a factor of 8. The proof size given in [DL17, Table 2] would correspond to $n = 256$ and $m = 4$ in the current paper, which would not have 128-bit security against quantum attacks. One would need to increase m to 6 or 7 to have the same security, which would end up with proof sizes that are still around 30% smaller than those in the current paper. We should also point out that when using the protocol in this paper, one can increase $k > \ell$, and the amortized proof size will drop due to the fact that $\mathbf{Z}_2$ only depends on ℓ (see for example the difference between parameter Sets 1 and 2 as well as 4 and 5 in Table 2). Thus if we have more samples, it is possible to eliminate the 30% disadvantage in the proof size.

The conclusion is that when the number of proofs is large (over 7000), then the protocols derived from [BDLN16,CDXY17,DL17] are roughly of the same efficiency as the one in the present paper. On the other hand, as the number of samples decreases, the previous protocols slow down exponentially, and thus the protocol in the current paper becomes increasingly superior.

5 Open Problems

We think that the most natural open problem is if one can devise a practical protocol that can even further reduce the number of required equations until amortization applies. One can of course trivially halve the number of samples, which will simply cause the overhead to double. The open

question therefore is whether it's possible to reduce the required number of samples while not increasing the overhead (but possibly slightly increasing the slack). Since the overhead is directly tied to the variable ℓ, a very tempting solution to try and reduce ℓ would be to choose the matrix $\mathbf{C}_2$ differently so that the probability decreases below $\frac{1}{2}$. For example, one could think that by choosing the coefficients from $\{0, 1, 2, 3\}$ rather than from $\{0, 1\}$ one could reduce the probability to $\frac{1}{4}$. This is unfortunately not true: if the vector $\mathbf{s}$ consists of coefficients that are all near $p/2$, then no matter what the distribution of $\mathbf{c}$ is, the inner product $\langle \mathbf{s}, \mathbf{c} \rangle \bmod p$ will be near 0 (and thus an inner product of a large vector with $\mathbf{c}$ will be small) only depending on the parity of the sum of the coefficients in $\mathbf{c}$. Thus the obvious idea of just changing the distribution of $\mathbf{C}_2$ will not result in an improvement. Still, we do not discount the possibility that something more clever could be done with the matrix $\mathbf{C}_2$.

Another technique that sometimes helps, but we did not find a use for in this paper, is the idea from [BCK+14] to use challenges of the form $\pm X^i$ rather than just from the set $\{0, 1\}$. In previous works, this reduced the soundness error of a protocol over polynomial lattices from $1/2$ to $1/2n$ which requires less repetitions of a protocol proving one equation by a factor of $\log 2n$ [BCK+14] or reduces the number of required samples before amortization kicks in by approximately the same factor [DL17]; but we have not been able to find a use for it in this work. Trying to apply this technique is also a very tempting avenue of research since it could reduce the number of samples by a factor of $O(\log n)$ which would allow us to efficiently amortize over as few as 50 equations.

As mentioned in the introduction, we do not see a way to apply the techniques in this paper towards practical improvements in proving amortized equations over $\mathbb{Z}_p$ (thus improving [BDLN16,CDXY17,DL17] in the generic setting and reducing the soundness to the SIS problem), and so this remains an open problem. We also mention that there could be some constant-factor optimizations over the parameters in Theorem 3.5 – especially in the reduction from Module-SIS. For practical applications it would be particularly useful to reduce them because, as seen in the sample instantiation, they are larger than the coefficients of the extracted solution.

Acknowledgments

We would like to thank Michael Walter for discussions and clarifications about discrete Gaussian sampling from [MW17].

References

AABN08. Michel Abdalla, Jee Hea An, Mihir Bellare, and Chanathip Namprempre. From identification to signatures via the fiat-shamir transform: Necessary and sufficient conditions for security and forward-security. *IEEE Trans. Information Theory*, 54(8):3631–3646, 2008.

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

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.

BDLN16. Carsten Baum, Ivan Damgård, Kasper Green Larsen, and Michael Nielsen. How to prove knowledge of small secrets. In *CRYPTO*, pages 478–498, 2016.

BDOP16. Carsten Baum, Ivan Damgård, Sabine Oechsner, and Chris Peikert. Efficient commitments and zero-knowledge protocols from ring-sis with applications to lattice-based threshold cryptosystems. *IACR Cryptology ePrint Archive*, 2016:997, 2016.

BG14. Shi Bai and Steven D. Galbraith. An improved compression technique for signatures based on learning with errors. In *CT-RSA 2014*, pages 28–47, 2014.

BKLP15. Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In *ESORICS*, pages 305–325, 2015.

CD09. Ronald Cramer and Ivan Damgård. On the amortized complexity of zero-knowledge protocols. In *CRYPTO*, pages 177–191, 2009.

CDXY17. Ronald Cramer, Ivan Damgård, Chaoping Xing, and Chen Yuan. Amortized complexity of zero-knowledge proofs revisited: Achieving linear soundness slack. In *EUROCRYPT*, pages 479–500, 2017.

Dam10. Ivan Damgård. On Σ-protocols, 2010. `http://www.cs.au.dk/~ivan/Sigma.pdf`.

DDLL13. Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO (1)*, pages 40–56, 2013.

DL17. Rafaël Del Pino and Vadim Lyubashevsky. Amortization with fewer equations for proving knowledge of small secrets. *IACR Cryptology ePrint Archive*, 2017:280, 2017. To appear in CRYPTO 2017.

DLL$^+$17. Léo Ducas, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. CRYSTALS - dilithium: Digital signatures from module lattices. *IACR Cryptology ePrint Archive*, 2017:633, 2017.

DPSZ12. Ivan Damgård, Valerio Pastro, Nigel P. Smart, and Sarah Zakarias. Multiparty computation from somewhat homomorphic encryption. In *CRYPTO*, pages 643–662, 2012.

FS86. Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In *CRYPTO*, pages 186–194, 1986.

GLP12. Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *CHES*, pages 530–547, 2012.

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.

LM06. Vadim Lyubashevsky and Daniele Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, pages 144–155, 2006.

LMPR08. Vadim Lyubashevsky, Daniele Micciancio, Chris Peikert, and Alon Rosen. SWIFFT: A modest proposal for FFT hashing. In *FSE*, pages 54–72, 2008.

LN17. Vadim Lyubashevsky and Gregory Neven. One-shot verifiable encryption from lattices. In *EUROCRYPT*, pages 293–323, 2017.

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.

LS15. Adeline Langlois and Damien Stehlé. Worst-case to average-case reductions for module lattices. *Des. Codes Cryptography*, 75(3):565–599, 2015.

LS17. Vadim Lyubashevsky and Gregor Seiler. Partially splitting rings for faster lattice-based zero-knowledge proofs. Cryptology ePrint Archive, Report 2017/523, 2017. `http://eprint.iacr.org/2017/523`.

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.

MP12. Daniele Micciancio and Chris Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718, 2012.

MW17. Daniele Micciancio and Michael Walter. Gaussian sampling over the integers: Efficient, generic, constant-time. *IACR Cryptology ePrint Archive*, 2017:259, 2017. To appear in Crypto 2017.

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

Ver10. Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *CoRR*, abs/1011.3027, 2010.

A Alternative Proofs

In this section, we present alternative ways in which the proof in $\mathcal{P}_{\mathsf{3Rounds}}$ could be presented. Its most useful use in practice is as a NIZK proof that is derived via the Fiat-Shamir transform, and we present it in Section A.1. We also show that one can get rid of the random oracle assumption in $\mathcal{P}_{\mathsf{3Rounds}}$ by expanding it into a 5-round proof (Section A.2). In this latter section we also describe

some techniques that one can use to make the 3-round and 5-round proofs zero-knowledge proofs zero-knowledge even in the case that the prover aborts, as well as how to increase the completeness of the interactive protocol with only a minor increase in the communication complexity. We point out that all the techniques in this section are standard, so we just describe them here for convenience.

A.1 Non-Interactive Proof

A three-round honest-verifier zero-knowledge proof such as $\mathcal{P}_{\mathsf{3Rounds}}$ is used in practice by converting it to a non-interactive protocol using the Fiat-Shamir transform [FS86]. This requires an additional cryptographic hash function H', modeled as a random oracle, that maps $\{0,1\}^*$ to the domain of $\mathcal{C}$. We present the Prove and Verify protocols as Algorithms 3 and 4 with all the notation as in the $\mathcal{P}_{\mathsf{3Rounds}}$ protocol. Notice that the Prove algorithm only outputs valid transcripts and so partial transcripts that end in an abort are never revealed to the verifier. For this reason it was enough to prove zero-knowledge of non-aborting transcripts in Section 3.2.

Algorithm 3 $\mathsf{Prove}(\mathbf{A}, \mathbf{S})$. $\mathsf{H} : \{0,1\}^* \to \{0,1\}^{k \times \ell}$ and $\mathsf{H}' : \{0,1\}^* \to \mathcal{C}$ are cryptographic hash functions modeled as random oracles.

$\mathbf{Y}_1 \leftarrow D_{\sigma_1}^{nm \times k}$, $\mathbf{W}_1 := \mathbf{A} \cdot \mathbf{Y}_1$
$\mathbf{Y}_2 \leftarrow D_{\sigma_2}^{nm \times \ell}$, $\mathbf{W}_2 := \mathbf{A} \cdot \mathbf{Y}_2$
$c_1 := \mathsf{H}'(\mathbf{W}_1, \mathbf{W}_2, \mathbf{T})$
$\mathbf{Z}_1 := \mathsf{Rot}_m(c_1) \cdot \mathbf{S} + \mathbf{Y}_1$
$\mathbf{C}_2 \in \{0,1\}^{k \times \ell} := \mathsf{H}(\mathbf{Z}_1, c_1)$
$\mathbf{Z}_2 := \mathbf{S} \cdot \mathbf{C}_2 + \mathbf{Y}_2$
$u_1 \leftarrow \mathsf{RejectionSample}(\mathbf{Z}_1, \mathsf{Rot}_m(c_1) \cdot \mathbf{S}, \sigma_1, \rho_1)$
$u_2 \leftarrow \mathsf{RejectionSample}(\mathbf{Z}_2, \mathbf{S} \cdot \mathbf{C}_2, \sigma_2, \rho_2)$
$u_3 \leftarrow \mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$
if $u_1 = 0 \vee u_2 = 0 \vee u_3 = 0$, goto Step 1
return $(\mathbf{Z}_1, \mathbf{Z}_2, c_1)$

Algorithm 4 $\mathsf{Verify}(\mathbf{A}, \mathbf{T}, \mathbf{Z}_1, \mathbf{Z}_2, c_1)$

$\mathbf{C}_2 := \mathsf{H}(\mathbf{Z}_1, c_1)$
$\mathbf{W}_1 := \mathbf{A} \cdot \mathbf{Z}_1 - \mathsf{Rot}_d(c_1) \cdot \mathbf{T}$
$\mathbf{W}_2 := \mathbf{A} \cdot \mathbf{Z}_2 - \mathbf{T} \cdot \mathbf{C}_2$
Accept iff $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2) = 1$ and $c_1 = \mathsf{H}'(\mathbf{W}_1, \mathbf{W}_2, \mathbf{T})$

A.2 5-Round Protocol

$\mathcal{P}_{\mathsf{3Rounds}}$ is a Σ-protocol that uses a cryptographic hash function H which needs to be modeled as a random oracle. Thus it can also be seen as a transformation of a 5-round protocol into a 3-round one. In this section, we provide the 5-round version $\mathcal{P}_{\mathsf{5Rounds}}$ (in Figure 2) which does not require a random oracle. In addition, we make the necessary (but standard) modifications in order to reduce the communication complexity and also prove that the protocol is zero-knowledge even if a partial transcript that ends in an abort is output. We point out that these latter two changes can just as easily be applied to the 3-round protocol $\mathcal{P}_{\mathsf{3Rounds}}$.

The purpose of using the cryptographic hash function H in $\mathcal{P}_{\mathsf{3Rounds}}$ was to force the prover to obtain the challenge $\mathbf{C}_2$ after choosing his response $\mathbf{Z}_1$. Thus one can convert the 3-round protocol into a 5-round one by making the prover send $\mathbf{Z}_1$ as a response and then receive $\mathbf{C}_2$ from the verifier. It's easy to see that the proof for the zero-knowledge property of the non-aborting transcripts from Section 3.2 carries over without any changes. To make the proof zero-knowledge for all transcripts, the prover needs to not reveal the values $\mathbf{W}_1, \mathbf{W}_2$ in case of abort. He can therefore send commitments to these values and only open them in case the transcript is non-aborting. The simulation for aborting transcripts would then simply send commitments of 0. Furthermore, since the probability that an aborting transcript is sent is independent of the secret $\mathbf{S}$, the simulator knows exactly with what probability to send an aborting transcript.

Protocol $\mathcal{P}_{\mathsf{5Rounds}}$

$\mathsf{Com}(\mu, r)$ is the commitment function of a commitment scheme whose parameters are provided as a CRS
Prover's Information: $\mathbf{S} \in \mathbb{Z}^{nm \times k}$
Public Instance-Specific Information: $\mathbf{A} \in \mathbb{Z}_p^{dn \times nm}, \mathbf{T} := \mathbf{A} \cdot \mathbf{S} \in \mathbb{Z}_p^{dn \times k}$

Prover		Verifier
$\mathbf{Y}_1 \leftarrow D_y^{nm \times k}$, $\mathbf{W}_1 := \mathbf{A} \cdot \mathbf{Y}_1$		
$\mathbf{Y}_2 \leftarrow D_y^{nm \times \ell}$, $\mathbf{W}_2 := \mathbf{A} \cdot \mathbf{Y}_2$		
$r_1 \leftarrow \{0,1\}^{256}$, $\mathsf{com}_1 := \mathsf{Com}(\mathbf{W}_1, r_1)$		
$r_2 \leftarrow \{0,1\}^{256}$, $\mathsf{com}_2 := \mathsf{Com}(\mathbf{W}_2, r_2)$		
	$\xrightarrow{\mathsf{com}_1, \mathsf{com}_2}$	
		$c_1 \leftarrow \mathcal{C}$
	$\xleftarrow{c_1}$	
$\mathbf{Z}_1 := \mathsf{Rot}_m(c_1) \cdot \mathbf{S} + \mathbf{Y}_1$		
$u_1 \leftarrow \mathsf{RejectionSample}(\mathbf{Z}_1, \mathsf{Rot}_m(c_1) \cdot \mathbf{S}, \sigma_1, \rho_1)$		
if $u_1 = 0$ then abort		
	$\xrightarrow{r_1, \mathbf{Z}_1}$	
		$\mathbf{C}_2 \leftarrow \{0,1\}^{k \times \ell}$
	$\xleftarrow{\mathbf{C}_2}$	
$\mathbf{Z}_2 := \mathbf{S} \cdot \mathbf{C}_2 + \mathbf{Y}_2$		
$u_2 \leftarrow \mathsf{RejectionSample}(\mathbf{Z}_2, \mathbf{S} \cdot \mathbf{C}_2, \sigma_2, \rho_2)$		
$u_3 \leftarrow \mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2)$		
if $u_2 = 0 \vee u_3 = 0$ then abort		
	$\xrightarrow{r_2, \mathbf{Z}_2}$	
		$\mathbf{W}_1 := \mathbf{A} \cdot \mathbf{Z}_1 - \mathsf{Rot}_d(c_1) \cdot \mathbf{T}$
		$\mathbf{W}_2 := \mathbf{A} \cdot \mathbf{Z}_2 - \mathbf{T} \cdot \mathbf{C}_2$
		Accept iff $\mathsf{IsSmall}(\mathbf{Z}_1, \mathbf{Z}_2) = 1$
		and $\mathsf{Com}(\mathbf{W}_1, r_1) = \mathsf{com}_1$
		and $\mathsf{Com}(\mathbf{W}_2, r_2) = \mathsf{com}_2$

Fig. 2. 5-Round HVZK Proof without Random Oracles.

A.3 Amplification of Completeness in the Interactive Protocol

Since our interactive protocols use rejection sampling, they do not have perfect completeness – so the basic protocol may need to be repeated several times. When the protocol is made non-interactive

via the Fiat-Shamir transform, it clearly attains perfect completeness with the completeness error of the interactive scheme only affecting the expected running time – that is if the prover in the interactive protocol succeeds with probability ϵ, the expected number of iterations of the non-interactive protocol will be $1/\epsilon$.

One can also use standard techniques to decrease the completeness error of the interactive protocol – the idea is the same as for constructing Merkle signatures using a hash tree, and we only provide a sketch here. The prover creates α commitments (rather than just 1) in the first round of the protocol, creates a hash tree from the commitments, and sends the root of the hash tree. The verifier then sends an ordered list of challenges and the prover attempts to create a response to challenge i using the secret information in commitment i. The first challenge to which he can successfully respond becomes the response, and the prover opens the corresponding commitment. In addition to sending the response and opening the commitment, the prover must prove that this commitment was part of the hash tree. For this, he sends the $\log \alpha$ adjacent nodes along the path from the commitment at the leaf to the root of the tree. If the completeness is ϵ, then this technique increases the completeness to $1-(1-\epsilon)^{\alpha}$. The size of the proof only increases by $\log \alpha$ hash values, and the running time of the first round of the protocol increases by a factor of α.

To prove zero-knowledge, we consider the simulator that creates a non-aborting transcript and places it at position i, where i is chosen from the same distribution as in the real proof (since the probability of an abort is public information, this distribution is geometric). The simulator then fills the other leafs with random values, honestly generates the hash tree, chooses all the other random challenges, and honestly opens the hash tree in the response stage.