Implementation and performance of a RLWE-based commitment scheme and ZKPoK for its linear and multiplicative relations

Ramiro Martínez [*1], Paz Morillo [1], and Sergi Rovira [2]

[1]Department of Mathematics, Universitat Politècnica de Catalunya, Jordi Girona 31, 08034, Barcelona, Catalonia, Spain
[2]WiSeCom-DTIC, Universitat Pompeu Fabra, Tànger, 122-140, 08018, Barcelona, Catalonia, Spain

July 2, 2023

Abstract

In this paper we provide the implementation details and performance analysis of the lattice-based post-quantum commitment scheme introduced by Martínez and Morillo in their work titled *RLWE-Based Zero-Knowledge Proofs for Linear and Multiplicative Relations* together with the corresponding Zero-Knowledge Proofs of Knowledge (ZKPoK) of valid openings, linear and multiplicative relations among committed elements. We bridge the gap between the existing theoretical proposals and practical applications, thoroughly revisiting the security proofs of the aforementioned paper to obtain tight conditions that allow us to find the best sets of parameters for actual instantiations of the commitment scheme and its companion ZKPoK. Our implementation is very flexible and its parameters can be adjusted to obtain a trade-off between speed and memory usage, analyzing how suitable for practical use are the underlying lattice-based techniques. Moreover, our implementation further extends the literature of exact Zero-Knowledge proofs, providing ZKPoK of committed elements without any soundness slack.

Keywords— Lattice-based Cryptography, Post-Quantum Cryptography, Implementation, Commitment Scheme, Zero-Knowledge Proofs of Knowledge.

1 Introduction

Lattice-based cryptography is currently a prolific research topic as it allows building many cryptographic primitives from post-quantum assumptions, that

*Corresponding Author: ramiro.martinez@upc.edu

is, basing its security on computational problems believed to be hard even for quantum computers. This became a significant concern since the publication of Shor's quantum algorithm [34], that efficiently solves both factorization and discrete logarithm problems.

In the future, a powerful enough quantum computer could be used to break the security of widely used cryptographic protocols, hence post-quantum alternatives have to be developed. Furthermore, any malicious agent could be storing current communications in order to access them whenever quantum computers are available. For this reason, quantum-safe cryptographic constructions are not only a future issue but a current desirable property to provide that long-term privacy.

Consequently, it is fundamental to study the practicality of lattice-based proposals. In this work we provide an efficient implementation[1] of Martínez and Morillo commitment scheme, presented in [32], together with the corresponding Zero-Knowledge Proofs of Knowledge (ZKPoK) of valid openings and linear and multiplicative relations among committed elements. Such a commitment scheme with these Proofs of Knowledge is a very flexible building block for more advanced cryptographic constructions, since it allows the realization of any arithmetic circuit.

The aim of this work is to bridge the gap between the existing theoretical proposals and practical applications, thoroughly revisiting the security proofs from [32] to obtain tight conditions that allow us to find the best sets of parameters for actual instantiations of the commitment scheme and its companion ZKPoK. The analysis of this implementation illustrates the usefulness of this construction and allows comparing it with alternative proposals.

1.1 Related Work

Multiple cryptographic constructions make use of ZKPoK derived from the seminal code-based identification scheme designed by Stern in [37]. It provides the most efficient ZKPoK for code-based cryptography [35] but has also been adapted to lattices. Examples of commitment schemes derived from Stern's protocol are [20] by Jain *et al.* that applies them to the Learning Parity with Noise problem and [39] by Xie *et al.* that adapts it to the lattice setting. The Stern-based techniques in [32] that we implement in this paper are an improvement of those of [39], while the notation is the same as yet another lattice-based commitment scheme presented by Benhamouda *et al.* in [10]. The latter uses a different approach called rejection sampling. This alternative has been further explored by Baum *et al.* in [9].

On the one hand, several improvements have recently been made on rejection sampling techniques for proving relations among committed elements. While [9] already had an efficient proof for linear relations, the recent work of Attema *et al.* [7] provides new proofs for multiplicative relations for this commitment

[1] We give the implementation as open-source in https://github.com/rammmiro/RLWE-commitment

scheme. It has also been modified to provide more efficient opening proofs in [38].

On the other hand, Stern-based techniques have been applied to prove knowledge of several relations (including multiplications) when committing (using lattices) to exponentially large integers using a polynomially large modulus q, as it was done by Libert *et al.* in [26]. This has been outperformed by Kuchta *et al.* in [24] and also in [29] by Lyubashevsky *et al.*, using the rejection sampling scenario. A general technique useful to reduce the soundness error of Stern-based protocols was presented in [11], greatly reducing the proof sizes at the expanse of significantly increasing the computational cost.

It is important to notice that some of these ZKPoK for committed elements relax the notion of *soundness* (with rejection sampling techniques the set of valid openings is strictly larger than the set of honestly generated commitments). For example, Bootle *et al.* [12] present some succinct Zero-Knowledge arguments with large soundness slack. A prover knowing a certain bound on the error term can only convince a verifier that they are able to extract an error which is exponentially larger. For that matter, the particular parameter that appears in the exponent has to be fixed, and the others adapted accordingly to satisfy the rest of the properties.

Some proposals as [13] or [19] try to circumvent these issues, verifying additional conditions. A very prolific research path is the use of a relaxed commitment scheme with companion rejection sampling ZKPoK, usually (variants of) [9], as a subroutine of a more complex protocol that then obtains efficient exact proofs. That is the case of [40, 18, 30, 28]. Nevertheless, our implementation increases the literature of exact Zero-Knowledge proofs, providing ZKPoK of committed elements without any soundness slack. This allows us to tightly choose all parameters as we do not have to take into account any error growth.

1.2 Notation

We follow the usual notation, denoting vectors with boldface roman letters as $\mathbf{v}$. If $\mathcal{A}$ is an algorithm we use $a \leftarrow \mathcal{A}$ to denote that a is the output of $\mathcal{A}$, if X is a set we use $x \leftarrow X$ to denote that x is sampled uniformly at random from X and if D is a probability distribution we use $d \leftarrow D$ to denote that d is sampled following the distribution D.

We denote by $\mathfrak{R}$ any poly-time verifiable polynomial binary relation, i.e. if $(x, w) \in \mathfrak{R}$ then the size of the *witness* w is polynomially bounded by the size of the *statement* x, that is, there exists a polynomial p such that $|w| \leq p(|x|)$. We abuse notation and write $x \notin \mathfrak{R}$ (and say x is *false*) if there is no w such that $(x, w) \in \mathfrak{R}$, or $x \in \mathfrak{R}$ (and say x is *true*) otherwise.

We denote the security parameter by λ in all security proofs.

1.3 Paper organization

In Section 2 we give the necessary cryptographic background for the rest of the paper, including commitment schemes, Zero-Knowledge proofs and hardness

assumptions. In Section 3 we detail the changes made to the original protocols of [32] and provide the pseudocode for all the protocols. In Section 4 we give the implementation details and our choices for the computation tasks of lattice membership, discrete Gaussian sampling, uniform sampling and multiplication in a truncated polynomial ring. In Section 5 we offer the experimental results and explain how to choose parameters for our protocols. We also provide a discussion on some interesting trade-offs. Finally, in 6, we provide conclusions and give some interesting future work.

2 Cryptographic background

In this section we briefly and informally describe *commitment schemes*, *Zero-Knowledge Proofs of Knowledge* and the *Ring Learning with Errors problem* (RLWE). We assume that the reader has (basic) prior knowledge of standard concepts of public-key cryptography. For additional formal definitions, we refer the reader to [32].

2.1 Commitment schemes

A *commitment scheme* is a two-party protocol with two phases that allows one party (the *sender*) to commit to a value that will only be revealed to another party (the *receiver*) showing an *opening* in the future. The following two requirements should be satisfied:

- Hiding: The receiver should not be able to gain any meaningful knowledge of the sender's value without the opening.

- Binding: It should be infeasible for the sender to output a commitment that can be opened to two different values.

These requirements should be satisfied even if the parties' behavior is malicious. For example, a straightforward secure commitment scheme can be built from a random oracle $\mathcal{O}$. The sender chooses uniformly at random a sufficiently large string $r \in \{0,1\}^\lambda$, where λ would be the security parameter, and computes $c \leftarrow \mathcal{O}(m\|r)$ where m is the message to be committed and $\|$ indicates concatenation of strings. To open it they just have to reveal m and r. In fact, this is the underlying auxiliary commitment scheme used in the protocols that will be briefly introduced in Section 3.

2.2 Zero-Knowledge Proofs of Knowledge

A Non-Interactive Zero-Knowledge Proof of Knowledge (NIZKPoK) allows a prover $\mathcal{P}$ to convince a verifier $\mathcal{V}$ that a certain statement x is true because they know some secret information w (a witness) that satisfies a given relation $(x, w) \in \mathfrak{R}$ while preserving confidentiality of such secret. In our case, the statement would be a commitment key and a commitment and the witness would

be a valid opening (or three commitments and three valid openings satisfying linear or multiplicative relations). More formally, $\mathcal{P}$ takes x and w as input and produces a proof Π and $\mathcal{V}$ accepts or rejects the proof by looking at x and Π. Every NIZKPoK system must satisfy the following properties:

- Completeness: If an honest prover knows the witness, then an honest verifier will always accept the proof provided that the protocol was followed.

- (Knowledge) Soundness: A protocol is said to be *sound* if no computationally bounded (for example with a limited number of oracle queries) dishonest adversary can produce a valid proof for a false statement ($x \notin \mathfrak{R}$), except with negligible probability.

  A protocol is said to have the stronger property of *Knowledge-Soundness* if there is an extractor such that provided $\mathcal{A}$ is an adversary that produces valid proofs for a statement x then the extractor is able to use it to obtain a valid witness such that $(x, w) \in \mathfrak{R}$ with a similar success probability.

- Zero-Knowledge: The proof itself should not reveal any additional information about the witness besides the fact that it exists.

2.3 Ring Learning with Errors

Let $f(x) = x^n + 1$ where n is a power of 2. Let $q \geq 3$ be an odd integer. Let $R = \mathbb{Z}[x]/\langle x^n + 1\rangle$ and $R_q = R/qR$. For the sake of being able to define norms of polynomials and vectors of polynomials, we choose $\{-\lfloor q/2 \rfloor, \ldots, \lfloor q/2 \rfloor\}$ as representatives for $\mathbb{Z}_q$. Let χ be a distribution over R. The $\mathrm{RLWE}_{n,q,\chi}$ problem is to distinguish the following two distributions over $\{(a_i, b_i)\}_i \subset R_q^2$ defined as follows: In the first distribution one samples (a_i, b_i) uniformly at random from R_q^2 and, in the second distribution, one secretly samples $s \leftarrow R_q$ uniformly and then defines $(a_i, b_i = a_i \cdot s + e_i) \in R_q^2$ by sampling $a_i \leftarrow R_q$ uniformly and $e_i \leftarrow \chi$. The $\mathrm{RLWE}_{n,q,\chi}$ assumption is that the $\mathrm{RLWE}_{n,q,\chi}$ problem is hard.

2.4 CSPRNG and XOF

A *cryptographically secure pseudorandom number generator* (CSPRNG) is a PRNG with the following additional properties:

- Given n consecutive bits of a random sequence obtained from the CSPRNG, there is no polynomial-time algorithm that can predict correctly the $(n+1)$th with non-negligible probability better than 0.5.

- If part or all the state is revealed, it should be impossible to reconstruct the stream of randomness produced before the leak.

In our implementation, we use OpenSSL to sample from a CSPRNG with at least 128 bits security level.

An *eXtendable Output Function* (XOF) is a generalization of a hash function. While a hash function produces an output of fixed length, an XOF can produce a digest of arbitrary length. Moreover, the extension of the length of the output (requesting more digest bits) only costs the generation of these extra bits. In our implementation, we use an XOF (SHAKE-128) to obtain an appropriate length buffer to store the cryptographically secure pseudorandom bytes used to sample an integer uniformly at random from a given interval.

3 The commitment and the ZKPoK

In this section, we describe the implementation approach and the modifications from the original protocols of [32]. For that matter, we first fix notation and provide a summarized description of the protocols that we have implemented in Section 3.1. In Section 3.2, we detail the modifications made to the ZKPoK protocols. Moreover, in Section 3.3 we establish all the conditions that guarantee each of the security notions (binding, hiding, soundness, zero-knowledge and correctness). Finally, in Section 3.4 we have included the full description of each of the protocols taking into account the tools that we have used to instantiate them.

Throughout the rest of the paper we will work over the quotient ring of polynomials $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ where n is a power of two and q is an odd prime.

As error distribution χ we are going to use $D^n_{\sigma,B}$ the truncated discrete Gaussian distribution over R. More specifically, when sampling a polynomial $p \leftarrow D^n_{\sigma,B}$ we independently sample each coefficient from a discrete Gaussian distribution over $\mathbb{Z}$ of parameter σ, D_σ, conditioned to be in the interval $[-B, B)$, where the bound B is another power of two. We sometimes abuse notation writing $||p||_\infty \leq B$ while what we are going to check is whether all the coefficients from p belong to the interval $[-B, B)$. The *discrete Gaussian distribution* of parameter σ over the integers assigns to $x \in \mathbb{Z}$ a probability which is proportional to

$$\rho_\sigma(x) = \exp\left(-\frac{1}{2}x^2/\sigma^2\right).$$

3.1 The Commitment Scheme and the companion ZKPoK

Our commitment scheme consists of the following three algorithms.

- **Key Generation**: This procedure takes as input lattice dimension n, modulus q, the number d of factors x^n+1 splits in when considered modulo q and vector dimension k and outputs a public key $pk = (\mathbf{a}, \mathbf{b}) \in \left(R_q^k\right)^2$.

- **Commitment**: On input a message $m \in R_q$ and a public key $(\mathbf{a}, \mathbf{b}) \in \left(R_q^k\right)^2$ it outputs a commitment $\mathbf{c} = \mathbf{a}m + \mathbf{b}r + \mathbf{e} \in R_q^k$ and an opening $o = (m, r, \mathbf{e})$ where $r \leftarrow R_q$ and $\mathbf{e} \leftarrow \left(D^n_{\sigma,B}\right)^k$.

- **Verification**: On input a a public key $(\mathbf{a}, \mathbf{b})$, a commitment $\mathbf{c}$ and an opening $o = (m, r, \mathbf{e})$ it accepts if $\mathbf{c} = \mathbf{a}m + \mathbf{b}r + \mathbf{e}$ and $||\mathbf{e}||_\infty \leq B$ and rejects otherwise.

Regarding the parameters, this scheme is secure under the assumption that the $\text{RLWE}_{n,q,D^n_{\sigma,B}}$ problem is hard if we choose n, q, σ and the number of samples k satisfying the properties that are going to be described in Section 3.3.

The companion opening, linear and multiplicative ZKPoK allow a prover to convince a verifier that they know a valid opening of a given commitment, three valid openings such that $m_3 = \lambda_1 m_1 + \lambda_2 m_2$ for given $\lambda_1, \lambda_2 \in R_q$ or three valid openings such that $m_3 = m_1 \cdot m_2$ respectively, without revealing any additional information about the specific messages and openings.

The main idea is to prove that coefficients of $\mathbf{e}$ are small by proving they have a binary decomposition with the desired limited length. Notice that our commitment scheme is not homomorphic, that is, the sum of two commitments does not always form a valid commitment (the norm of the added noise terms might exceed the bound). For that reason we cannot just add the two original commitments, and we would need to create a new commitment to the sum and prove knowledge of the linear relation (in a protocol that essentially runs the opening protocol three times in parallel disclosing the linear relation). To prove the product relation we again have to prove knowledge of the three openings in parallel, but this time taking care of the crossed terms that would appear when multiplying the secret messages while masked.

3.2 Protocol modifications

In this subsection, we describe what modifications we have introduced in the ZKPoK protocols defined in [32] in order to allow better security reductions and performance improvements. The implementation of the commitment scheme does not have any important difference from the original description of [32] and therefore it is omitted.

In order to choose secure sets of parameters that allow us to build an efficient implementation we first redo the security proofs from [32], as, in the same manner than [10], they were sometimes far from tight, and some relations were only asymptotically defined. To do so we have removed some constants, introduced new ones and replaced some conditions with others that play the same role but allow a tighter analysis.

It is important to emphasize that this is one of the main challenges when instantiating real world cryptographic protocols from a theoretic paper. Proving security from a theoretical point of view requires asymptotical proofs, as its goal is ensuring that undesired events occur with as low probability as we desire provided some parameters are large enough. These proofs guarantee the existence of parameters for any security level (under certain hypothesis), but do not inform of the particular parameters given that they might not explicitly describe how large these parameters should be, and have to be generic enough to work with any set of parameters.

Claiming a specific security level is something different, as the constants hidden in the asymptotic relations play a really important role, and we are able to tailor some inequalities to the specific set of parameters we would like to work with. These different proofs might lead to a different set of constraints for the parameters, that we are going to enumerate in the following subsection.

Both [32, 10] use $\sigma \in \mathcal{O}\left(n^{3/4}\right)$ as error parameter, but this is not useful for instantiating the commitment. On the one hand, fixing a particular relation $\sigma = c \cdot n^{3/4}$ might require absurdly large n for the properties to hold for some c, or, on the other hand, it might produce far from tight inequalities. Explicitly computing how large n has to be, or how tight are the proofs, is not straightforward in general.

Asymptotic relations are defined for functions, but, in order to study the best σ for a given n, a different analysis is required. It is also relevant to notice that the original proofs used as an assumption the hardness of the underlying RLWE problem, but did not discuss how the relations among the parameters affect this hardness. This again has to be specifically consider when instantiating the commitment.

Remove intermediate constant γ. Consequently, we no longer make use of the intermediate constant γ controlling the relation between q and n (inherited from [10], and useful to compare both schemes from a theoretical point of view, but not for choosing parameters) as we now explicitly check the relations between q and n for each security property.

Decoupling of noise bound from n introducing new bound B. Both [10] and [32] bounded the error term by n, so that σ could be asymptotically defined with respect to n and security proofs could be easily done. As we have mentioned, choosing specific parameters requires computing actual probabilities, and not its asymptotic behavior. Doing so we found out that it is possible to choose much tighter bounds on the error terms preserving all the other properties. For that matter, we have introduced a new parameter B, intended to be a much tighter upper bound to the norm of the error term.

Alternative constraint of the modulus q. Besides redoing the proofs for security considerations, we also change another condition, regarding the modulus q, to allow further optimizations.

To improve the efficiency of polynomial multiplications, we use the partial FFT approach described in [33] or [31]. The main idea of FFT multiplication algorithms (the specifics will be briefly described in section 4.4) is to transform the polynomials into a different domain where products are computationally cheaper, and then antitransform back the result.

This new approach requires $q \equiv 2d + 1 \mod 4d$ so that $x^n + 1$ factorizes in d irreducible polynomials in $\mathbb{Z}_q[x]$. Notice that when $d = 2$, which is the most efficient case as we will discuss, this translates into $q \equiv 5 \mod 8$ while the original proposals [10, 32] required $q \equiv 3 \mod 8$ so that $x^n + 1$ factorizes as well into two irreducible polynomials in $\mathbb{Z}_q[x]$.

Transform the Interactive protocols into Non-Interactive ones. We have also transformed, via Fiat-Shamir, the original Interactive-ZKPoK into NIZKPoK, computing the challenges from a cryptographically secure pseudo-

random number generator seeded with the previous elements from the conversation. To make the non-interactive version secure the number of repetitions of the protocols, denoted by δ_{OL} in the opening and linear proofs and by δ_M in the multiplicative one, have to be increased.

Redefine the input of the key generation algorithm. While the original version of the key generation algorithm Gen from [32] takes as input just the security parameter written in unary 1^λ we have split this process. First, we describe in Section 5.1 how to obtain a valid set of parameters (and we prove in Appendix C that this procedure produces indeed a set of parameters that guarantees certain security properties). In the GitHub repository we provide a `sagemath` script that takes as input (λ, n, q, d) and outputs the optimal $(\lambda, n, q, d, k, \sigma, B, \delta_{OL}, \delta_M)$. Then the proper key generation algorithm takes these parameters as input and outputs the key pair $(\mathbf{a}, \mathbf{b})$ and the auxiliary elements.

Increase the length of the openings for the auxiliary commitments so they take up an integer number of bytes. To instantiate the auxiliary commitment scheme we have chosen to use SHA3-256 as a hash function, computing $\mathsf{aCom}(m, o) = \text{SHA3-256}(m \| o)$ with $o \leftarrow \{0,1\}^{\lambda'}$, where $\lambda' \coloneqq 8 \cdot \lceil \lambda/8 \rceil$ for convenience, so that the opening has an integer number of bytes.

Reduce the proof size generating multiple seeds from a master seed. While in the original proposal the prover was supposed to send a different seed for each of the permutations, we choose to define just one master seed for every iteration from which the others are derived using SHAKE-128 as XOF.

3.3 Security conditions

In this subsection we establish all the conditions, some inherited from [32] and some improved ones, that guarantee each of the security notions.

For a detailed proof discussing why these specific conditions guarantee security, the reader is encouraged to read Appendices A and B.

Binding:

The binding property is guaranteed similarly than in [32] and [10], using a counting argument to ensure that no two valid openings exist for a single commitment except with a very small probability in the sampling of the commitment key. The argument is generalized to the fact that $x^n + 1$ factorizes into d irreducible polynomials (instead of only considering 2) and bounding the undesired probability by a specific $2^{-\lambda}$ instead of proving that it is asymptotically negligible in n. The conditions are:

- $q \equiv 2d + 1 \pmod{4d}$ (1)
- $k \geq \frac{\lambda + 2n \log_2(q)}{n(\log_2(q)/d - \log_2(4B-1))}$ (2)
- $d < \frac{\log_2(q)}{\log_2(4B-1)}$. (3)

Hiding:

To ensure the commitment is hiding we just have to check that the underlying computational problem is hard enough (we denote by $\mathsf{bitsec}(RLWE)$ the number of security bits estimated by Albrecht *et al.* Lattice Estimator[2] [1]) and that the noise distribution is bounded with sufficient probability (as we are going to truncate it). The inequalities are:

- $\mathsf{bitsec}(RLWE) \geq \lambda$ (4)
- $\Pr\left[||\mathbf{e}||_\infty \leq B \;\middle|\; \mathbf{e} \leftarrow D_{\sigma_e}^{kn} \right] \geq 1 - 2^{-\lambda}.$ (5')

Regarding the second condition, we recall that we abuse notation writing $||\mathbf{e}||_\infty \leq B$ as what we really assume is that every coefficient e of each of the polynomials in the vector of polynomials $\mathbf{e}$ satisfies $-B \leq e < B$.

The condition we need is (5'), but, provided that we do not know how to explicitly compute that probability, we are going to latter define a slightly stronger condition in eq. (5) that will imply (5'). This condition will be presented in Section 5.1 as it uses several definitions that are going to be introduced there.

Soundness:

Soundness is ensured by combining the knowledge-soundness of the interactive protocol with the security guarantees of the Fiat-Shamir transform. Soundness is immediate for the Fiat-Shamir transform of classical special sound Σ-protocols (with only 3 moves), but needs a more detailed analysis when the protocol has $2\mu + 1$ rounds and requires parallel repetitions.

The original proposal[32] already showed that the soundness error of the opening and the linear relation was $\kappa_{OL}^{\delta_{OL}}$ and $\kappa_M^{\delta_M}$ with $\kappa_{OL} = (q+1)/2q$ and $\kappa_M = (q^2 + 3q - 2)/2q^2$ respectively.

We take advantage of the more fine-graded special-soundness definitions for multiple round protocols and the proofs from [4] to show that $\kappa_{OL}^{\delta_{OL}}$ and $\kappa_M^{\delta_M}$ are also the knowledge error of their respective protocols with the standard definitions. For that matter, the first necessary security conditions are inherited from [32]:

- $((q+1)/2q)^{\delta_{OL}} \leq 2^{-\lambda}$ (6)
- $\left((q^2 + 3q - 2)/2q^2\right)^{\delta_M} \leq 2^{-\lambda}.$ (7)

Then in order to be able to directly apply the known results about the Fiat-Shamir transformation of a multi-round protocol we formally prove in section A.3 that both soundness and knowledge-soundness are preserved in the ROM if the verifier just sends some seeds from which the actual challenges are

[2] Commit `f9f4b3c69d5be6df2c16243e8b1faa80703f020c` from `github.com/malb/lattice-estimator`

latter pseudorandomly derived, provided the seed space is large enough. We believe that formal proof is of independent interest, and it provides the next condition, where $\mathcal{S}$ is the seed space (in the Non-Interactive version is the output space of the extendable output function that is applied to the previous elements to get the challenge):

- $|\mathcal{S}| \geq 2^{\lambda} q^{\delta}$. (8)

Finally, we can use the known results about the security of the Fiat-Shamir transform, that ensure soundness and knowledge-soundness are preserved but with a security loss of $O(Q^4)$ or $O(Q^2)$ depending on if we consider it in the QROM or in the plain ROM respectively, where Q is the number of oracle queries made by the adversary.

To get provable security, one should increase δ_{OL} and δ_M up to the order of 3λ or 5λ so that $\kappa^{\delta} \approx 2^{-3\lambda}$ or $\kappa^{\delta} \approx 2^{-5\lambda}$ to compensate for the respective ROM or QROM security loss.

While these security losses are tight in general, it is usually assumed that for practical protocols it is milder (and sometimes it is even completely disregarded). We follow an intermediate approach and study how the currently known attacks could be applied to the considered protocols in appendix B. We provide upper bounds to the success probability of the best attacks (to the best of our knowledge, the family of attacks presented in [5], the full version of [6]) and get conditions on the parameters that ensure these probabilities are below $2^{-\lambda}$. From that, we get the final soundness condition that should apply to both $\delta = \delta_{OL}$ and $\delta = \delta_M$, which will be satisfied if the number of repetitions δ is large enough:

$$\begin{aligned} (2\lambda - 1) \log_2 \left(\frac{2\lambda - 1}{\delta(1 - q^{-1})} \right) + \\ +(\delta - 2\lambda + 1) \log_2 \left(\frac{\delta - 2\lambda + 1}{\delta q^{-1}} \right) \geq 2\lambda. \end{aligned} \tag{9}$$

Zero-Knowledge:

The interactive version of the protocol is Honest-Verifier Zero-Knowledge because there exists a simulator that taking the challenges as input outputs conversations that follow the same distribution as real conversations between an honest prover and an honest verifier.

Provided that the conversation itself is then distributed as something that can be computed by an algorithm that does not know the witness, we can ensure no relevant information is leaked, because anything that can be computed from the conversation could also be efficiently computed using the simulator.

Finally, in the ROM, as we assume challenges in the Fiat-Shamir transformed version follow a uniform distribution, the non-interactive proof would be in that sense equivalent to the interactive conversation, and we can ensure again that no information is leaked. We use cryptographically secure pseudorandom number generators, so the Zero-Knowledge property is only computational.

Correctness:

It is unconditional, as we already truncate the error distributions so that every verification holds.

3.4 Pseudocode

Before detailing the protocols, we require extra notation and several auxiliary functions.

Let ϕ be the function that takes as input a vector of size $2nk$ from $\mathbb{Z}_q^{2nk}$ and outputs a vector of k polynomials in R_q^k that has as coefficients the first k blocks of n elements in the input vector[3].

Let $\mathbf{B}$ be the vector of k polynomials from R_q^k that has all its coefficients equal to the bound B.

Let Hash and XOF be a hash function and an extendable output function, respectively.

And finally, let π_τ be the function that takes as input a vector of integers and permutes its elements using a pseudorandom permutation derived from the seed $\tau \in \{0,1\}^{8\lceil\lambda/8\rceil}$ (the specific procedure is going to be defined in Section 4.3).

We will call $\tau_i \in \{0,1\}^{8\lceil\lambda/8\rceil}$ the master seed for the ith iteration and expand it into $\log_2(B)+1$ seeds $\tau_{ij} \in \{0,1\}^{8\lceil\lambda/8\rceil}$ using XOF.

A vector of polynomials $\mathbf{e} \in R_q^k$ with small coefficients, all of them in the interval $[-B, B)$, can be transformed into another vector of non-negative small coefficients by adding $\mathbf{B}$. Then we can decompose and extend it into $\log_2(B)+1$ binary vectors $\mathbf{e}'_j$ of length $2nk$ whose first half contains k blocks of n bits that represent the jth bits of the binary decomposition of the coefficients of each of the polynomials once $\mathbf{B}$ has been added and the second half contains one block of 0 followed by one block of 1 so that $\mathbf{e}'_j$ has the same number of each (then permuting vectors extended this way allows us to hide all information but the fact that the norm of the original polynomials is bounded by B, see [32]).

We denote by expand the function that takes as input a vector of polynomials $\mathbf{e}$ with all its coefficients in $[-B, B)$ and outputs $\log_2(B)+1$ vectors $\mathbf{e}'_j$ following the previously described procedure.

We denote by $\mathfrak{B}_n \subset \{0,1\}^{2n}$ the set of binary strings of length $2n$ with exactly n zeroes and n ones, so that correctly computed $\mathbf{e}'_j$ belong to $\mathfrak{B}_{nk}$.

We denote by PRN a function that takes as input a seed and outputs vectors of pseudorandom integers from $\mathbb{Z}_q$. Analogously, we denote by PRB a function that takes as input a seed and outputs δ pseudorandom bits.

[3]Notice that the function ϕ does not consider the second half of the input vector.

Protocol 1

Proving Knowledge of a Valid Opening

$\{\mathbf{e}'_j\}_j \leftarrow \mathsf{expand}(\mathbf{e})$
for $i \in 1, \ldots, \delta_{OL}$ **do**
  $\tau_i \leftarrow \{0,1\}^{8\lceil \lambda/8 \rceil}$
  $\{\tau_{ij}\}_j \leftarrow \mathsf{XOF}(\tau_i)$
  $\{\mathbf{f}_{ij}\}_j \leftarrow \mathbb{Z}_q^{2nk}$
  $\mu_i, \rho_i \leftarrow R_q$
  $o_{1i}, o_{2i} \leftarrow \{0,1\}^{8\lceil \lambda/8 \rceil}$
  $\mathbf{y}_i = \mathbf{a}\mu_i + \mathbf{b}\rho_i + \phi(\sum_j 2^j \mathbf{f}_{ij})$
  $c_{1i} \leftarrow \mathsf{Hash}(\tau_i \| \mathbf{y}_i \| o_{1i})$
  $c_{2i} \leftarrow \mathsf{Hash}(\{\pi_{\tau_{ij}}(\mathbf{f}_{ij})\}_j \| \{\pi_{\tau_{ij}}(\mathbf{e}'_j)\}_j \| o_{2i})$

$\mathrm{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\mathbf{c}\|\{c_{1i}, c_{2i}\}_i)$
$\{\alpha_i\}_i \leftarrow \mathsf{PRN}(\mathrm{seed}_1)$

for $i \in 1, \ldots, \delta_{OL}$ **do**
  for $j \in 0, \ldots, \log_2(B)$ **do**
    $\mathbf{g}_{ij} = \pi_{\tau_{ij}}(\mathbf{f}_{ij} + \alpha_i \mathbf{e}'_j)$

$\mathrm{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\mathbf{c}\|\mathrm{seed}_1\|\{\mathbf{g}_{ij}\}_{ij})$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\mathrm{seed}_2)$

for $i \in 1, \ldots, \delta_{OL}$ **do**
  if $b_i = 0$ **then**
    $s_i = \rho_i + \alpha_i r$
  else if $b_i = 1$ **then**
    for $j \in 0, \ldots, \log_2(B)$ **do**
      $\widetilde{\mathbf{e}}'_{ij} = \pi_{\tau_{ij}}(\mathbf{e}'_j)$

return $\{c_{i1}, c_{i2}\}_i, \{\mathbf{g}_{ij}\}_{ij}, \{(\tau_i, \mathbf{y}_i, s_i, o_{1i})\}_{i \text{ s.t. } b_i=0}, \{(\{\widetilde{\mathbf{e}}'_{ij}\}_j, o_{2i})\}_{i \text{ s.t. } b_i=1}$

Protocol 2

Verifying Knowledge of a Valid Opening

$\mathrm{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\mathbf{c}\|\{c_{1i}, c_{2i}\}_i)$
$\mathrm{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\mathbf{c}\|\mathrm{seed}_1\|\{\mathbf{g}_{ij}\}_{ij})$
$\{\alpha_i\}_i \leftarrow \mathsf{PRN}(\mathrm{seed}_1)$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\mathrm{seed}_2)$
for $i \in 1, \ldots, \delta_{OL}$ **do**
  if $b_i = 0$ **then**
    $\{\tau_{ij}\}_j \leftarrow \mathsf{XOF}(\tau_i)$
    $c_{1i} \stackrel{?}{=} \mathsf{Hash}(\tau_i \| \mathbf{y}_i \| o_{1i})$
    $\mathbf{z}_i \coloneqq \mathbf{y}_i + \alpha_i(\mathbf{c} + \mathbf{B}) - \mathbf{b}s_i - \phi(\sum_j 2^j \pi^{-1}_{\tau_{ij}}(\mathbf{g}_{ij}))$
    $\mathbf{z}_i \stackrel{?}{\in} \mathcal{L}(\mathbf{a})$
  else if $b_i = 1$ **then**
    $c_{2i} \stackrel{?}{=} \mathsf{Hash}\left(\{\mathbf{g}_{ij} - \alpha_i \widetilde{\mathbf{e}}'_{ij}\}_j \| \{\widetilde{\mathbf{e}}'_{ij}\}_j \| o_{2i}\right)$
    for $j \in 0, \ldots, \log_2(B)$ **do**
      $\widetilde{\mathbf{e}}'_{ij} \stackrel{?}{\in} \mathfrak{B}_{nk}$

Protocol 3

Proving Knowledge of a Linear Relation

for $h \in 1,2,3$ **do**
  $\{\mathbf{e}'_{hj}\}_j \leftarrow \mathsf{expand}(\mathbf{e}_h)$
for $i \in 1,\ldots,\delta_{OL}$ **do**
  for $h \in 1,2,3$ **do**
    $\tau_{hi} \leftarrow \{0,1\}^{8\lceil \lambda/8 \rceil}$
    $\{\tau_{hij}\}_j \leftarrow \mathsf{XOF}(\tau_{hi})$
    $\{\mathbf{f}_{hij}\}_j \leftarrow \mathbb{Z}_q^{2nk}$
  $\mu_{1i}, \mu_{2i}, \{\rho_{hi}\}_h \leftarrow R_q$
  $\mu_{3i} \coloneqq \lambda_1\mu_{1i} + \lambda_2\mu_{2i}$
  $o_{1i}, o_{2i} \leftarrow \{0,1\}^{8\lceil \lambda/8 \rceil}$
  for $h \in 1,2,3$ **do**
    $\mathbf{y}_{hi} = \mathbf{a}\mu_{hi} + \mathbf{b}\rho_{hi} + \phi(\sum_j 2^j \mathbf{f}_{hij})$
  $c_{1i} \leftarrow \mathsf{Hash}(\{\tau_{hi}\}_h \| \{\mathbf{y}_{hi}\}_h \| o_{1i})$
  $c_{2i} \leftarrow \mathsf{Hash}(\{\pi_{\tau_{hij}}(\mathbf{f}_{hij})\}_{hj} \| \{\pi_{\tau_{hij}}(\mathbf{e}'_{hj})\}_{hj} \| o_{2i})$
$\mathrm{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\lambda_1\|\lambda_2\|\{c_{1i}, c_{2i}\}_i)$
$\{\alpha_i\}_i \leftarrow \mathsf{PRN}(\mathrm{seed}_1)$
for $i \in 1,\ldots,\delta_{OL}$ **do**
  for $h \in 1,2,3$ **do**
    for $j \in 0,\ldots,\log_2(B)$ **do**
      $\mathbf{g}_{hij} = \pi_{\tau_{hij}}(\mathbf{f}_{hij} + \alpha_i \mathbf{e}'_{hj})$
$\mathrm{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\lambda_1\|\lambda_2\|\mathrm{seed}_1\|\{\mathbf{g}_{hij}\}_{hij})$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\mathrm{seed}_2)$
for $i \in 1,\ldots,\delta_{OL}$ **do**
  if $b_i = 0$ **then**
    for $h \in 1,2,3$ **do**
      $s_{hi} = \rho_{hi} + \alpha_i r_{hi}$
  else if $b_i = 1$ **then**
    for $h \in 1,2,3$ **do**
      for $j \in 0,\ldots,\log_2(B)$ **do**
        $\widetilde{\mathbf{e}}'_{hij} = \pi_{\tau_{hij}}(\mathbf{e}'_{hj})$
return $\{c_{1i}, c_{2i}\}_i$,$\{\mathbf{g}_{hij}\}_{hij}$,
  $\left\{\{\tau_{hi}\}_h, \{\mathbf{y}_{hi}\}_h, \{s_{hi}\}_h, o_{1i}\right\}_{i \text{ s.t. } b_i=0}$,
  $\left\{\{\widetilde{\mathbf{e}}'_{hij}\}_{hj}, o_{2i}\right\}_{i \text{ s.t. } b_i=1}$

Protocol 4

Proving Knowledge of a Multiplicative Relation

for $h \in 1,2,3$ **do**
  $\{\mathbf{e}'_{hj}\}_j \leftarrow \mathsf{expand}(\mathbf{e}_h)$
for $i \in 1,\ldots,\delta_M$ **do**
  for $h \in 1,2,3$ **do**
    $\tau_{hi} \leftarrow \{0,1\}^{8\lceil\lambda/8\rceil}$
    $\{\tau_{hij}\}_j \leftarrow \mathsf{XOF}(\tau_{hi})$
    $\{\mathbf{f}_{hij}\}_j \leftarrow \mathbb{Z}_q^{2nk}$
    $\mu_{hi}, \rho_{hi} \leftarrow R_q$
    $\mathbf{y}_{hi} = \mathbf{a}\mu_{hi} + \mathbf{b}\rho_{hi} + \phi(\sum_j 2^j \mathbf{f}_{hij})$
  $\mu_{\times i}, \mu_{+i} \leftarrow R_q$
  $m_{\times i} = \mu_{1i}\mu_{2i}, \quad m_{+i} = \mu_{1i}m_2 + \mu_{2i}m_1$
  $o_{1i}, o_{2i}, o_{3i}, o_{4i}, o_{5i} \leftarrow \{0,1\}^{8\lceil\lambda/8\rceil}$
  $c_{1i} \leftarrow \mathsf{Hash}(\{\tau_{hi}\}_h \| \{\mathbf{y}_{hi}\}_h \| o_{1i})$
  $c_{2i} \leftarrow \mathsf{Hash}(\mu_{3i} \| \mu_{\times i} \| \mu_{+i} \| o_{2i})$
  $c_{3i} \leftarrow \mathsf{Hash}(\{\pi_{\tau_{hij}}(\mathbf{f}_{hij})\}_{hj} \| \{\pi_{\tau_{hij}}(\mathbf{e}'_{hj})\}_{hj} \| o_{3i})$
  $c_{4i} \leftarrow \mathsf{Hash}(\mu_{\times i} + m_{\times i} \| \mu_{+i} + m_{+i} \| o_{4i})$

$\mathrm{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a} \| \mathbf{b} \| \{\mathbf{c}_h\}_h \| \{c_{1i}, c_{2i}, c_{3i}, c_{4i}\}_i)$
$\{\alpha_i, \beta_i\}_i \leftarrow \mathsf{PRN}(\mathrm{seed}_1)$

for $i \in 1,\ldots,\delta_M$ **do**
  $\gamma_{1i} \coloneqq \alpha_i, \quad \gamma_{2i} \coloneqq \alpha_i, \quad \gamma_{3i} \coloneqq \beta_i$
  $c_{i5} \leftarrow \mathsf{Hash}(\beta\mu_{\times i} + \alpha_i\beta_i\mu_{+i} + {\alpha_i}^2\mu_{3i} \| o_{i5})$
  for $h \in 1,2,3$ **do**
    for $j \in 0,\ldots,\log_2(B)$ **do**
      $\mathbf{g}_{hij} = \pi_{\tau_{hij}}(\mathbf{f}_{hij} + \gamma_{hi}\mathbf{e}'_{hj})$

$\mathrm{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a} \| \mathbf{b} \| \{\mathbf{c}_h\}_h \| \mathrm{seed}_1 \| \{c_{5i}\}_i \| \{\mathbf{g}_{hij}\}_{hij})$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\mathrm{seed}_2)$

for $i \in 1,\ldots,\delta_M$ **do**
  if $b_i = 0$ **then**
    $t_{\times i} = \mu_{\times i} + m_{\times i}, \quad t_{+i} = \mu_{+i} + m_{+i}$
    for $h \in 1,2,3$ **do**
      $s_{hi} = \rho_{hi} + \gamma_{hi} r_h$
  else if $b_i = 1$ **then**
    for $h \in 1,2,3$ **do**
      for $j \in 0,\ldots,\log_2(B)$ **do**
        $\widetilde{\mathbf{e}}'_{hij} = \pi_{\tau_{hij}}(\mathbf{e}'_{hj})$
return $\{c_{1i}, c_{2i}, c_{3i}, c_{4i}, c_{5i}\}_i, \{\mathbf{g}_{hij}\}_{hij},$
$\left\{(\{\tau_{hi}\}_h, \{\mathbf{y}_{hi}\}_h, t_{\times i}, t_{+i}, \{s_{hi}\}_h, o_{1i}, o_{4i}, o_{5i})\right\}_{\substack{i \text{ s.t.} \\ b_i=0}},$
$\left\{(\{\widetilde{\mathbf{e}}'_{hij}\}_{hj}, \mu_{3i}, \mu_{\times i}, \mu_{+i}, o_{2i}, o_{3i}, o_{5i})\right\}_{\substack{i \text{ s.t.} \\ b_i=1}}$

Protocol 5

Verifying Knowledge of a Linear Relation

$\text{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\lambda_1\|\lambda_2\|\{c_{1i}, c_{2i}\}_i)$
$\text{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\lambda_1\|\lambda_2\|\text{seed}_1\|\{\mathbf{g}_{hij}\}_{hij})$
$\{\alpha_i\}_i \leftarrow \mathsf{PRN}(\text{seed}_1)$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\text{seed}_2)$
for $i \in 1, \ldots, \delta_{OL}$ **do**
  if $b_i = 0$ **then**
    for $h \in 1, 2, 3$ **do**
      $\{\tau_{hij}\}_j \leftarrow \mathsf{XOF}(\tau_{hi})$
      $\mathbf{z}_{hi} \coloneqq \mathbf{y}_{hi} + \alpha_i(\mathbf{c}_h + \mathbf{B}) - \mathbf{b}s_{hi} - \phi(\sum_j 2^j \pi^{-1}_{\tau_{hij}}(\mathbf{g}_{hij}))$
      $\mathbf{z}_{hi} \stackrel{?}{\in} \mathcal{L}(\mathbf{a})$
    $c_{1i} \stackrel{?}{=} \mathsf{Hash}\left(\{\tau_{hi}\}_h\|\{\mathbf{y}_{hi}\}_h\|o_{1i}\right)$
    $\mathbf{z}_{3i} \stackrel{?}{=} \lambda_1\mathbf{z}_{1i} + \lambda_2\mathbf{z}_{2i}$
  else if $b_i = 1$ **then**
    $c_{2i} \stackrel{?}{=} \mathsf{Hash}(\{\mathbf{g}_{hij} - \alpha_i\widetilde{\mathbf{e}}'_{hij}\}_{hj}\|\{\widetilde{\mathbf{e}}'_{hij}\}_{hj}\|o_{2i})$
    for $h \in 1, 2, 3$ **do**
      for $j \in 0, \ldots, \log_2(B)$ **do**
        $\widetilde{\mathbf{e}}'_{hij} \stackrel{?}{\in} \mathfrak{B}_{nk}$

Protocol 6

Verifying Knowledge of a Multiplicative Relation

$\text{seed}_1 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\{c_{1i}, c_{2i}, c_{3i}, c_{4i}\}_i)$
$\text{seed}_2 \leftarrow \mathsf{XOF}(\mathbf{a}\|\mathbf{b}\|\{\mathbf{c}_h\}_h\|\text{seed}_1\|\{c_{5i}\}_i\|\{\mathbf{g}_{hij}\}_{hij})$
$\{\alpha_i, \beta_i\}_i \leftarrow \mathsf{PRN}\,(\text{seed}_1)$
$\{b_i\}_i \leftarrow \mathsf{PRB}(\text{seed}_2)$
for $i \in 1, \ldots, \delta_M$ **do**
  $\gamma_{1i} \coloneqq \alpha_i, \quad \gamma_{2i} \coloneqq \alpha_i, \quad \gamma_{3i} \coloneqq \beta_i$
  if $b_i = 0$ **then**
    for $h \in 1, 2, 3$ **do**
      $\mathbf{z}_{hi} \coloneqq \mathbf{y}_{hi} + \gamma_{hi}(\mathbf{c}_h + \mathbf{B}) - \mathbf{b}s_{hi} - \phi(\sum_j 2^j \pi^{-1}_{\tau_{hij}}(\mathbf{g}_{hij}))$
      $\mathbf{z}_{hi} \stackrel{?}{\in} \mathcal{L}(\mathbf{a})$
      Let $t_{hi} \in R_q$ s.t. $\mathbf{z}_{hi} = \mathbf{a}t_{hi}$
    $c_{1i} \stackrel{?}{=} \mathsf{Hash}(\{\tau_{hi}\}_h\|\{\mathbf{y}_{hi}\}_h\|o_{1i})$
    $c_{4i} \stackrel{?}{=} \mathsf{Hash}(t_{\times i}\|t_{+i}\|o_{4i})$
    $c_{5i} \stackrel{?}{=} \mathsf{Hash}(\beta_i t_{\times i} + \alpha_i\beta_i t_{+i} + {\alpha_i}^2 t_{3i} - \beta_i t_{1i} t_{2i}\|o_{5i})$
  else if $b_i = 1$ **then**
    $c_{2i} \stackrel{?}{=} \mathsf{Hash}(\mu_{3i}\|\mu_{\times i}\|\mu_{+i}\|o_{2i})$
    $c_{3i} \stackrel{?}{=} \mathsf{Hash}(\{\mathbf{g}_{hij} - \gamma_{hi}\widetilde{\mathbf{e}}'_{hij}\}_{hj}\|\{\widetilde{\mathbf{e}}'_{hij}\}_{hj}\|o_{3i})$
    $c_{5i} \stackrel{?}{=} \mathsf{Hash}(\beta_i\mu_{\times i} + \alpha_i\beta_i\mu_{+i} + {\alpha_i}^2\mu_{3i}\|o_{5i})$
    for $h \in 1, 2, 3$ **do**
      for $j \in 0, \ldots, \log_2(B)$ **do**
        $\widetilde{\mathbf{e}}'_{hij} \stackrel{?}{\in} \mathfrak{B}_{nk}$

Protocols 1, 3 and 4 and Protocols 2, 5 and 6 respectively show the tasks of a prover $\mathcal{P}$ and a verifier $\mathcal{V}$ to create and verify proofs of knowledge of a valid opening to commitment $\mathbf{c}$ with public key $(\mathbf{a}, \mathbf{b})$, a linear or a multiplicative relation (that is the committed messages m_1, m_2 and m_3 from commitments $\mathbf{c}_1$,$\mathbf{c}_2$ and $\mathbf{c}_3$ either satisfy $m_3 = \lambda_1 m_1 + \lambda_2 m_2$ for given $\lambda_1, \lambda_2 \in R_q$ or $m_3 = m_1 \cdot m_2$).

In order to get NIZKPoK from the original public coin interactive protocols, we run multiple parallel iterations to achieve a small soundness error and replace the verifier answers with pseudorandom responses that depend on the previous elements and the statement of the proof. To ease the comparison with the original interactive protocols, we highlight these responses in the pseudocode by writing them justified to the right.

4 Implementation

In this section we explain in detail the choices that we have made in our implementation.

All the algorithms are implemented in C. In order to account for as wide a range of applications as possible, we have chosen to work with arbitrary precision arithmetic. We use the GNU MP library (GMP v6.2.1) and FLINT (v2.9.0) to do so. Finally, we use the OpenSSL library (v3.0.2) to compute hash values and for the implementation of a cryptographically secure uniform sampler of integers in $\mathbb{Z}_q$.

One of the main drawbacks of Stern's approach is that the non-negligible soundness error requires multiple repetitions to obtain soundness. Nevertheless, we can take advantage of the parallel repetitions to parallelize the execution and speed up the protocols. We use OpenMP for that, and it has a great impact on the final performance.

The two main computational tasks that we require to implement our schemes are sampling (from a discrete Gaussian distribution, uniformly random integers in $\mathbb{Z}_q$ and uniformly random permutations) and multiplying in a truncated polynomial ring. It is also relevant to describe how to check if a vector of polynomials belongs to a given lattice. In the following subsections, we will give an overview of the approach that we have followed to instantiate such tasks in our implementation.

4.1 Lattice membership

The original description of the protocol presented in [32] did not detail how to check if a point $\mathbf{z}$ belongs to $\mathcal{L}(\mathbf{a})$, the ideal lattice defined by $\mathbf{a}$, neither how to recover its coordinates. In order to efficiently do so we use one property of the commitment key that was not mentioned either, but that can be deduced from the binding property of the commitment scheme.

Lemma 1. *If* $(\mathbf{a}, \mathbf{b}) \in \left(R_q^k\right)^2$ *defines a commitment key for a binding commitment scheme in a ring* $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ *such that* $x^n + 1$ *splits in* d

irreducible $p_j(x)$ modulo q for $j \in \{1, \ldots, d\}$ then there exist d indexes i_j such that the component $a_{i_j} \in R_q$ is invertible modulo $p_j(x)$.

Proof. This can be directly proved by contradiction. Assume it is not the case and for some $j \in \{1, \ldots, d\}$ we have that $a_i \equiv 0$ modulo $p_j(x)$ for all $i \in \{1, \ldots, k\}$.

Then we can define m_0 such that $m_0 \equiv 0 \mod p_{j'}(x)$ for all $j' \neq j$ and $m_0 \equiv 1 \mod p_j(x)$. By construction $m_0 \neq 0$ but $\mathbf{a}m_0 = 0$, and therefore, we could open any commitment to any message m to a different message $m + m_0$, breaking the binding property of the commitment scheme. □

We explicitly check this condition (and abort if it is not satisfied) in the key generation protocol, but it will hold except with negligible probability if the used set of parameters is secure. We also output the specific indexes i_j and the inverses ${a_{i_j}}^{-1} \mod p_j$ as auxiliary parameters during the key generation protocol (even if omitted during the text, it is implicit that participants receiving the public key $(\mathbf{a}, \mathbf{b})$ also get these auxiliary elements precomputed).

With that property in mind now we can see that, provided a vector of polynomials $\mathbf{z} \in R_q^k$ if there exists a polynomial t such that $\mathbf{z} = \mathbf{a} \cdot t$ this t should verify that $z_{i_j} \equiv a_{i_j} t \mod p_j(x)$ and we could completely determine it from $t \equiv {a_{i_j}}^{-1} z_{i_j} \mod p_j(x)$.

Since we have already restricted the possible candidates for t to one single polynomial, we would just need to check that for every $i \in \{1, \ldots, k\}$ and every $j \in \{1, \ldots, d\}$ it holds that $z_i \equiv a_i {a_{i_j}}^{-1} z_{i_j} \mod p_j(x)$.

Observe that, in order to check if $\mathbf{z} \in \mathcal{L}(\mathbf{a})$ it is not even necessary to explicitly compute the t such that $\mathbf{z} = \mathbf{a} \cdot t$, as, using again the invertibility of the a_{i_j}, it is sufficient to just check $a_{i_j} z_i \equiv z_{i_j} a_i \mod p_j(x)$.

Following a similar argument for the linear relation ZKPoK in order to ensure that $t_3 = \lambda_1 t_1 + \lambda_2 t_2$ we just check that $\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3 \in \mathcal{L}(\mathbf{a})$ and $\mathbf{z}_{3,i_j} = \lambda_1 \mathbf{z}_{1,i_j} + \lambda_2 \mathbf{z}_{2,i_j}$ for every $j \in \{1, \ldots, d\}$.

Only in the case of the multiplicative relation protocol, we have chosen to explicitly compute internally the actual t's involved. In this case, we thought computations were cleaner this way, but an argument similar to the previous could work too.

4.2 Discrete Gaussian sampling

We want our schemes to be useful for as many types of devices as possible. For this reason, we have chosen to instantiate discrete Gaussian sampling using the *Discrete Ziggurat* (DZ) algorithm [21]. We have implemented the protocol from scratch, and it can work with multi-precision arithmetic. This protocol allows us to choose a trade-off between computational speed and memory usage. When we have no restriction on memory consumption, DZ achieves a performance close to the fastest known algorithm for this task [23]. Using DZ has another advantage, it allows us to directly sample from a bounded discrete Gaussian distribution. The details of DZ are out of scope of this work, and we refer the interested reader to the original paper.

4.3 Uniform sampling

Even if computationally simpler than Gaussian sampling, the implemented protocols make heavy use of uniform sampling of integers in $\mathbb{Z}_q$ and it ends up being more computationally expensive.

Our implementation makes use of the RAND_priv_bytes function from OpenSSL as a source of pseudorandomness. To sample from $\mathbb{Z}_q$ we have to do rejection sampling. We can sample $\lceil \log_{2^8}(q) \rceil$ bytes to get an integer uniformly distributed in $[0, 2^{8\lceil \log_{2^8}(q) \rceil} - 1]$ and then reduce it $\mod 2^{\lceil \log_2(q) \rceil}$ to get an integer s uniformly distributed in $[0, 2^{\lceil \log_2(q) \rceil} - 1]$. We keep s if $s < q$ and otherwise we discard it and start again.

The intermediate reduction modulo a power of 2 guarantees a reasonable rejection rate smaller than $1/2$, which is really worthy as reducing modulo a power of two can be done at a low level and is computationally cheap, so this approach turns out to be a faster option than similar alternatives which require reductions modulo q.

If we have to sample multiple integers, for example to sample vectors of polynomials in $\mathbb{Z}_q[x]/\langle x^n + 1 \rangle$, in order to amortize the cost of calling the RAND_priv_bytes function we first populate a buffer of pseudorandom bytes and use them sequentially, only refreshing the buffer with new randomness when there are less than $\lceil \log_{2^8}(q) \rceil$ bytes left.

To deterministically sample integers from $\mathbb{Z}_q$ using a seed we use the same approach, replacing RAND_priv_bytes with an XOF as SHAKE-128 initialized with both the seed and a counter that keeps track of how many times we have already refreshed the buffer. We denote by PRN the function that takes as input a given seed and outputs the desired number of pseudorandom integers modulo q following this procedure.

We also need to uniformly sample multiple permutations from the set of permutations of $2nk$ elements. We encode a permutation as a product of $2nk$ transpositions $(1, i_1)(2, i_2) \dots (2nk - 1, i_{2nk-1})(2nk, i_{2nk})$ where (j, i_j) is such that $i_j \geq j$. That way, sampling $2nk$ indexes uniformly in the respective intervals $i_j \leftarrow [j, 2nk]$ determines a permutation uniformly distributed in the set of permutations of $2nk$ elements (i.e. following the Fisher-Yates algorithm).

We can deterministically compute such pseudorandom indexes from a seed in a similar manner than before. Sampling from an interval $[j, 2nk]$ is equivalent to sampling from $[0, 2nk - j]$ and then adding j. However, for this case, where indexes to be sampled are much smaller, we alternate the order of the rejection and the modular reduction to reduce the rejection rate. To sample an integer uniformly from $[0, 2nk - j]$ we can also sample an integer uniformly from $[0, m(2nk - j + 1) - 1]$ for any $m \in \mathbb{Z}$ and then reduce it $\mod (2nk - j + 1)$. The greater multiple of $(2nk - j + 1)$ smaller or equal than $2^{8\lceil \log_{2^8}(2nk-j+1) \rceil}$ is $2^{8\lceil \log_{2^8}(2nk-j+1) \rceil} - (2^{8\lceil \log_{2^8}(2nk-j+1) \rceil} \mod (2nk - j + 1))$. We start sampling $s \leftarrow [0, 2^{8\lceil \log_{2^8}(2nk-j+1) \rceil} - 1]$. Then we reduce it $\mod (2nk - j + 1)$ and add j if $s < 2^{8\lceil \log_{2^8}(2nk-j+1) \rceil} - (2^{8\lceil \log_{2^8}(2nk-j+1) \rceil} \mod (2nk - j + 1))$ or reject it and repeat otherwise.

We denote by π_τ the permutation obtained following this procedure from a

seed $\tau \in \{0,1\}^{8\lceil \lambda/8 \rceil}$.

This approach greatly reduces the rejection rate and improves the efficiency of the algorithm. It was not the case for the $\mathbb{Z}_q$ sampler because in that case the modular reduction over q would cost more than the modular reduction over a power of two, and the benefit obtained from the reduced rejection rate would not compensate that. Now we are in a different scenario, provided that $2nk$ is much smaller than q and modular reductions are much more efficient when we do not have to deal with arbitrary large integers.

4.4 Multiplication in a truncated polynomial ring

It is well-known that the fastest algorithm to date to compute products of polynomials in a ring is the *Fast Fourier Transform* (FFT) which can do so in quasilinear time. To be able to use the power of FFT it is required that the polynomial $x^n + 1$ splits into linear factors. Unfortunately, the security proof of our scheme heavily relies on $x^n + 1$ not splitting into many factors. To tackle this issue we changed the condition of the modulus q so that it splits into d factors of degree n/d in order to be able to implement a polynomial multiplication algorithm using a partial FFT which allows as fast as possible multiplications for this kind of rings. This family of techniques are folklore, but the particular details on how and when can we implement different generalizations and how to precompute the required auxiliary elements might be cumbersome and are detailed in [33].

This partial FFT involves computing as many steps of the regular FFT as possible, ending with d polynomials of degree n/d in the transformed domain, that can be multiplied using any other multiplication algorithm. We choose to multiply the d pairs of smaller polynomials using Karatsuba's multiplication algorithm.

Most of the computations we have to do, as sampling uniformly random polynomials, adding them and verifying equalities, can also be done in the transformed domain, so in our implementation almost every polynomial is directly represented in the transformed domain and never anti-transformed back.

5 Instantiation and performance

In this section we first explain how to obtain secure sets of parameters to instantiate our commitment scheme, and then we provide some results, in terms of running time and size, for both the commitment algorithms and the companion NIZKPoK.

As we have already mentioned, we estimate the hardness of the underlying RLWE problem using the Lattice Estimator of Albrecht *et al.* [1].

5.1 How to choose parameters

We have already seen in Section 3 (and it will be developed in Appendix A) that defining the security theoretically, i.e. with asymptotic proofs, requires a different analysis than the one necessary for claiming a specific level of security for a particular set of parameters. Besides that, finding the best set of parameters that fulfills these particular conditions is not direct either.

The requirement of equation (5') states that a probability has to be bounded, but we do not know how to explicitly compute that probability. Then our only option is to impose a stronger condition that implies (5'). We can do it using two auxiliary functions, $\mathsf{boundedPr}$ and $\mathsf{vecBoundedPr}$.

$$\mathsf{boundedPr}(\sigma, B) \coloneqq \max\left\{0, \frac{B^2}{2\sigma^2}\log_2(e) - 1\right\}$$

Proposition 2. *boundedPr outputs the greater a for which we can guarantee that the probability of a sample from D_σ not belonging to $[-B, B]$ is lower or equal than 2^{-a}.*

Proof. Directly comes from Lemma 4.4 in [27]. □

$$\mathsf{vecBoundedPr}(a, d) \coloneqq \max\{a - \log_2(d), 0\}$$

Proposition 3. *vecBoundedPr outputs the greater b for which we can ensure that the sub-infinity norm of a vector of dimension d not being bounded by a certain bound has a probability smaller or equal to 2^{-b} if a single sample has a probability smaller or equal to 2^{-a}.*

Proof. Direct, iteratively applying the triangular inequality to bound the statistical distance. □

Then, in practice, we replace (5') with

$$\mathsf{vecBoundedPr}(\mathsf{boundedPr}(\sigma, B-1), k \cdot n) \geq \lambda. \tag{5}$$

Notice that we use $B-1$ as the event we want to consider is given by the error terms belonging to $[-B, B)$ which is ensured if they belong to $[-(B-1), B-1]$. A more tight analysis of this probability would not significantly change the final parameters.

An additional problem is to choose the *best* set of parameters that satisfies the required conditions. There are many inequalities that should hold at the same time, and reducing one parameter might end up making another condition more restrictive and yielding us to a worse global outcome.

To obtain our results, we have chosen to carefully study the selection of parameters and define a specific procedure that ends up in an optimal outcome for this particular scheme. We summarize it here and detail the procedure (proving and explaining in what sense the outcome is optimal) in Appendix C.

Given λ, n and q (that also define the possible d such that $q \equiv 2d + 1 \mod 4d$) we can compute the rest of parameters.

We can explicitly compute the minimum number of repetitions for the opening or linear and multiplicative ZKPoK, δ_{OL} and δ_M respectively, needed to satisfy eqs. (6) and (7), and then we just increase them until they also satisfy eq. (9).

We notice that, if the bound B was fixed, we could obtain the optimal (in this case smaller possible) k from the binding condition. Similarly, if B and k were fixed we could find the optimal (in this case greatest possible) σ that still satisfies the hiding condition related to the tails probability and check if it is enough for the hiding requirement regarding the hardness of the underlying RLWE problem.

The best B is the minimum one that ensures that the implied best k allows the existence of an available σ. We compute it explicitly, testing increasing powers of 2 until we find the first that works.

5.2 Size and running time

In this section we present the sizes of the commitment, the NIZKPoK of a valid opening, a linear and a multiplicative relation, and the running time for the keygen, commit, verifier, proveropening, verifieropening, proverlinear, verifierlinear, provermultiplicative and verifiermultiplicative algorithms for some sets of parameters of interest.

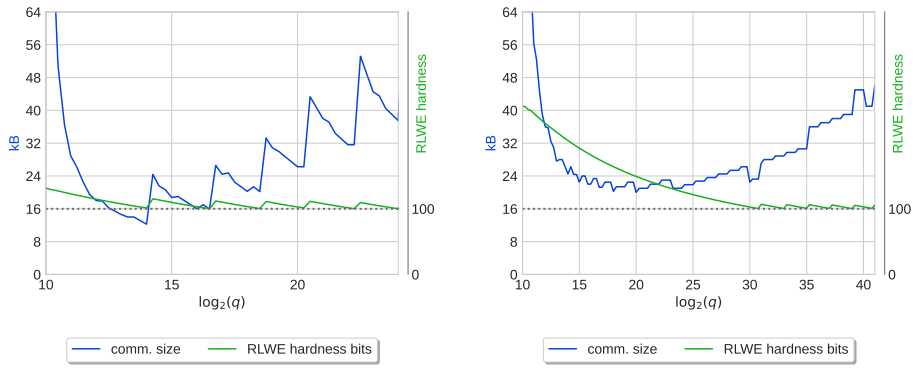
The performance values presented in this section and in Appendix D correspond to the average of 100 executions of each algorithm, compiled with gcc (version 11.3.0 and -O3 -march=native flags) using an Intel® Core™ i7-8700 CPU (6 cores and 12 threads) running at 3.20GHz with Ubuntu 22.04.1 LTS and 16GB of RAM.

We have chosen to represent sets of parameters with $\lambda = 100$ security bits, lattice dimension $n \in \{512, 1024\}$, and $d = 2$ as long as the multiplicative NIZKPoK takes less than 512 MB. For each modulus bitsize b we have chosen four q modules such that $2^b < q < 2^{b+1}$ (equally spaced in the log scale). Finally, for each of these (λ, n, q, d) the optimal set of parameters has been computed following Section 5.1.

This is not an exhaustive list of parameters. In Appendix D we extend these results to $d = 4$, which is the only other possibility for which secure sets of parameters exist, but we will see that $d = 2$ is, by all means, the best choice. Even if we do not test the running times of parameter sets when the size of the multiplicative proof would exceed 512 MB we include the size analysis of additional sets of parameters, again in Appendix D, and explain why there is no possible trade-off that would provide better results.

We start by plotting in Figure 1 the size of the commitment in kB for the best set of parameters against the size of the modulus, for both $n = 512$ and $n = 1024$, together with the hardness of the underlying RLWE problem.

It is interesting to see a zig-zag pattern in the $n = 512$ figure that we can easily understand looking at the RLWE hardness data. The discrete nature



(a) $\lambda = 100,\ n = 512,\ d = 2$ (b) $\lambda = 100,\ n = 1024,\ d = 2$

Figure 1 Commitment sizes

of the parameters, for example the power of two bound B of the noise terms, means that even if we target 2^{100} as RLWE hardness the minimum B that induces a σ that provides that level already achieves a greater value. This margin allows increasing q without having to increment B, because even if the hardness decreases when the ratio of σ/q does it might still be greater than 2^{100}. The intermingled restrictions from the security conditions imply that, while B can remain constant, even if choosing a greater q implies using greater integers the minimum dimension of the vectors k decreases and the commitment sizes can be smaller. This happens up to the point when the security would decrease below the 2^{100} level, which forces us to use the next power of 2 for B and that modifies all the relations ending up with the jumps visible in Figure 1.

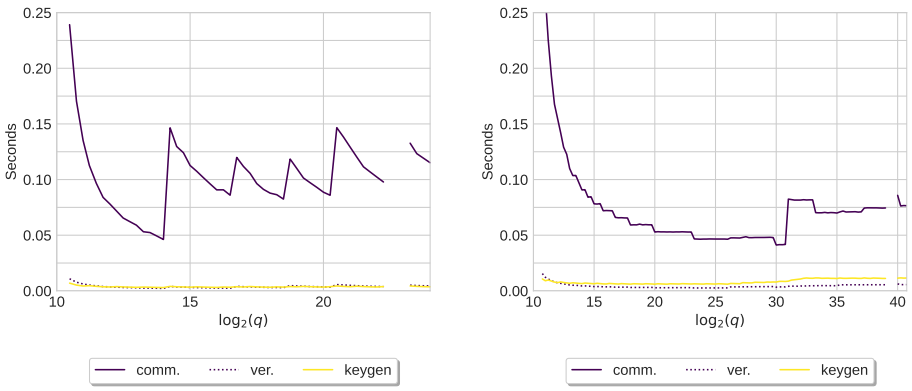
While the smaller size is achieved with $n = 512$ the commitments with $n = 1024$ start with a much harder RLWE problem, and that grants the existence of a much wider range of still reasonable sizes, given that the jumps start for much greater q and the zig-zag pattern is much less prominent.

This non-smooth behavior, a consequence of the numerous non-linear restrictions on the parameters, forces the final user to do a thorough study of the available parameters for this kind of schemes, like the one we are presenting in this section, because even starting with very similar q we could end up defining two instances with very different performance.

We then plot the running times of the key generation, the commitment, and the verification of an opening in Figure 2.

The times are highly correlated with the sizes, and the jumps are even more significant. We see that key generation and verification are extremely fast, and committing can also be done very efficiently.

The NIZKPoK sizes are plotted in Figure 3 and running times in Figure 4. Proofs of linear and multiplicative relations take almost the same space and have the same running time. In both cases, it is roughly three times the amount required for the proof of knowledge of a valid opening.



(a) $\lambda = 100,\ n = 512,\ d = 2$ (b) $\lambda = 100,\ n = 1024,\ d = 2$

Figure 2 Commitment times

Using the proposed NIZKPoK, we can prove knowledge of an opening under a second, and enjoy an even faster verification of the proofs, slightly above half the time required for generating the proof. We see however that in this case, the performance escalates worse with larger q than the commitment time. That is the case because now an increase in B not only affects the rest of parameters but also the size and running time themselves, because the bit decomposition of the errors implies that both grow with $\log_2(B)$.

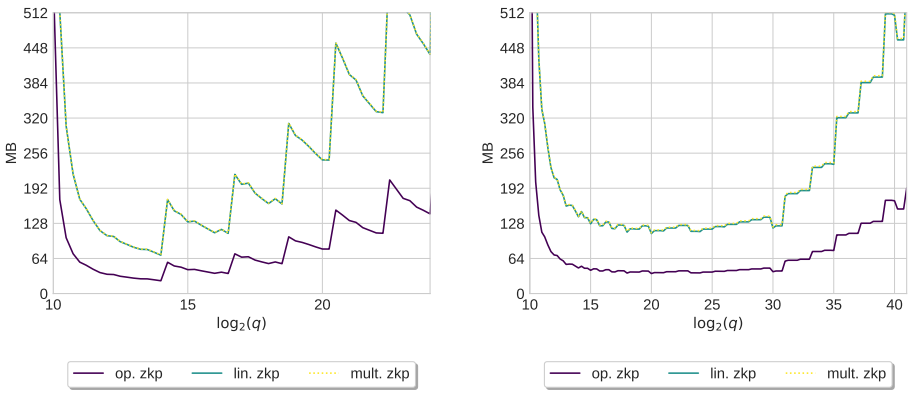
In what follows we present several secure sets of parameters that we choose to highlight because they are optimum regarding size, time or commitment to message ratio. We introduce them in Table 1, show their commitment and proof sizes in Table 2 and their associated running times for each protocol in Table 3. For each of these criteria, we have picked the best with $n = 512$ and the best with $n = 1024$, highlighting in boldface the best value for each n. Some sets are optimal regarding many different aspects, while others only stand out in one table and not in the others.

We want to remark that Table 1 contains only one set of parameters with $n = 512$ because this set is the best option regarding the ratio k between the commitment and the message size as well as size and running time of every protocol. It is not the case with $n = 1024$ where different sets are better for different characteristics. We present in boldface the k that are optimal for each n.

It is noticeable that all the modulus q are smaller than 2^{32}, so arbitrary-precision arithmetic is not essential for these sets of parameters. While decomposing the errors into bits seems a great overhead, it is not that expensive provided that the bound B can be as small as just 8.

Then in Table 2 we show the commitment and proof sizes of the previous sets of parameters, again highlighting in boldface the best cases for each n.

We see that the commitment sizes that we can obtain are completely prac-



(a) $\lambda = 100,\ n = 512,\ d = 2$ (b) $\lambda = 100,\ n = 1024,\ d = 2$

Figure 3 NIZKPoK sizes

Table 1: Best parameters with $n = 512$ and $n = 1024$

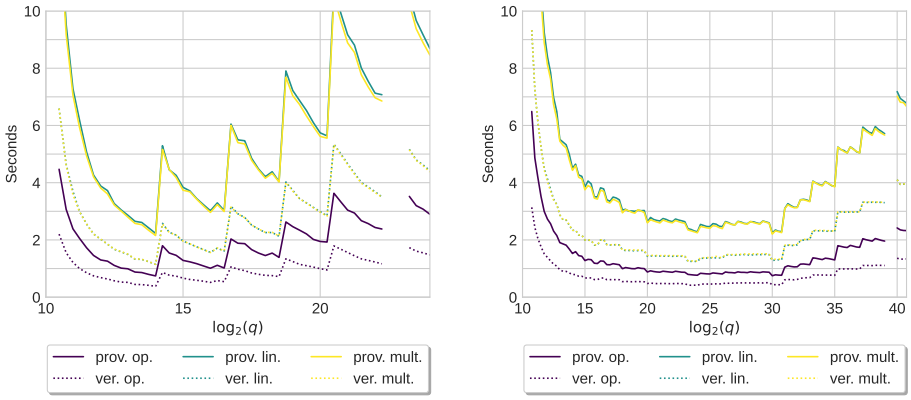
n	q	d	λ	k	σ	B	δ_{OL}	δ_M
512	16381	2	100	**14**	0.55	8	221	221
1024	1048573	2	100	8	0.55	8	213	213
1024	11863253	2	100	7	0.55	8	211	211
1024	16777213	2	100	7	0.55	8	210	210
1024	67108837	2	100	7	0.55	8	209	209
1024	1073741789	2	100	**6**	0.55	8	208	208
1024	1276901389	2	100	**6**	0.55	8	208	208
1024	1518500213	2	100	**6**	0.55	8	208	208
1024	1805811253	2	100	**6**	0.55	8	208	208

tical. On the other hand, proof sizes are still quite large. We see here that we have 12 kB commitments when $n = 512$ and around 20 kB commitments when $n = 1024$. The opening proof size with $n = 512$ is of the order of 20 MB, and almost a 60% more when $n = 1024$ and tripling the respective opening proof size for relation proofs.

We finally present all protocol times, in milliseconds, in Table 3. Again, we highlight with boldface the optimum results.

Execution time is definitely the main asset of this scheme, as we already obtain very fast committing and verifying time and almost practical proving and proof verifying times regarding that we have only developed a prototype implementation that could benefit from higher parallelization and further optimizations. It is interesting to notice that regarding running time, there is not much difference between $n = 512$ and $n = 1024$.

It is important to notice that while we can obtain the smallest commitment



(a) $\lambda = 100,\ n = 512,\ d = 2$ (b) $\lambda = 100,\ n = 1024,\ d = 2$

Figure 4 NIZKPoK times

Table 2: Sizes of the best parameters

n	q	com. size	op. size	lin. size	mult. size
512	16381	**12 kB**	**23.34 MB**	**69.99 MB**	**70.48 MB**
1024	1048573	**20 kB**	**36.47 MB**	**109.38 MB**	**110.70 MB**
1024	11863253	21 kB	37.83 MB	113.45 MB	115.02 MB
1024	16777213	21 kB	37.65 MB	112.91 MB	114.48 MB
1024	67108837	22 kB	40.53 MB	121.56 MB	123.24 MB
1024	1073741789	22 kB	39.85 MB	119.53 MB	121.46 MB
1024	1276901389	23 kB	41.16 MB	123.46 MB	125.45 MB
1024	1518500213	23 kB	41.16 MB	123.46 MB	125.45 MB
1024	1805811253	23 kB	41.16 MB	123.46 MB	125.45 MB

size for secure instantiations with $n = 512$ the ratio between the size of the commitment and the size of the committed message can be much smaller, $k = 6$, if we choose $n = 1024$, and even if the sizes increase the running times are still reasonable.

This alone proves the value of the approach from [32]. Even if the commitment itself has the same structure as the one from [10] the latter imposes

$$k > \frac{18\lfloor \log_2(q)/\log_2(n) \rfloor}{3\lfloor \log_2(q)/\log_2(n) \rfloor - 16}$$

with the additional restriction of $\lfloor \log_2(q)/\log_2(n) \rfloor > 6$. While asymptotically the bound is $k > 6$ for the minimal possible $q \sim 2^{70}$ when $n = 1024$ one would start with $k \geq 26$, and even increasing $q \sim 2^{100}$ one would still have $k \geq 13$.

Even with the modifications proposed in [9], that substantially change the original scheme using the Module Learning With Errors (MLWE) problem, while

Table 3: Running time of the best parameters (in milliseconds)

n	q	com.	ver.	key.	$\mathcal{P}_{op}$	$\mathcal{V}_{op}$	$\mathcal{P}_{lin}$	$\mathcal{V}_{lin}$	$\mathcal{P}_{mult}$	$\mathcal{V}_{mult}$
512	16381	**46**	**2**	**3**	**742**	**380**	**2228**	**1146**	**2154**	**1149**
1024	1048573	53	**3**	**6**	877	478	2627	1426	2595	1453
1024	11863253	46	**3**	**6**	783	416	2365	1251	2315	1266
1024	16777213	47	**3**	**6**	765	**413**	2286	**1243**	2250	**1260**
1024	67108837	46	**3**	**6**	810	462	2462	1371	2394	1383
1024	1073741789	**41**	**3**	8	**748**	437	**2244**	1275	**2202**	1298
1024	1276901389	42	4	9	781	434	2348	1301	2318	1326
1024	1518500213	**41**	4	9	770	435	2302	1307	2286	1326
1024	1805811253	42	4	8	764	434	2290	1306	2255	1326

their proposed overhead is again $k = 6$, they still need larger q and their proposed instantiation of a modified [10] produces 54.5 kB commitments, much greater than the values we have obtained.

In the previous graphs and tables the verifier times measure the time required to verify the total δ_{OL} or δ_M parallel verifications from which, on average, half of them correspond to $b_i = 0$ challenges and half of them to $b_i = 1$ challenges. It is relevant to point out that the second case is much faster than the first. We provide a general implementation where these challenges b_i are chosen pseudorandomly approximating a uniform distribution over $\{0, 1\}$. However, a specific implementation for a particular set of parameters where the specific running-time ratio between the two options can be computed could be optimized drawing the challenges from a non-uniform distribution, choosing more frequently the faster option $b = 1$. That would require an increase in the number of rounds to preserve the security level, but it could produce a significant net benefit (the details of this strategy, called Thrifty Zero-Knowledge, are well described in [15]).

5.3 Discussion and trade-offs

In this section, we provide a few additional comments regarding our implementation and possible changes to it that offer trade-offs that could be beneficial for some applications.

Discussion

- **Disclaimer.** Our implementation is a prototype, and it is therefore not ready for production. We provide it with a pure academic interest in mind. Nevertheless, we have tried to offer the most secure instantiation of our protocols as possible, but we are aware that there are security checks that we have not done. For example, we have not tried to make

our implementation constant time. Our tests only intend to benchmark the protocols, so commitments, proofs and intermediate results are kept in memory. Only the first two should be output to a file in order to later verify them, ensuring that the verifier has no access to the prover's private information. Taking care of these security issues is beyond the scope of this work and should be considered by any interested party that wants to use and deploy our protocols in a real-world scenario.

- **Non-exhaustive list of parameters.** We have provided experiments for a significant number of parameters. We believe that these should be enough to cover most of the use-cases that could take advantage of our protocols. However, our list is not exhaustive and there are other sets of parameters that could be of interest which we have not considered in this work.

Possible trade-offs

- **Interactive vs non-interactive.** We provide the non-interactive version of the protocols of[32]. However, there might be a case where an interactive version is needed, and our implementation can be adapted to the interactive setting with minor changes. Provided that in an interactive identification scheme scenario multiple failed attempts could be easily detected, it could be feasible to consider admissible non-negligible soundness errors such as the standard 2^{-32} instead of the 2^{-100} bound considered in the non-interactive case where the cheating probability can be amplified retrying again and again. Given that our protocol uses parallel repetitions to reduce the soundness error, the communication size would proportionally decrease if the targeted threshold can be reduced.

- **Native integers vs arbitrarily long ones.** To account for a wide range of applications, our instantiation works with arbitrarily long integers. However, this adds an unnecessary overhead on time and memory when the modulus q can fit in a native integer of the given language. Adapting our code to work only for native integers is a time-consuming task but straightforward task that we believe could be worth undertaking for some practical scenarios.

6 Conclusion and future work

We have provided an efficient and flexible implementation of the lattice-based commitment scheme proposed in[32]. Besides the commitment scheme, we have also implemented the non-interactive version of the ZKPoK for its opening and linear and multiplicative relations. Moreover, this work shows that current theoretical lattice-based schemes can be made usable in practice under careful implementation choices.

That is, with our implementation we can commit to a value in less than 50 milliseconds. The key generation and verification of a commitment are extremely fast, taking between 2 and 6 milliseconds. The prover's work for an opening takes 0.7 seconds, and for a linear/multiplicative relation it takes about 2 seconds. Verification costs a little more than half that time. Regarding sizes, a commitment requires 12 kB for $n = 512$ and 20 kB for $n = 1024$. An opening requires around 24 MB for $n = 512$ and 37 MB for $n = 1024$. Finally, for a linear/multiplicative relation, we need about 70 MB for $n = 512$ and about 110 MB for $n = 1024$.

We have given an extensive and detailed analysis on the modifications required to port the protocols presented in[32] from theory to practice. To do so, we have reworked the security proofs, moving away from a theoretical asymptotic analysis and focusing on unveiling how the security requirements affect our choice of parameters.

A thorough explanation on how to choose optimal parameters and our preferences for performing key computational tasks such as lattice membership testing, discrete Gaussian sampling and multiplication in a truncated polynomial ring is also provided.

Although we have tried to include as many optimizations as possible, there are still some interesting paths to explore that we believe could lead to efficiency improvements. We explain some of them below.

Throughout this paper, we have been balancing two clashing outcomes regarding the error size. The hiding property of the commitment scheme benefits from larger errors that increase the hardness of the underlying RLWE problem, but we had to bound them in order to be able to define the correctness of the commitment and the soundness of the ZKPoK, getting greater sizes the higher the bound has to be.

In order to enforce hiding, we have used that the probability of the tails is small enough so that the statistical distance between the two distributions (truncated and not) is again small. Besides that, there is an alternative similar strategy that allows us to relate the probability of an undesired event (breaking the hiding property) when we replace a distribution using the Rényi divergence of the two distributions (see [25] for the definition and its probability preservation property and [8] for how to compute the divergence between Gaussians and truncated Gaussians) instead of their statistical distance. It would require a specific analysis, more involved than the one with the statistical distance, but it would lead to bounds that might allow increasing the σ parameter without increasing the bound B (or to reduce the bound B without decreasing σ) and still preserve hiding, so we leave it as future work.

We also want to remark that, provided that additions and multiplications are compatible with the partial FFT, one can choose to define as message spaces the quotients of R_q over each of the factors $p_1(x), p_2(x), \ldots, p_d(x)$ of $x^n + 1$ and compute commitments and proofs to d polynomials of degree n/d (directly from the transformed domain) in parallel. However, it would be even more interesting to explore how the binding proof would improve if we just restrict the message space to $R_q / \langle p_1(x) \rangle$ and encode a message $m \in R_q / \langle p_1(x) \rangle$ as the polynomial

$m' \in R_q$ such that $m \equiv m' \mod p_i(x)$ for i from 1 to d, that way ensuring that the difference of encodings of different messages is different modulo each $p_i(x)$.

This strategy might provide an interesting trade-off between the message space size and the commitments and proofs sizes and efficiency, that is out of the scope of this work as it greatly deviates from the original scheme that we are implementing but should be considered for applications where the size of the message space can be reduced.

Finally, an additional research path would be to redesign these ZKPoK to base their security in the Module Learning With Errors instead of the Ring Learning With Errors problem, provided that the most succinct lattice-based commitments and ZKPoK to date are based on the first, and it would be interesting to analyze how practical are Stern-based techniques when dealing with the MLWE the same way we have done in this paper with the RLWE case.

Regardless of the possible improvements we believe that the thorough analysis presented in this work is of interest to the community, even if the techniques described are not the smallest lattice-based NIZKPoK, because it is important to get quantitative results in order to be able to compare different alternatives and find out the main bottlenecks. It is also important to provide detailed instructions on how the parameters have been obtained because lattice-based cryptography is an active research field, and these computations should always be easily recomputable just updating the hardness estimation (in our case using a new version of the lattice estimator module), without requiring every reader to redo the analysis on its own in order to get up-to-date secure parameters or different security levels.

Declarations

- Funding: This work is supported by the European Union PROMETHEUS project (Horizon 2020 Research and Innovation Program, grant 780701), the Spanish Ministry of Economy and Competitiveness under Project MTM2016-77213-R and the Spanish Ministry of Science and Innovation under Projects PID2019-109379RB-I00 and RTI2018-102112-B-I00.

- Competing interests: All authors certify that they have no affiliations with or involvement in any organization or entity with any financial interest or non-financial interest in the subject matter or materials discussed in this manuscript.

References

[1] Martin R Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathematical Cryptology*, 9(3):169–203, 2015.

[2] Richard Arratia and Louis Gordon. Tutorial on large deviations for the binomial distribution. *Bulletin of mathematical biology*, 51(1):125–131, 1989.

[3] Thomas Attema, Ronald Cramer, and Lisa Kohl. A compressed Σ-protocol theory for lattices. In Tal Malkin and Chris Peikert, editors, *CRYPTO 2021, Part II*, volume 12826 of *LNCS*, pages 549–579, Virtual Event, August 16–20, 2021. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-84245-1_19.

[4] Thomas Attema and Serge Fehr. Parallel repetition of $(k_1, \ldots, k_\mu)$-special-sound multi-round interactive proofs. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part I*, volume 13507 of *LNCS*, pages 415–443, California, USA, August 15–18, 2022. Springer, Heidelberg, Germany. doi:10.1007/978-3-031-15802-5_15.

[5] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. Cryptology ePrint Archive, Report 2021/1377, 2021. https://eprint.iacr.org/2021/1377.

[6] Thomas Attema, Serge Fehr, and Michael Klooß. Fiat-shamir transformation of multi-round interactive proofs. In Eike Kiltz and Vinod Vaikuntanathan, editors, *TCC 2022, Part I*, volume 13747 of *LNCS*, pages 113–142, Chicago, IL, USA, November 7–10, 2022. Springer, Heidelberg, Germany. doi:10.1007/978-3-031-22318-1_5.

[7] Thomas Attema, Vadim Lyubashevsky, and Gregor Seiler. Practical product proofs for lattice commitments. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 470–499, California, USA, August 17–21, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-56880-1_17.

[8] Shi Bai, Tancrède Lepoint, Adeline Roux-Langlois, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. *Journal of Cryptology*, 31(2):610–640, April 2018. doi:10.1007/s00145-017-9265-9.

[9] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 368–385, Amalfi, Italy, September 5–7, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-98113-0_20.

[10] Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In Günther Pernul, Peter Y. A. Ryan, and Edgar R. Weippl, editors, *ESORICS 2015, Part I*, volume 9326 of *LNCS*, pages 305–325, Vienna, Austria, September 21–25, 2015. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-24174-6_16.

[11] Ward Beullens. Sigma protocols for MQ, PKP and SIS, and Fishy signature schemes. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part III*, volume 12107 of *LNCS*, pages 183–211, Zagreb, Croatia, May 10–14, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-45727-3_7.

[12] Jonathan Bootle, Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. A non-PCP approach to succinct quantum-safe zero-knowledge. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part II*, volume 12171 of *LNCS*, pages 441–469, California, USA, August 17–21, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-56880-1_16.

[13] Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 176–202, California, USA, August 18–22, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-26948-7_7.

[14] Ming-Shing Chen, Andreas Hülsing, Joost Rijneveld, Simona Samardjiska, and Peter Schwabe. From 5-pass MQ-based identification to MQ-based signatures. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 135–165, Hanoi, Vietnam, December 4–8, 2016. Springer, Heidelberg, Germany. doi:10.1007/978-3-662-53890-6_5.

[15] Simon Cogliani, Houda Ferradi, Rémi Géraud, and David Naccache. Thrifty zero-knowledge. In Feng Bao, Liqun Chen, Robert H. Deng, and Guojun Wang, editors, *Information Security Practice and Experience*, pages 344–353, Cham, 2016. Springer International Publishing.

[16] Jelle Don, Serge Fehr, and Christian Majenz. The measure-and-reprogram technique 2.0: Multi-round fiat-shamir and more. In Daniele Micciancio and Thomas Ristenpart, editors, *CRYPTO 2020, Part III*, volume 12172 of *LNCS*, pages 602–631, California, USA, August 17–21, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-56877-1_21.

[17] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 356–383, California, USA, August 18–22, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-26951-7_13.

[18] Muhammed F. Esgin, Ngoc Khanh Nguyen, and Gregor Seiler. Practical exact proofs from lattices: New techniques to exploit fully-splitting rings. In Shiho Moriai and Huaxiong Wang, editors, *ASIACRYPT 2020, Part II*, volume 12492 of *LNCS*, pages 259–288, Daejeon, South Korea, December 7–11,

2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-64834-3_9.

[19] Muhammed F. Esgin, Ron Steinfeld, Joseph K. Liu, and Dongxi Liu. Lattice-based zero-knowledge proofs: New techniques for shorter and faster constructions and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 115–146, California, USA, August 18–22, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-26948-7_5.

[20] Abhishek Jain, Stephan Krenn, Krzysztof Pietrzak, and Aris Tentes. Commitments and efficient zero-knowledge proofs from learning parity with noise. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 663–680, Beijing, China, December 2–6, 2012. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-34961-4_40.

[21] Buchmann Johannes, Cabarcas Daniel, G¨opfert Florian, H¨ulsing Andreas, and Patrick Weiden. Discrete Ziggurat: A Time-Memory Trade-off for Sampling from a Gaussian Distribution over the Integers Johannes. *Selected Areas in Cryptography – SAC 2013. SAC 2013. Lecture Notes in Computer Science*, 8282, 2013. doi:10.1007/978-3-662-43414-7_20.

[22] Daniel Kales and Greg Zaverucha. An attack on some signature schemes constructed from five-pass identification schemes. In Stephan Krenn, Haya Shulman, and Serge Vaudenay, editors, *CANS 20*, volume 12579 of *LNCS*, pages 3–22, Vienna, Austria, December 14–16, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-65411-5_1.

[23] D. Knuth and A. Yao. *Algorithms and Complexity: New Directions and Recent Results*, chapter The complexity of nonuniform random number generation. Academic Press, Orlando, USA, 1976.

[24] Veronika Kuchta, Amin Sakzad, Ron Steinfeld, and Joseph K. Liu. Lattice-based zero-knowledge arguments for additive and multiplicative relations. *Designs, Codes, and Cryptography*, 89(5):925 – 963, 2021. URL: https://www.scopus.com/inward/record.uri?eid=2-s2.0-85102024795&doi=10.1007%2fs10623-021-00851-1&partnerID=40&md5=9eb6bb6828951701a504a75d39132b0c, doi:10.1007/s10623-021-00851-1.

[25] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. GGHLite: More efficient multilinear maps from ideal lattices. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 239–256, Copenhagen, Denmark, May 11–15, 2014. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-55220-5_14.

[26] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Lattice-based zero-knowledge arguments for integer relations. In Hovav Shacham and Alexandra Boldyreva, editors, *CRYPTO 2018, Part II*, volume 10992

of *LNCS*, pages 700–732, California, USA, August 19–23, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-96881-0_24.

[27] Vadim Lyubashevsky. Lattice signatures without trapdoors. Cryptology ePrint Archive, Report 2011/537, 2011. https://eprint.iacr.org/2011/537.

[28] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Maxime Plançon. Lattice-based zero-knowledge proofs and applications: Shorter, simpler, and more general. In Yevgeniy Dodis and Thomas Shrimpton, editors, *CRYPTO 2022, Part II*, volume 13508 of *LNCS*, pages 71–101, California, USA, August 15–18, 2022. Springer, Heidelberg, Germany. doi:10.1007/978-3-031-15979-4_3.

[29] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Practical lattice-based zero-knowledge proofs for integer relations. In Jay Ligatti, Xinming Ou, Jonathan Katz, and Giovanni Vigna, editors, *ACM CCS 2020*, pages 1051–1070, Virtual Event, USA, November 9–13, 2020. ACM Press. doi:10.1145/3372297.3417894.

[30] Vadim Lyubashevsky, Ngoc Khanh Nguyen, and Gregor Seiler. Shorter lattice-based zero-knowledge proofs via one-time commitments. In Juan Garay, editor, *PKC 2021, Part I*, volume 12710 of *LNCS*, pages 215–241, Virtual Event, May 10–13, 2021. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-75245-3_9.

[31] Vadim Lyubashevsky and Gregor Seiler. Short, invertible elements in partially splitting cyclotomic rings and applications to lattice-based zero-knowledge proofs. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 204–224, Tel Aviv, Israel, April 29 – May 3, 2018. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-78381-9_8.

[32] Ramiro Martínez and Paz Morillo. RLWE-based zero-knowledge proofs for linear and multiplicative relations. In Martin Albrecht, editor, *17th IMA International Conference on Cryptography and Coding*, volume 11929 of *LNCS*, pages 252–277, Oxford, UK, December 16–18, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-35199-1_13.

[33] Ramiro Martínez and Paz Morillo. Revisiting fast fourier multiplication algorithms on quotient rings, 2023. doi:10.48550/arXiv.2304.08860.

[34] P. W. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994. doi:10.1109/SFCS.1994.365700.

[35] Yongcheng Song, Jiang Zhang, Xinyi Huang, Wei Wu, and Haining Yang. Statistical zero-knowledge and analysis of rank-metric zero-knowledge proofs of knowledge. *Theoretical Computer Science*, 952:113731,

2023. URL: https://www.sciencedirect.com/science/article/pii/S0304397523000440, doi:10.1016/j.tcs.2023.113731.

[36] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 617–635, Tokyo, Japan, December 6–10, 2009. Springer, Heidelberg, Germany. doi:10.1007/978-3-642-10366-7_36.

[37] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 13–21, California, USA, August 22–26, 1994. Springer, Heidelberg, Germany. doi:10.1007/3-540-48329-2_2.

[38] Yang Tao, Xi Wang, and Rui Zhang. Short zero-knowledge proof of knowledge for lattice-based commitment. In Jintai Ding and Jean-Pierre Tillich, editors, *Post-Quantum Cryptography - 11th International Conference, PQCrypto 2020*, pages 268–283, Paris, France, April 15–17, 2020. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-44223-1_15.

[39] Xiang Xie, Rui Xue, and Minqian Wang. Zero knowledge proofs from ring-LWE. In Michel Abdalla, Cristina Nita-Rotaru, and Ricardo Dahab, editors, *CANS 13*, volume 8257 of *LNCS*, pages 57–73, Paraty, Brazil, November 20–22, 2013. Springer, Heidelberg, Germany. doi:10.1007/978-3-319-02937-5_4.

[40] 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 Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 147–175, California, USA, August 18–22, 2019. Springer, Heidelberg, Germany. doi:10.1007/978-3-030-26948-7_6.

Appendix A Security Proofs

Throughout this appendix, we are going to establish sufficient conditions for each of the security properties of the commitment scheme and the associated NIZKPoK.

A.1 Binding

The binding property proofs of [10, 32] rely on the algebraic structure of $R_q = \mathbb{Z}_q[x] / \langle x^n + 1 \rangle$. This structure heavily depends on the modulus q, as it has an impact on the factorization of $x^n + 1$, which is irreducible over the integers but splits into two irreducible polynomials of degree $n/2$ when $q \equiv 3 \pmod 8$ [36].

However, changing this structure allows some efficiency improvements. Lyubashevsky and Seiler proved in [31] an interesting characterization of R_q whenever q satisfies a different condition, summarized in Lemma 4.

Lemma 4 (Corollary 1.2 in [31]). *Let $n \geq d > 1$ be powers of 2 and $q \equiv 2d + 1 \pmod{4d}$ be a prime. Then the polynomial $x^n + 1$ factors as*

$$x^n + 1 \equiv \prod_{j=1}^{d} \left(x^{n/d} - \alpha_j \right) \pmod q$$

for distinct $\alpha_j \in \mathbb{Z}_q^*$, where $x^{n/d} - \alpha_j$ are irreducible in R_q.

That is the condition we are going to use.

$$q \equiv 2d + 1 \pmod{4d}. \qquad \text{(eq. (1))}$$

There is an important trade-off in the result of Lemma 4, the more factors the faster the multiplication of ring elements can be implemented (the optimum is achieved when $x^n + 1$ splits into linear factors, and we can apply FFT multiplication), but achieving the binding property becomes harder. Considering that we might end up with a larger commitment size or, if d is too large, we might end up not being able to find a secure set of parameters at all.

In the following paragraphs, we redo the binding property proof for a modulus q satisfying the conditions of Lemma 4 with an arbitrary d. That is, we want to prove that, with overwhelming probability over the choice of $\mathbf{a}, \mathbf{b} \in R_q^k$ we have that

$$\text{accept} \leftarrow \mathsf{Ver}\left(\mathbf{c}, m', o'; pk\right), \text{ and}$$
$$\text{accept} \leftarrow \mathsf{Ver}\left(\mathbf{c}, m'', o''; pk\right) \implies m' = m''.$$

Two accepted openings to the same commitment would be:

$$\mathbf{c} = \mathbf{a}m' + \mathbf{b}r' + \mathbf{e}'$$
$$\mathbf{c} = \mathbf{a}m'' + \mathbf{b}r'' + \mathbf{e}''$$

Therefore, if $m' \neq m''$ we have that $\mathbf{a}(m' - m'') + \mathbf{b}(r' - r'') + (\mathbf{e}' - \mathbf{e}'') = 0$. If $q \equiv 2d + 1 \pmod{4d}$, with overwhelming probability over the choice of $\mathbf{a}$ and

$\mathbf{b}$, and provided that k is large enough, there are no $m, r \in R_q$ and $\mathbf{e} \in R_q^k$ small such that $\mathbf{a}m + \mathbf{b}r + \mathbf{e} = 0$ holds and $m \neq 0$.

We bound the probability that this solution exists. For a fixed m, r and $\mathbf{e}$ we count the proportion of pairs $(\mathbf{a}, \mathbf{b})$ for which the equality holds. In order to estimate the overall probability of choosing a pair $(\mathbf{a}, \mathbf{b})$ such that there exists a solution, we use a union bound adding up all previous probabilities. We finally see that it is negligible if parameters are carefully selected.

Fixed m, r and $\mathbf{e}$ for each $\mathbf{b}$ we have $\mathbf{a}m = -\mathbf{b}r - \mathbf{e}$. In each component $a_i m = -b_i r - e_i$. $q \equiv 2d + 1 \pmod{4d}$ implies that $x^n + 1$ splits into d irreducible polynomials $p_1(x), p_2(x), \ldots, p_d(x)$ of degree n/d. We know that $m \not\equiv 0 \mod x^n + 1$, therefore, $m \not\equiv 0 \mod p_j(x)$ for some $j \in \{1, \ldots, d\}$.

For this case we know that choosing different a_i we have that $a_i m$ takes at least $q^{n/d}$ different values $\pmod{p_j(x)}$. There are $q^{n/d}$ equivalence classes $\pmod{p_j(x)}$ and only one of them is $-b_i r - e_i \pmod{p_j(x)}$, therefore at most $1/q^{n/d}$ of the possible a_i hold the equation. As this is independently true for each i, we have that the probability of $(\mathbf{a}, \mathbf{b})$ to fit the equation for these particular m, r and $\mathbf{e}$ is at most $1/q^{nk/d}$.

If we want to consider the possibility that there exists a solution, we can bound this probability with a union bound. There are q^n possible m, q^n possible r and $(4B-1)^{kn}$ possible $\mathbf{e}$. Therefore, if $(\mathbf{a}, \mathbf{b}) \leftarrow \mathsf{keygen}(n, q, d, k)$, we can upper bound the probability

$$\Pr_{(\mathbf{a},\mathbf{b})}\left[\exists m, r, \mathbf{e} \middle| \begin{array}{c} \mathbf{a}m + \mathbf{b}r + \mathbf{e} = 0 \\ \wedge \mathbf{e} \in [-2B+1, 2B-1]^k \end{array}\right]$$

by

$$\frac{q^{2n}(4B-1)^{kn}}{q^{nk/d}} \leq 2^{-\lambda}.$$

Taking logarithms, the condition that has to be satisfied is

$$n(2\log_2(q) + k(\log_2(4B-1) - \log_2(q)/d)) \leq -\lambda.$$

If we ensure $(\log_2(4B-1) - \log_2(q)/d)$ is negative then we only need to choose

$$k \geq \frac{\lambda + 2n\log_2(q)}{n(\log_2(q)/d - \log_2(4B-1))}. \qquad \text{(eq. (2))}$$

Notice here the first condition just imposes the maximum value of d we can still use, preserving the binding property

$$d < \frac{\log_2(q)}{\log_2(4B-1)}. \qquad \text{(eq. (3))}$$

The smaller the error bound B gets the more options we have for this trade-off. As larger bounds also increase the size of the proof, our goal would be to find the smaller bound B that still ensures the hiding property.

A.2 Hiding

The advantage of an adversary against the hiding property is defined as the ability they would have distinguishing two commitments to two different messages chosen by themselves. More formally, we define $Adv_{\mathsf{hid}}(\mathcal{A})$ as

$$\left| \Pr \left[b = b' \;\middle|\; \begin{array}{c} (m_0, m_1, aux) \leftarrow \mathcal{A}_1(\mathbf{a}, \mathbf{b}), \\ r \leftarrow R_q,\ \mathbf{e} \leftarrow {D_{\sigma,B}^n}^k \\ b' \leftarrow \{0,1\},\ \mathbf{c} = \mathbf{a}m_{b'} + \mathbf{b}r + \mathbf{e} \\ b \leftarrow \mathcal{A}_2(\mathbf{c}, aux) \end{array} \right] - \frac{1}{2} \right|.$$

From a standard analysis, we can ensure

$$\begin{aligned} &Adv_{\mathsf{hid}}\left(\mathcal{A}\right) \leq \\ &\max_{\mathcal{B}} \left\{ Adv_{\mathsf{RLWE}}(\mathcal{B}) \right\} + \Pr\left[\left|\left|\mathbf{e}\right|\right|_\infty > B \middle| \mathbf{e} \leftarrow {D_\sigma}^{kn} \right], \end{aligned}$$

where $Adv_{\mathsf{RLWE}}(\mathcal{B})$ is the advantage of another adversary against the decisional version of the RLWE problem.

The usual reduction implies that the maximum is taken among adversaries $\mathcal{B}$ that perform the same plus a constant number of operations as $\mathcal{A}$. Its particular advantage against the decisional RLWE problem can not be precisely bounded either, as the best we can do is estimate the hardness of the underlying problem using Albrecht *et al.*'s Lattice Estimator.

For that matter, we just ensure that

$$\mathsf{bitsec}(RLWE) \geq \lambda. \qquad \text{(eq. (4))}$$

The other term comes from the fact that we are not using discrete Gaussian distributed errors but truncated discrete Gaussian distributed errors. The advantage difference can be bounded by the statistical distance among the discrete Gaussian distribution and the truncated distribution, that is, precisely the probability of the tails we are omitting. For that reason, since we want the advantage to be negligible we need to ensure that

$$\Pr\left[\left|\left|\mathbf{e}\right|\right|_\infty > B \;\middle|\; \mathbf{e} \leftarrow {D_\sigma}^{kn} \right] \leq 2^{-\lambda}. \qquad \text{(eq. (5'))}$$

As we have mentioned, it is implied by

$$\mathsf{vecBoundedPr}(\mathsf{boundedPr}(\sigma, B-1), k \cdot n) \geq \lambda. \qquad \text{(eq. (5))}$$

A.3 Soundness

Soundness comes from the knowledge-soundness of the interactive version from [32] and the soundness preservation of the Fiat-Shamir transform. For that reason, eqs. (6) and (7) are directly inherited from [32].

Notice that the original soundness proof from [32] implicitly assumes that given $\mathbf{z} \in \mathcal{L}(\mathbf{a})$ one can compute $t \in R_q$ such that $\mathbf{z} = \mathbf{a}t$. This has been shown to be true under the conditions that imply the binding property of the commitment in Section 4.1.

The proof is based on the ROM, sampling the first challenges α_i (and β_i in the multiplicative proof) following the rejection sampling procedure described in Section 4.3 that uses SHAKE-128 as XOF and the second challenge directly from SHAKE-128 again both using as input the statement of the proof and the previous elements of the underlying interactive conversation. Both pseudorandom functions are modelled as random oracles to apply the Fiat-Shamir transform.

The equivalence between special-soundness and knowledge-soundness and the soundness preservation of the Fiat-Shamir transform are immediate for Σ-protocols (the canonical 3-move form of most interactive ZKPoK protocols). Even if sometimes it is taken for granted that is not the case for protocols with a non-negligible soundness error that therefore require parallel repetitions, protocols with more than 3 moves, or protocols with both more than 3 moves and parallel repetitions (which is our case).

A.3.1 Knowledge-Soundness for interactive $(2\mu + 1)$-move protocols with parallel repetitions

The [32] soundness proof already partially addressed these issues, providing a witness extractor sufficient to ensure soundness of the interactive protocol. We further improve their analysis to show that the protocols satisfy the standard definition of knowledge-soundness.

Soundness of an interactive protocol imposes that for any malicious prover $\mathcal{P}^*$ we have $\Pr\left[\langle \mathcal{P}^*, \mathcal{V}\rangle (x) = \textit{accept} \;\middle|\; x \notin \mathfrak{R}\right] \leq \epsilon$. We can request a stronger property, ensuring that the prover not only can prove that x is valid but also that they know a witness w of $(x, w) \in \mathfrak{R}$. More formally we say that there exists an extractor that, given oracle access to a successful enough prover, can be used to efficiently compute a witness with sufficient probability.

Definition 5 (Knowledge-Soundness of an interactive protocol as in [4])**.** *An interactive proof $\langle \mathcal{P}, \mathcal{V}\rangle$ for relation $\mathfrak{R}$ is* knowledge sound *with* knowledge error $\kappa(x)$ *if there exists a positive polynomial p and an algorithm $\mathcal{E}$ with the following*

properties. The extractor $\mathcal{E}$ given input x and rewindable oracle access to a (potentially dishonest) prover $\mathcal{P}^*$ is expected to run in polynomial time in $|x|$ and outputs a witness w such that $(x,w) \in \mathfrak{R}$ with probability

$$\Pr\left[(x,w) \in \mathfrak{R} \;\middle|\; w \leftarrow \mathcal{E}^{\mathcal{P}^*}(x)\right] \geq \frac{\Pr\left[\langle \mathcal{P}^*, \mathcal{V}\rangle(x) = accept\right] - \kappa(x)}{p(|x|)}.$$

It implies the regular soundness definition (if $\kappa < \epsilon$ a witness can be computed with a positive probability and therefore x is valid) and can usually be more directly proved.

The additional property of being public-coin is usually combined with the previous one, but we prefer to define it separately.

Definition 6 (Public-Coin)**.** *An interactive protocol $\langle \mathcal{P}, \mathcal{V}\rangle$ is* public-coin *if all of $\mathcal{V}$'s random choices are made public.*

To be able to prove soundness, [32] uses yet another property called k-special-soundness (see [32] for the formal definition) that states that a witness can be extracted from k accepting conversations with different challenges. Notice that this does not immediate provide knowledge-soundness, as one has to build the extractor that efficiently gets these k conversations.

The original soundness proof from [32] manages to prove that, provided oracle access to a malicious prover that produces accepting conversations with probability $\epsilon \geq ((q+1)/2q)^\delta$, lets call $\kappa = (q+1)/2q$, there exists an efficient extractor that outputs a valid witness with probability $2(\epsilon - \kappa^\delta)^3/27$. It was sufficient to prove soundness of the interactive proof for which it is sufficient to see that a witness can be extracted with positive probability, but it does not satisfy the standard notion of knowledge-soundness with knowledge error κ^δ that would require the probability to be proportional to $(\epsilon - \kappa^\delta)$, while that proof only achieves a relaxed version where we have a cubic loss in the probability. The standard definition also asks for an extractor that, given oracle access to any successful enough malicious prover, outputs a witness with sufficient probability. The current proof proves the existence of such an extractor that works with each prover, but that extractor could be different each time. That is because the original proof shows that it is sufficient to focus on a single individual thread of the δ repetitions, however the index of that thread could be different for each prover. This could be naïvely addressed and the extractor can be made universal by choosing that index at random (at the expense of dividing by δ the guaranteed success probability) or trying it for every index (at the expense of increasing its computational cost by a factor δ).

Many proposals in the literature do not even analyze how the extractor would work. As it was mentioned in [32] the potential execution tree considering all

possible challenges would be of the order of q^δ. Their proposed extractor would work in a running time just proportional to q (or to δq to comply with the standard definition as we have seen). Once a useful node in the execution tree (an initial message for which the prover can correctly answer to sufficiently many challenges) is found, all the $q+2$ transcripts have to be obtained rewinding the prover. That is still a significant work regarding that only 4 of them are really relevant, as the witness can be obtained from 4 transcripts with α as the first challenge in two of them, $\alpha' \neq \alpha$ in the other two, and both $b = 0$ and $b = 1$ for each α. We can say, as it is done in [32], that $q+2$ transcripts with different pairs of challenges in $\mathbb{Z}_q \times \{0, 1\}$ ensure by the pigeonhole principle the existence of the four transcripts that we really need, but it seems an overkill as the k-special-soundness definition from [32] does not fully capture the nature of multi-round protocols. A more fine-graded definition for $(2\mu + 1)$-protocols is the following.

Definition 7 ($(k_1, \ldots, k_\mu)$-Special Soundness as in [3])**.** *A $(2\mu+1)$-move public-coin protocol is $(k_1, \ldots, k_\mu)$-special sound if there is an efficient algorithm that on input $\prod_{j=1}^{\mu} k_j$ accepting transcripts*

$$\left\{(x, a, \{c_{i_1,\ldots,i_j}\}_{j=1}^{\mu}, \{b_{i_1,\ldots,i_j}\}_{j=1}^{\mu})\right\}_{i_j \in \{1,\ldots,k_j\}}$$

such that $c_{i_1,\ldots,i_j} \neq c_{i_1,\ldots,i'_j}$ when $i_j \neq i'_j$, outputs a witness w such that $(x, w) \in \mathfrak{R}$.

It is clear that a single iteration of our opening protocol is $(2, 2)$-special sound, as we just need four transcripts with α and α' as the first challenge and both $b = 0$ and $b = 1$ as the second. The same applies to the protocol proving a linear relation.

The multiplicative protocol is more interesting, as we need six pairs (β, α_1), (β, α_2), (β, α_3) and (β', α'_1), (β', α'_2), (β', α'_3) for which there are transcripts for both $b = 0$ and $b = 1$. This structure is that of a $(2, 3, 2)$-special sound 7-move protocol. Of course, our 5-move protocol can be artificially transformed into that just sending β as first challenge, waiting for an empty response from the prover, and finally sending α as a second challenge. Since these two protocols are by all means equivalent any property that we can deduce when interpreting it as a $(2, 3, 2)$-special sound 7-move interactive protocol would also be satisfied by our 5-move multiplicative protocol (the non-interactive version would differ as it would require a different number of oracle calls, for that reason we would think of it as a 7-move protocol when proving properties of the interactive version and as a 5-move protocol when transforming it into a non-interactive protocol).

To see how it implies knowledge-soundness it remains to specify how to efficiently obtain such tree of conversations (see Figure 1 of [4] for a nice visualization of the tree structure). A general tight efficient extractor was first defined in [3]. It is significantly more involved to extract the useful accepting conversations from δ parallel repetitions of a $(k_1, \ldots, k_\mu)$-special sound protocol.

Fortunately [4] proves (Theorem 3 in their paper) that a $(k_1, \ldots, k_\mu)$-special sound protocol is knowledge sound with knowledge error

$$\kappa = 1 - \prod_{j=1}^{\mu} \frac{|\mathcal{C}_j| - k_j + 1}{|\mathcal{C}_j|}.$$

That means a single repetition of the opening or linear interactive protocols would be knowledge sound with knowledge error $\kappa_{OL} = (q+1)/2q$ and a single repetition of the interactive multiplicative protocol would be knowledge sound with knowledge error $\kappa_M = (q^2 + 3q - 2)/2q^2$.

Furthermore, the really novel contribution of [4] (presented in their Theorem 4) is that the parallel repetition of a $(k_1, \ldots, k_\mu)$-special sound protocol δ times is knowledge sound with knowledge error κ^δ (when the δ parallel repetitions effectively reduce the knowledge error from κ to κ^δ they call it a strong parallel repetition result).

This knowledge error is the same as the soundness error obtained by [32] (equal to the standard cheating probability obtained by a prover that tries to guess the challenge in advance) but now the knowledge-soundness satisfies the standard definition with probability directly proportional to $(\epsilon - \kappa^\delta)$. Besides that the extractor would only requiere a number of rewinds proportional to $\prod_{i=1}^{\mu} k_i$ (proportional to 4 in the opening and linear or 12 for the multiplicative protocol), instead of proportional to q.

A.3.2 Multiple vs. single challenge set

Notice that our notion of a public-coin interactive protocol assumes that in each move the challenge is uniformly drawn from a possibly different challenge set $\mathcal{C}_i$. Some articles proving security of the Fiat-Shamir transform, as [6] or [16], assume for convenience that all challenges are drawn from a single universal challenge set $\mathcal{C}$. In [6] the authors mention that the proof of their main result (which unfortunately does not apply to our protocols) could be rewritten to admit arbitrary challenge sets. We think that a detailed general proof of why this assumption of having a single challenge set can be taken without any loss of generality in the ROM is of independent interest and it allows us to use any transformation regardless of their challenge sets convention without any extra work.

Theorem 8 (Single vs. Multiple Challenge sets)**.** *Any public-coin* $(2\mu + 1)$*-move Honest Verifier Zero-Knowledge Proof of Knowledge between a prover* $\mathcal{P}$ *and a verifier* $\mathcal{V}$*, where challenge* c_i *in the* i*th round is uniformly sampled by* $\mathcal{V}$ *from a challenge set* $\mathcal{C}_i$*, can be transformed (in the ROM) into another public-coin* $(2\mu+1)$*-move Honest Verifier Zero-Knowledge Proof of Knowledge between*

another prover $\mathcal{P}'$ and another verifier $\mathcal{V}'$ where every challenge s_i is uniformly sampled from a sufficiently large single challenge set $\mathcal{S}$ preserving completness, Zero-Knowledge, soundness and knowledge-soundness.

That new protocol is defined in the following way. $\mathcal{P}'$ computes its initial commitment a as $\mathcal{P}$ would do and sends it to $\mathcal{V}'$. In each round i the verifier $\mathcal{V}'$ samples a seed uniformly at random from the challenge set $s_i \leftarrow \mathcal{S}$, and the prover $\mathcal{P}'$ then calls a random oracle $\mathcal{O}_i$ that outputs uniformly random elements c_i from $\mathcal{C}_i$ if the seed had not been called before or the previous response otherwise. Then $\mathcal{P}'$ answers b_i in the same way $\mathcal{P}$ would do if challenged with that c_i. Finally, $\mathcal{V}'$ checks the verifying equations as $\mathcal{V}$ would do with the challenges $c_i \leftarrow \mathcal{O}_i(s_i)$.

Proof. It is clear that completeness is directly inherited from the original protocol. Honest Verifier Zero-Knowledge is also preserved, as one can compute the c_i calling the oracles with the s_i and then use the original simulator. Provided that the oracles output uniformly random c_i and after that point everything is computed in the same manner, the simulated conversations would again follow the same distributions as the ones between honest $\mathcal{P}'$ and $\mathcal{V}'$.

The only properties that require a dedicated insightful analysis are soundness and knowledge-soundness. Let $\mathcal{P}^\circ$ be a malicious prover against the single challenge set protocol with access to μ oracles $\{\mathcal{O}_i\}_{i=1}^{\mu}$ and a success probability ϵ. We can also construct a prover $\mathcal{P}^*$ against the multiple challenge protocol, interacting with $\mathcal{P}^\circ$ providing its challenges and simulating their oracle queries. We will refer to s_i as the inner challenges and the corresponding $c_i \leftarrow \mathcal{O}_i(s_i)$ as the outer challenges. We can do it following the next procedure.

$\mathcal{P}^*$ starts running $\mathcal{P}^\circ$ to produce the first commitment a. Whenever $\mathcal{P}^\circ$ calls oracle $\mathcal{O}_i$ with an $s \in \mathcal{S}$ then $\mathcal{P}^*$ samples $c \leftarrow \mathcal{C}_i$ uniformly at random if s was never queried before or returns the previously sampled or programmed element. To compute the $s_i \in \mathcal{S}$ inner challenge seeds prover $\mathcal{P}^*$ gets challenges c_i from the verifier and outputs with some probability $p_i^{(c_i)}$ a uniformly random seed from the ones that have already been queried to the oracle $\mathcal{O}_i$ and had been answered with the outer challenge c_i or otherwise $\mathcal{P}^*$ chooses a uniformly random seed from the ones that have not been asked by $\mathcal{P}^\circ$ to the oracle $\mathcal{O}_i$ and program its simulated oracle to subsequently answer c_i to that seed s_i.

Let's define some disjoint partitions of $\mathcal{S}$ useful to define these probabilities and prove the desired properties. Let $\mathcal{S}_i^c \subseteq \mathcal{S}$ be the subset of seeds that have been oracle called by $\mathcal{P}^\circ$ to $\mathcal{O}_i$ and have received c as answer. Let $\mathcal{S}_i^{\neg c} \subseteq \mathcal{S}$ be the subset of seeds that have been oracle called by $\mathcal{P}^\circ$ and have received an answer different from c. And finally, let $\mathcal{S}_i^{\varnothing} \subseteq \mathcal{S}$ be the subset of seeds that have not been queried yet. That way at any time we have $\mathcal{S} = \mathcal{S}_i^c \sqcup \mathcal{S}_i^{\neg c} \sqcup \mathcal{S}_i^{\varnothing}$ for every i and c.

If we fix $p_i^{(c_i)} = (|\mathcal{S}_i^{c_i}| \cdot |\mathcal{C}_i|)/|\mathcal{S}|$ then the success probability of $\mathcal{P}^*$ would be that of $\mathcal{P}^\circ$. Intuitively the expected value of seeds that would be mapped to c_i would be $|\mathcal{S}| / |\mathcal{C}_i|$ and we already have $|\mathcal{S}_i^c|$ many so this seems the right guess for that probability, and we are going to confirm that it indeed works. Observe this is always a viable strategy if $|S|$ is large enough. On the one hand if $\mathcal{S}_i^c = \varnothing$ then $p_i^{(c)} = 0$ and we never have to sample from an empty set. On the other hand we can ensure that $p_i^{(c)} \leq 1$ if $|\mathcal{S}| \geq Q_i |\mathcal{C}_i|$ where Q_i is the number of oracle queries to oracle $\mathcal{O}_i$ and therefore is an upper bound of $|\mathcal{S}_i^c|$. This also guarantees that $\mathcal{S}_i^{\varnothing} \neq \varnothing$ either.

To prove that the success probability is the same, we need to check that the challenges s_i and the oracle answers simulated by $\mathcal{P}^*$ follow the same distribution than the ones $\mathcal{P}^\circ$ would receive interacting with an honest verifier and random oracles.

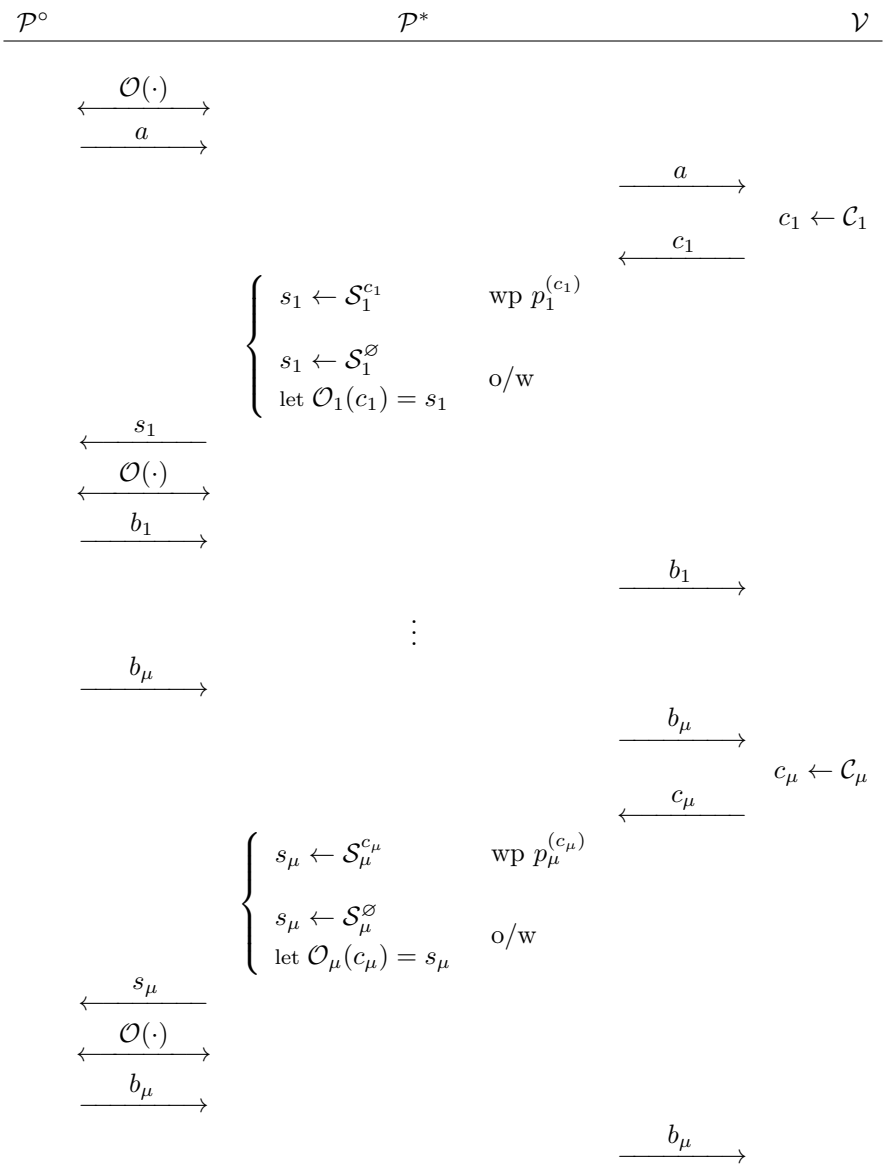
Let's analyze the probability of providing $\mathcal{P}^\circ$ with a given s as the ith inner challenge. Either this s has been submitted before to $\mathcal{O}_i$ by $\mathcal{P}^\circ$ or not. In the first case, let's denote by c the answer $\mathcal{P}^*$ gave it. The probability of choosing such s is the probability of the ith outer challenge being c multiplied by the probability $p_i^{(c)}$ of choosing s among the $\mathcal{S}_i^c$ already queried seeds that output c multiplied again by the probability of getting that particular seed when sampling uniformly from that set:

$$\Pr\left[s_i = s \;\middle|\; s \in \mathcal{S}_i^c\right] = \frac{1}{|\mathcal{C}_i|} \cdot \frac{|\mathcal{S}_i^c| \cdot |\mathcal{C}_i|}{|\mathcal{S}|} \cdot \frac{1}{|\mathcal{S}_i^c|} = \frac{1}{|\mathcal{S}|}$$

On the other hand, if s was not submitted before, i.e. $s \in \mathcal{S}_i^{\varnothing}$, for each possible challenge c its probability is now $1 - p_i^{(c)}$ multiplied by the probability of choosing that s from $\mathcal{S}_i^{\varnothing}$:

$$\begin{aligned}
\Pr\left[s_i = s \;\middle|\; s \in \mathcal{S}_i^{\varnothing}\right] &= \sum_{c \in \mathcal{C}_i} \frac{1}{|\mathcal{C}_i|} \cdot \frac{|\mathcal{S}| - |\mathcal{S}_i^c| \cdot |\mathcal{C}_i|}{|\mathcal{S}|} \cdot \frac{1}{\left|\mathcal{S}_i^{\varnothing}\right|} \\
&= \frac{1}{|\mathcal{S}|} \cdot \frac{|\mathcal{S}| - \sum_{c \in \mathcal{C}_i} |\mathcal{S}_i^c|}{\left|\mathcal{S}_i^{\varnothing}\right|} \\
&= \frac{1}{|\mathcal{S}|} \cdot \frac{\left|\mathcal{S}_i^{\varnothing}\right|}{\left|\mathcal{S}_i^{\varnothing}\right|} = \frac{1}{|\mathcal{S}|}
\end{aligned}$$

Then, as every s has the same probability, $\mathcal{P}^\circ$ receives challenges uniformly distributed in $\mathcal{S}$ the same way it would have if it was interacting with a real honest verifier.



Protocol 7: Multiple from single challenge set adversary.[4]

[4]The usual abbreviations *wp* and *o/w* are used for "with probability" and "otherwise".

Regarding its calls to the oracle, every query with an s that has not been set as inner challenge is answered as usual, sampling a uniform c. Every s_i that they have received as inner challenge has already a defined answered c_i. Nevertheless, provided that this c_i has been sent by an honest verifier it also follows a uniformly random distribution as an oracle call would do.

As we have granted that the interaction $\mathcal{P}^*$ has with $\mathcal{P}^\circ$ follows the same distribution as the interaction with a verifier and μ random oracles then the success probability is exactly the same, as we wanted to prove. That implies that the soundness error of the new single-oracle protocol is at most equal to the one from the multi-challenge protocol, because any success probability of the former can also be realized with the latter.

Knowledge-soundness preservation comes from the same transformation, as we can again use the constructed $\mathcal{P}^*$ with the original extractor to obtain a witness with the same probability. □

We can use this proof to ensure the security of a version with a single challenge set, which allows us to continue the proof using the literature that builds their theorems under this assumption. Although the reduction might seem artificial it is not far from the real world as our own implementation internally uses some seeds to generate the pseudorandom challenges, precisely as described (and therefore can be interpreted as having different challenge sets or a single one just depending on if we consider as challenge the seed or the corresponding pseudorandom output). This general proof is also easier to apply than redoing the proof of the Fiat-Shamir transform security or redefining $(k_1, \ldots, k_\mu)$-special soundness.

A.3.3 The Fiat-Shamir transform

Now that we have verified that the interactive version of our protocol would satisfy the standard notions of soundness we can apply the Fiat-Shamir transform to obtain a sound non-interactive proof.

With the last transformation, we can see that our protocol satisfies the definition of a *Public-Coin Interactive Proof* system (PCIP) as defined in [16], as they assume every challenge is sampled from the same set.

Definition 9 (Fiat-Shamir transform for general PCIP adapted from [16])**.** *The Fiat-Shamir transform for general PCIP defines a non-interactive proof system where the proof* Π *is generated by a prover* $\mathcal{P}_{FS}^H$ *by computing the prover messages of the transcript of an interactive protocol with challenges deterministically obtained from the previous elements,* $c_1 = H(0, x, a)$ *and* $c_i = H(i-1, x, c_{i-1}, b_{i-1})$ *for* $i > 1$*, where* H *is a hash function with the appropriate domain. The verifier*

$\mathcal{V}^H_{FS}$ *accepts if the transcript obtained when computing the challenges is accepting (we denote by* $\mathcal{V}^H_{FS}(x, \Pi)$ *the output of verifier* $\mathcal{V}^H_{FS}$ *when presented with a proof* Π *for a statement* x*).*

Remark 10. *Their original definition does not include the statement* x *in the* c_i *challenges with* $i > 1$*, as it is already implicit because each hash takes as input the previous challenge and the first does have* x *as part of its input. Nevertheless, they mention that "any additional strings can be included in the argument when computing* c_i *using* H*, without influencing the security properties of the non-interactive proof system in a detrimental way". This transform also corresponds to the alternative definition proposed in [5].*

The model assumes that H behaves as a (Quantum) Random Oracle and our goal is to prove soundness in that setting. We first recall that the soundness of non-interactive proof systems can not be directly addressed as before. The success probability of an adversary might unavoidably grow by just trying again if it fails at the first attempt. For that reason, we can only bound its probability when restricting the number of oracle queries Q, ensuring it only grows polynomially with it.

Definition 11 (Soundness of a non-interactive proof system $(\mathcal{P}^H_{FS}, \mathcal{V}^H_{FS})$, adapted from [17])**.** *We say that a non-interactive proof system* $(\mathcal{P}^H_{FS}, \mathcal{V}^H_{FS})$ *is* sound *if there exists a negligible function* $\eta(\lambda)$ *and a constant* e *such that for any adversary* $\mathcal{A}$ *making* Q *queries to a uniformly random* H *and any* $\lambda \in \mathbb{N}$*:*

$$\Pr_H \left[\mathcal{V}^H_{FS}(x, \Pi) = accept \wedge x \notin \mathfrak{R} \mid (x, \Pi) \leftarrow \mathcal{A}^H \right] \leq Q^e \eta(\lambda).$$

We can prove it in the QROM using the recent result from [16] that ensures the existence of a quantum algorithm $\mathcal{B}$ such that we can lower bound

$$\Pr \left[x = x^\circ \wedge v = accept \mid (x, v) \leftarrow \left\langle \mathcal{B}^{\mathcal{A}}, \mathcal{V} \right\rangle \right]$$

by

$$C \cdot \Pr_H \left[x = x^\circ \wedge \mathcal{V}^H_{FS}(x, \Pi) = accept \mid (x, \Pi) \leftarrow \mathcal{A}^H \right] - \epsilon_{x^\circ}$$

where $C = \frac{\mu!}{(2Q+\mu+1)^{2\mu}}$, the additive error term ϵ_{x° is equal to $\mu!/|\mathcal{C}|$ when summed over all x°, and can be made arbitrarily small. Following the notation from [16] $(x, v) \leftarrow \left\langle \mathcal{B}^{\mathcal{A}}, \mathcal{V} \right\rangle$ means that $\mathcal{B}^{\mathcal{A}}$ first outputs a statement x and then interacts with $\mathcal{V}$ so that $v = \left\langle \mathcal{B}^{\mathcal{A}}, \mathcal{V} \right\rangle (x)$.

As [16] mentions, this implies soundness preservation as long as the challenge space $\mathcal{C}$ has size superpolynomial in the security parameter since μ is constant and Q is polynomial in the security parameter. That makes ϵ_{x° negligible and we can also aim to make the bound on the success probability of the interactive version κ^δ negligible in λ by choosing large enough δ. There we have the negligible function required by the definition. Then, provided that μ is constant, $(2Q + \mu + 1)^{2\mu}/(\mu!)$ can be bounded by some power of Q.

In order to target a provable security level one would just need to choose $|\mathcal{C}|$ and δ large enough so that the $\kappa^\delta + \mu!/|\mathcal{C}|$ compensates the $(2Q+\mu+1)^{2\mu}/(\mu!)$ security loss. One could achieve a soundness error smaller than $2^{-\lambda}$ choosing both $\delta \approx \log_2(|\mathcal{C}|) \approx 5\lambda$, as we can always assume $Q < 2^\lambda$ and get $(\kappa^\delta + \mu!/|\mathcal{C}|)(2Q+\mu+1)^{2\mu}/(\mu!) \lesssim 2^{-\lambda}$, increasing the proofs by a factor of 5.

This $O(Q^{2\mu})$ security loss is tight in general, as discussed in [16]. The 2 in the exponent comes from the fact that we are considering the QROM, the quantum version of the ROM, and have to consider Grover-search attacks.

Similar proofs with the plain ROM exist with a security loss of just $(Q+1)^\mu$, as informally claimed in [6]. In this important work they show that for many interactive proofs, mainly those with $(k_1, \ldots, k_\mu)$-special soundness, one can prove a better bound on the security loss of only $O(\mu Q)$, linear instead of exponential in the number of rounds. However, that is not the case of the considered scheme, as the parallel repetition of a $(k_1, \ldots, k_\mu)$-special sound protocol is not $(k_1, \ldots, k_\mu)$-special sound itself.

Besides soundness, one can similarly prove knowledge-soundness because it is also preserved by Fiat-Shamir.

Definition 12 (Proof of Knowledge as in [17])**.** *The non-interactive proof system $(\mathcal{P}_{FS}, \mathcal{V}_{FS})$ is a computational proof of knowledge if there exists a polynomial-time algorithm $\mathcal{E}$, a polynomial $p(\eta)$, constants $d, e \geq 0$ and a negligible function $\mu(\eta)$, such that for any (quantum) polynomial-time algorithm $\mathcal{A}$ making at most Q oracle queries, any $\eta \in \mathbb{N}$ and any x we can bound*

$$\Pr\left[(x,w) \in \mathfrak{R} \;\middle|\; w \leftarrow \mathcal{E}^{\mathcal{A}}(x)\right]$$

by

$$\leq \frac{1}{Q^e p(\eta)} \Pr_H\left[\mathcal{V}_{FS}^H(x,\Pi) \;\middle|\; \Pi \leftarrow \mathcal{A}^H\right]^d - \mu(\eta).$$

It can be similarly defined for adaptive adversaries allowing $\mathcal{A}$ to choose x

As [16] mentions in their Corollary 15 the Proof of Knowledge property is preserved in the QROM because any dishonest prover against the Fiat-Shamir version with success probability ϵ can be used to get a dishonest prover against the interactive protocol with a success probability of $\epsilon \cdot (2Q+1)^{-2\mu}$, that can be used to extract a witness.

A.4 Zero-Knowledge

The only additional condition would be to use a cryptographically secure Pseudo Random Number Generator to obtain computational Zero-Knowledge.

For our prototype implementation, we choose SHA3-256 as hash function and SHAKE-128 as XOF.

A.5 Correctness

As we mentioned before, correctness is unconditional.

Appendix B Known Attacks

Quoting from [6]:

> *If one wants to rely on the proven security reduction, one needs to choose a large security parameter for* Π*, in order to compensate for the order* Q^μ *security loss, effecting its efficiency; alternatively, one has to give up on proven security and simply assume that the security loss is much milder than what the general bound suggests. Often, the security loss is simply ignored.*

Disregarding additional security losses is called the ϵ^δ-heuristic, as one directly assumes the soundness error exponentially decreases with parallel repetitions without considering any possible attacks [22].

However, even if assuming the security loss is milder than the worst case scenario we should still consider which is the best known attack. For example, the post-quantum signature scheme [14], which has a structure very similar to the one considered through this paper as it is based too on a $q2$-identification scheme, was successfully attacked in [22]. We remark that the existence of these attacks only implies that larger parameters should be chosen. We can ensure that the Fiat-Shamir transform is not inherently broken because if we sufficiently increase these parameters (number of repetitions and challenge space size) we could obtain provable security. Regarding the studied protocols, as far as we are concerned, the best known attack is the general strategy presented in [5] against the parallel repetition of $(k_1, \ldots, k_\mu)$-special sound protocols that satisfy an additional property they call $(l_1, \ldots, l_\mu)$-special unsoundness (sometimes $(l_1, \ldots, l_\mu)$-out-of-$(|\mathcal{C}_1|, \ldots, |\mathcal{C}_\mu|)$-special unsoundness) with N responses per round. Let's formally introduce that concept, show that the studied protocols satisfy the definition and reanalyze accordingly the security of the non-interactive proofs.

Definition 13 ($(l_1, \ldots, l_\mu)$-special unsoundness with N potential responses per round as in [6])**.** *We say that a public-coin interactive proof* $\langle \mathcal{P}, \mathcal{V} \rangle$ *has*

$(l_1, \ldots, l_\mu)$-special unsoundness *if there exists a dishonest prover $\mathcal{A}$ such that when interacting with $\mathcal{V}$ on input x the following holds:*

- *$\mathcal{A}$ starts in* active mode*, meaning that at every round after $\mathcal{A}$'s message there exists a subset $\Gamma_i \subseteq \mathcal{C}_i$ of size l_i such that if the challenge $c_i \in \Gamma_i$ then $\mathcal{A}$ switches to* passive mode*.*
- *If $\mathcal{A}$ switches to* passive mode *then it remains in* passive mode *and the final conversation is accepting.*

Besides that, we say that it has $(l_1, \ldots, l_\mu)$-special unsoundness with N potential responses per round *if during the active mode $\mathcal{A}$ can efficiently produce N distinct messages with the previously mentioned property.*

As they say, many $(k_1, \ldots, k_\mu)$-special sound protocols are also $(k_1-1, \ldots, k_\mu-1)$-special unsound. It is possible to design an adversary $\mathcal{A}$ against the interactive version of the opening protocol that shows it is $(1,1)$-special unsound.

In the first round $\mathcal{A}$ just needs to guess $\Gamma_1 = \{\widetilde{\alpha}\} \subseteq \mathbb{Z}_q$ and prepare both commitments so that they would be able to pass every verification if the challenge α turns out to be $\widetilde{\alpha}$. $\mathcal{A}$ can do so making up $\mathbf{e}'_j \leftarrow \mathfrak{B}_{nk}$ (so that the $\pi_{\tau_j}(\mathbf{e}'_j) \stackrel{?}{\in} \mathfrak{B}_{nk}$ verification checks out), and also making up $\mathbf{z} \leftarrow \mathcal{L}(\mathbf{a})$ and preparing the first commitment choosing $\mathbf{y} = \mathbf{z} - (\widetilde{\alpha}(\mathbf{c}+\mathbf{B}) - \mathbf{b}s - \phi(\sum_j 2^j(\mathbf{f}_j + \widetilde{\alpha}\mathbf{e}'_j)))$ with a random $s \in R_q$ so that everything checks if α turns out to be $\widetilde{\alpha}$ (as then $\widetilde{\mathbf{z}}$ would be $\mathbf{z}$ and therefore it would belong to $\mathcal{L}(\mathbf{a})$). If the guess is right they can switch to passive mode and satisfactorily answer with the made up s or $\mathbf{e}'_j$. Observe they have many options for $\mathbf{z}$, s or $\mathbf{e}'_j$ so it has the many responses per round property.

If the guess was wrong, the adversary could still try to guess the second challenge. If they guess that $\Gamma_2 = \{1\}$ then they only need to answer with the usual $\mathbf{g}_j = \pi_{\tau_j}(\mathbf{f}_j + \alpha\mathbf{e}'_j)$ and then they would be able to satisfactorily answer if the challenge is indeed $b = 1$ with the made up $\mathbf{e}'_j$. However, in this case they would only have one possibility. Therefore, to be able to have many responses per round the adversary should guess $\Gamma_2 = \{0\}$ and prepare the $\mathbf{g}_j$ so that the $b = 0$ checks are satisfied. It can just randomly sample $\mathbf{g}_j$ for $j \geq 1$ and then solve for $\mathbf{g}_0$ so that $\mathbf{z} = \mathbf{y} + \alpha(\mathbf{c}+\mathbf{B}) - \mathbf{b}s - \phi(\sum_j 2^j {\pi_{\tau_j}}^{-1}(\mathbf{g}_j))$ and they can satisfactorily answer if the challenge is $b = 0$. The freedom of choosing $\mathbf{g}_j$ for $j \geq 1$ now guarantees the many responses per round property.

The opening protocol is therefore $(1,1)$-special unsound with $O(q^{nk})$ responses per round. Then we can ensure the existence of an attack exploiting the parallel repetition structure of our non-interactive proof.

Theorem 14 (δ-fold parallel repetition attack from [5])**.** *Let Π be a $(2\mu+1)$-move public-coin interactive proof with challenge spaces $\mathcal{C}_1, \ldots, \mathcal{C}_\mu$. Suppose Π*

has $(l_1, \ldots, l_\mu)$-special unsoundness with N responses per round. Let Π^δ be the δ-fold parallel repetition of Π. Let $m_1, \ldots, m_\mu \in \mathbb{N}$ such that $\sum_{i=1}^{\mu} m_i = \delta$. Let $Q = \mu Q'$ for $Q' \in \mathbb{N}$ with $Q' \sum_{i=1}^{\mu} (l_i / |\mathcal{C}_i|)^{m_i} < 1/4$ and $Q' \leq N$. There is a Q-query dishonest prover against $FS[\Pi^\delta]$ so that for every statement x the probability

$$\Pr\left[\mathcal{V}_{FS}^{RO}(x, \mathcal{P}^{*,RO}) = accept\right]$$

is bigger than

$$\frac{1}{2}\left(\frac{Q}{\mu}\right)^{\mu} \prod_{i=1}^{\mu} (l_i / |\mathcal{C}_i|)^{m_i}.$$

The theorem from [5] provides a lower bound on the attacker success probability because they want to emphasize that the security loss can be $O(Q^\mu)$ and not only $O(\mu Q)$. However, in order to choose parameters so that the scheme is secure against this attack what we want is an upper bound. Lets describe the attack strategy and find that upper bound.

The idea is the following. Choosing $m_1, \ldots, m_\mu \in \mathbb{N}$ such that $\sum_{i=1}^{\mu} m_i = \delta$ the attacker tries to guess the challenges, because if $c_i \in \Gamma_i$ for some of the parallel executions then they would be able to change into passive mode for that repetition and continue answering till the end. Their goal is to get at least $\sum_{j=1}^{i} m_j$ parallel executions in passive mode before moving to the next round, so that they end with a complete fake proof. If not sufficiently many new guesses are correct in a given round to move to the next they just resample new messages to get fresh challenges. To get into account the fact that the adversary is computationally limited and only queries the oracle Q times its designed to abort after Q/μ unsuccessful attempts per round. See [5] for a full description.

We need to upper bound the success probability of the attack and select the number of repetitions in a way that ensures this upper bound is smaller than $2^{-\lambda}$. Lets first analyze the involved probability distributions and recap some useful probability propositions.

Definition 15 (Binomial Distribution)**.** *The* binomial distribution *of parameters n and p, denoted $B(n, p)$, is the discrete probability distribution that counts the number of successes in n independent experiments, each of them having two possible outcomes,* success *with probability p or* failure *with probability $1 - p$.*

Considering the attack at the ith round there are $\delta - m'_{i-1}$ parallel protocols in active mode (we denote by $m'_{i-1} \geq \sum_{j<i} m_j$, with $m'_0 = 0$ the number of repetitions in passive mode after the $(i-1)$th round) and each of them turns into passive mode with probability $l_i / |\mathcal{C}_i|$ (the probability of $c \in \Gamma_i$ if $c \leftarrow \mathcal{C}_i$). The number of protocol repetitions that turn into passive mode follows a binomial distribution of parameters $\delta - m'_{i-1}$ and $l_i / |\mathcal{C}_i|$.

Definition 16 (Geometric Distribution)**.** *The* geometric distribution *of parameter p, denoted $Geo(p)$, is the discrete probability distribution that counts the*

number of trials until the first success (including the successful one) when sequentially conducting independent experiments, each of them having two posible outcomes, success *with probability* p *or* failure *with probability* $1-p$.

The attacker advances a round of the protocol if the number of parallel repetitions that turn into passive mode is at least $(\sum_{j\leq i} m_j) - m'_{i-1}$, lets say that happens with probability p_i (it is the probability of a certain binomial distribution surpassing a certain threshold). If that is not the case then it computes a different message and tries again. If no limit was imposed, the number of trials until it continues to the next round would follow a geometric distribution of parameter p_i.

Particularizing that to our current protocol, let

$$X \sim B(\delta, q^{-1})$$

be the probability distribution that models the number of parallel repetitions in which the adversary would be able to correctly answer in passive mode. They continue if that number is at least m_1 (recall that $m_1, m_2 \in \mathbb{Z}_{\geq 0}$ are any two non-negative integers such that $m_1 + m_2 = \delta$). The number of attempts until that would happen would follow a distribution

$$Y \sim Geo(\Pr[X \geq m_1])$$

and therefore the adversary that aborts if they cannot continue after Q/μ tries advance to the second round with probability $\Pr[Y \leq Q/2]$.

Provided that m'_1 repetitions are already in passive mode, which happens with probability $\Pr[X = m'_1 \mid X \geq m_1]$, the adversary can only fake the proof if it guesses the remaining $\delta - m'_1$ challenges.

For that reason, we analogously define

$$W_m \sim B(\delta - m, 1/2) \text{ and,}$$

$$Z_m \sim Geo(\Pr[W_m = \delta - m])$$

as we might need to consider the case $m'_1 = m$ for any $m_1 \leq m \leq \delta$. These two distributions model the number of correctly guessed challenges and the number of trials until all remaining challenges are guessed. The adversary successfully fakes a proof if $Z_m \leq Q/\mu$.

In conclusion, the adversary success probability is

$$\epsilon = \Pr\left[Y \leq Q/2\right] \cdot \sum_{m=m_1}^{\delta} P_s(m),$$

where

$$P_s(m) := \Pr\left[X = m \;\middle|\; X \geq m_1\right] \Pr\left[Z_m \leq Q/2\right].$$

Before we start, let's recall some useful probability propositions. We might indicate that we are using them, placing a reference in a parenthesis next to where the property has been used.

Proposition 17 (Bernoulli's inequality)**.** *For every integer $r \geq 1$ and every real number $x > -1$ we have $(1+x)^r \geq 1 + rx$.*

Corollary 18. *Let $A \sim Geo(p)$, then $\Pr[A \leq k] \leq kp$.*

Proof. That is the case because $\Pr[A \leq k] = 1-(1-p)^k \overset{(Prop.\ 17)}{\leq} 1-(1-kp) = kp$. □

Proposition 19 (Chernoff bound for the binomial distribution)**.** *Let $A \sim B(n,p)$, then the probability $\Pr[A \geq k]$ is smaller than*

$$\exp\left(-n\left(\frac{n-k}{n}\ln\left(\frac{n-k}{n(1-p)}\right) + \frac{k}{n}\ln\left(\frac{k}{np}\right)\right)\right).$$

Proof. Theorem 1 from [2]. □

We can use these tools to bound the attack success probability. Observe that the adversary can choose m_1, m_2 as long as $m_1 + m_2 = \delta$ to obtain different strategies. However, if m_1 is too large, obtaining enough challenges in Γ_1 might be too difficult, and they would not be able to continue to the second round except with too small probability.

Case $\Pr[X \geq m_1] \leq 2^{-(2\lambda-1)}$.

If m_1 is so large that $\Pr[X \geq m_1] \leq 2^{-(2\lambda-1)}$ then

$$\begin{aligned}\epsilon \leq \Pr\left[Y \leq Q/2\right] &\overset{(Cor.\ 18)}{\leq} Q/2 \Pr[X \geq m_1] \\ &\leq Q2^{-2\lambda} \overset{(Q<2^\lambda)}{\leq} 2^{-\lambda}.\end{aligned}$$

We can continue studying different cases.

Case $2^{-(2\lambda-1)} < \Pr[X \geq m_1] \leq 2^{-(\lambda-1)}$.

If m_1 is not that large but still implies that $2^{-(2\lambda-1)} < \Pr[X \geq m_1] \leq 2^{-(\lambda-1)}$ we can nevertheless use Corollary 18 with $\Pr\left[Y \leq Q/2\right]$ to get a non-trivial bound $\Pr\left[Y \leq Q/2\right] \leq Q/2 \Pr[X \geq m_1]$, but we still have to consider what happens in the second round.

Observe that the success probability of the second round is $P_s(m)$

We can guarantee the first probability is small if m is large enough, and we can only guarantee the second is small if m is small enough. For that matter, we have to split the summation. On the one hand, we can use Corollary 18 to bound

$$\Pr\left[Y \leq Q/2\right] \leq Q/2 \Pr[X \geq m_1]$$

and also

$$\Pr\left[Z_m \leq \frac{Q}{2}\right] \leq \frac{Q}{2} \Pr\left[W_m = \delta - m\right] \leq 2^{-(\delta-m+1)}Q.$$

Using these bounds and the fact that $2^{-(\delta-m)} \leq 2^{-2\lambda}$, we find that we can bound

$$\Pr\left[Y \leq Q/2\right] \sum_{m=m_1}^{\delta-2\lambda} P_s(m)$$

by

$$\begin{aligned}
&\leq Q^2 2^{-2(\lambda+1)} \Pr[X \geq m_1] \sum_{m=m_1}^{\delta-2\lambda} \Pr\left[X = m \;\middle|\; X \geq m_1\right] \\
&= Q^2 2^{-2(\lambda+1)} \Pr[X \geq m_1] \Pr\left[X \leq \delta - 2\lambda \;\middle|\; X \geq m_1\right] \\
&\leq Q^2 2^{-2(\lambda+1)} \Pr[X \geq m_1] \\
&\leq Q^2 2^{-2(\lambda+1)} 2^{-(\lambda-1)} \\
&< 2^{-(\lambda+1)}. && (Q < 2^\lambda)
\end{aligned}$$

Using a similar argument, we can bound

$$\Pr\left[Y \leq Q/2\right] \sum_{m=\delta-2\lambda+1}^{\delta} P_s(m)$$

by

$$\begin{aligned}
&\leq Q/2 \Pr[X \geq m_1] \frac{\Pr\left[X \geq \delta - 2\lambda + 1\right]}{\Pr\left[X \geq m_1\right]} \\
&< 2^{\lambda-1} \Pr\left[X \geq \delta - 2\lambda + 1\right] && (Q < 2^\lambda) \\
&\leq 2^{-(\lambda+1)} && (\text{provided } \delta \text{ is large enough})
\end{aligned}$$

What we need is to choose δ so that $\Pr\left[X \geq \delta - 2\lambda + 1\right] \leq 2^{-2\lambda}$. It is always possible if δ is large enough, as it represents the probability of succeeding in all but a constant number of experiments. We would latter find a sufficient condition using Proposition 17.

Notice we are splitting the summation considering $m_1 \leq \delta - 2\lambda$. If that is not the case, the first would be an empty sum, and we could even remove some terms from the second sum. In any case, it is clear that, provided δ satisfies the requirement, $\epsilon < 2^{-(\lambda+1)} + 2^{-(\lambda+1)} = 2^{-\lambda}$.

We can finally analyze the case where we cannot take advantage of the adversary aborting before moving to attack the second round.

Case $2^{-(\lambda-1)} < \Pr[X \geq m_1]$.

We just have to follow a similar approach, splitting again the summation. We start bounding the first part.

$$
\begin{aligned}
&\Pr\left[Y \leq Q/2\right] \sum_{m=m_1}^{\delta-2\lambda} P_s(m) \leq \\
&\leq \sum_{m=m_1}^{\delta-2\lambda} \Pr\left[X = m \;\middle|\; X \geq m_1\right] \Pr\left[Z_m \leq Q/2\right] \\
&\leq \sum_{m=m_1}^{\delta-2\lambda} \Pr\left[X = m \;\middle|\; X \geq m_1\right] Q 2^{-(\delta-m+1)} \\
&\leq Q 2^{-(2\lambda+1)} \sum_{m=m_1}^{\delta-2\lambda} \Pr\left[X = m \;\middle|\; X \geq m_1\right] && \left(2^{-(\delta-m+1)} \leq 2^{-(2\lambda+1)}\right) \\
&= Q 2^{-(2\lambda+1)} \Pr\left[X \leq \delta - 2\lambda \;\middle|\; X \geq m_1\right] \\
&\leq Q 2^{-(2\lambda+1)} && \left(Q < 2^{\lambda}\right) \\
&< 2^{-(\lambda+1)}
\end{aligned}
$$

And bound again the second part.

$$
\begin{aligned}
&\Pr\left[Y \le Q/2\right] \sum_{m=\delta-2\lambda+1}^{\delta} P_s(m) \le \\
&\quad \le \sum_{m=\delta-2\lambda+1}^{\delta} \Pr\left[X = m \;\middle|\; X \ge m_1\right] \Pr\left[Z_m \le \frac{Q}{2}\right] \\
&\quad \le \sum_{m=\delta-2\lambda+1}^{\delta} \Pr\left[X = m \;\middle|\; X \ge m_1\right] \\
&\quad = \frac{\Pr\left[X \ge \delta - 2\lambda + 1\right]}{\Pr\left[X \ge m_1\right]} \\
&\quad \le \frac{\Pr\left[X \ge \delta - 2\lambda + 1\right]}{2^{-(\lambda-1)}} \\
&\quad \le 2^{-(\lambda+1)} \qquad \text{(provided } \delta \text{ is large enough)}
\end{aligned}
$$

And again the only condition is that we wave to choose δ large enough so that

$$\Pr\left[X \ge \delta - 2\lambda + 1\right] \le 2^{-2\lambda}$$

and therefore

$$\epsilon < 2^{-(\lambda+1)} + 2^{-(\lambda+1)} = 2^{-\lambda}.$$

It only remains to obtain a sufficient condition for $\Pr\left[X \ge \delta - 2\lambda + 1\right] \le 2^{-2\lambda}$ using the Proposition 19. Let

$$A = \frac{2\lambda - 1}{\delta(1 - q^{-1})}$$

and

$$B = \frac{\delta - 2\lambda + 1}{\delta q^{-1}}.$$

Then, we can bound

$$
\begin{aligned}
&\Pr\left[X \ge \delta - 2\lambda + 1\right] \le \\
&\quad \le \exp\left(-\delta\left((1 - q^{-1})A\ln(A) + q^{-1}B\ln(B)\right)\right).
\end{aligned}
$$

It would be sufficient to ensure that this expression can be upper bounded by $2^{-2\lambda}$. Taking (base 2) logarithms (and replacing natural logarithms with binary logarithms) the final condition would be

$$(2\lambda - 1)\log_2\left(\frac{2\lambda - 1}{\delta(1 - q^{-1})}\right) +$$
$$+ (\delta - 2\lambda + 1)\log_2\left(\frac{\delta - 2\lambda + 1}{\delta q^{-1}}\right) \geq 2\lambda.$$

Provided that it is not a difficult computation, we can just try increasing δ until we find the first one that satisfies the condition.

Observe that the success probability of the attack could be slightly improved by tweaking some design decisions (for example, the fact that the Q maximum number of queries is equally split among both rounds), but that would not have any meaningful impact. On the other hand, some assumptions are ostensibly conservative. When considering the hardness of the RLWE problems, we wanted to ensure that known attacks take more than 2^λ operations. Here however we are bounding by 2^λ just the number of oracle calls, not the number of total operations, that would be much greater. By analyzing the expected running time, one could then bound it by 2^λ and get stricter upper bounds for Q.

One can see that an equivalent analysis applies to the linear relation protocol. It is also $(1, 1)$-special unsound, and an adversary can be constructed by repeating the same strategy three times in parallel, just defining $\mathbf{z}_3 = \lambda_1\mathbf{z}_1 + \lambda_2\mathbf{z}_2$. Then the same kind of attack applies with the same success, and the conditions to the parameters also apply to the linear case.

It is more difficult to assess the security of the multiplicative relation protocol. Recall that the $(2, 3, 2)$-special soundness is that of a 7-move protocol. However, although in the interactive version the protocol is equivalent whether the verifier sends α and β sequentially or at the same time that is not the case in the non-interactive version after applying Fiat-Shamir as both challenges come from the same hash in the 5-move version and would require two different hashes if transforming the 7-move version.

It is possible to prove that it is $(1, 1)$-out-of-$(q^2, 2)$-unsound, following a similar strategy than the one used with the linear protocol. Nonetheless, the success probability would be lower, as the first challenge set is now $\mathbb{Z}_q^2$ and guessing pairs of integers is harder than guessing a single integer. However, there is still a partial attack with the same success probability as the one against the opening protocol because a malicious adversary $\mathcal{A}$ knowing valid openings to two commitments $\mathbf{c}_1 = \mathbf{a}m_1 + \mathbf{b}r_1 + \mathbf{e}_1$ and $\mathbf{c}_2 = \mathbf{a}m_2 + \mathbf{b}r_2 + \mathbf{e}_2$ could fake a multiplicative proof with an arbitrary $\mathbf{c}_3 \in R_q^k$ just guessing β. They would choose $\Gamma_1 = \{(\alpha, \widetilde{\beta}) \mid \alpha \in \mathbb{Z}_q\}$ and proceed as an honest prover with regard to $\mathbf{c}_1$ and $\mathbf{c}_2$ and as the adversary against the opening protocol with regard to $\mathbf{c}_3$. To make everything satisfy the checks if the challenge β turns out to be $\widetilde{\beta}$ they just need to choose $\mathbf{z}_3 = \mathbf{a}(\mu_3 + \widetilde{\beta}m_1m_2)$ (with minor additional changes on how

the other commitments are computed if the guess turns out to be wrong).

In the end $(q, 1)$-out-of-$(q^2, 2)$-special unsoundness is by all means equivalent to the $(1, 1)$-out-of-$(q, 2)$-special unsoundness (the analysis on the success probability only depends on $l_1/\left|\mathcal{C}\right|_1$ and $l_2/\left|\mathcal{C}\right|_2$, and these two are equal for both scenarios), and the same condition on the parameters would apply for the multiplicative protocol.

Appendix C Finding parameters

Analyzing the security for concrete parameters, we have established a set of restrictions, eqs. (1) to (9), that ensure that a commitment scheme would have a certain security under established hypothesis.

However, finding a set of parameters that fulfils all these conditions might not be direct. It is even harder to find the *best* set of parameters. On the one hand it depends on what is the final purpose as we might prefer a smaller commitment size, smaller NIZKPoK sizes, faster implementations, lower overhead between the commitment and the message sizes... and we might have additional restrictions, again on the size of certain parameters or in any other characteristic.

One has to consider that decreasing one parameter that reduces the commitment size might reduce the suitable space for other parameters, forcing us to increase them, therefore (possibly) increasing the proof size, or any other kind of trade-offs.

Our strategy consists on finding an order such that when the previous parameters are fixed we can find an optimum value for the next one, where optimum means that, conditioned on the previously fixed parameters, it is the value that minimizes size and number of operations of both the commitments and proofs while still satisfying the relations and, most importantly, it makes the conditions for the remaining parameters as loose as possible (i.e. it does not further restrict the available search space more than it is strictly necessary).

For this particular commitment scheme λ, n and q are design parameters that we can freely select depending on the level of security (λ), message space (n,q), or additional properties that we desire. The power of 2 denoted by d that determines the number of irreducible factors $x^n + 1$ splits into when considered modulo q is directly determined by q as the only d that satisfies eq. (1). In fact, one should first choose d in order to find a suitable q, so we include it in the set of four parameters characterizing a commitment instantiation set of parameters (λ, n, q, d).

In order to ease the explanation, we are going to present the procedure from the last to the first parameter. If we had already fixed $(\lambda, n, q, d, \delta_{OL}, \delta_M, B, k)$ satisfying eqs. (1) to (3) and (6) to (9) then σ would be upper bounded by the condition from eq. (5). This bound for σ would be our best candidate.

The only other remaining condition is eq. (4), and we use Albrecht *et al.*'s Lattice Estimator as a black box to compute the hardness of the underlying RLWE problem. Albeit it is only reasonable to assume that greater errors would yield to a harder problem, that is, bitsec should be increasing in σ. Then there are two options, either the σ that comes from the bound derived from eq. (5) satisfies eq. (4), and we choose it, or it does not and we can conclude that no σ exists so that we can add it to the previous (λ, n, q, d, B, k) to get a secure set of parameters. Let $pp = (n, q, \sigma, k)$, and abbreviate $\mathsf{bitsec}(pp)$ by $\mathsf{bsec}(pp)$. We define

$$\mathsf{best}_\sigma(\lambda, n, q, d, B, k) \coloneqq$$
$$= \begin{cases} \mathsf{FromB}_\sigma(\mathsf{vecPrTo}(\lambda, kn), B-1) & \text{if } \mathsf{bsec}(pp) \geq \lambda \\ \perp & \text{otherwise} \end{cases}$$

where $\mathsf{FromB}_\sigma(a, B)$ outputs the greater σ so that we can still ensure a sample from D_σ would be strictly greater than B with a probability lower or equal than 2^{-a},

$$\mathsf{FromB}_\sigma(a, B) \coloneqq B\sqrt{\frac{\log_2(e)}{2(a+1)}},$$

and $\mathsf{vecPrTo}(b, l)$ outputs the smaller a so that we can ensure that if a sample from a distribution holds a condition except with some probability smaller than 2^{-a} then all l independent samples from the same distribution will hold the same condition except with a probability smaller than 2^{-b}, $\mathsf{vecPrTo}(b, l) \coloneqq b + \log_2(l)$, so that the final result is, if possible, the greater σ satisfying eq. (5).

In a completely analogous way to Propositions 2 and 3 about $\mathsf{boundedPr}$ and $\mathsf{vecBoundedPr}$ these two formulas come again from Lemma 4.4 in [27] and the triangular inequality of the Statistical Distance (as the Statistical Distance between a Gaussian and a bounded Gaussian is the probability of the tails).

Something similar happens with k if we assume $(\lambda, n, q, \delta_{OL}, \delta_M, d, B)$ fixed satisfying eqs. (1), (3) and (6) to (9). The condition defined by eq. (2) establishes a lower bound, and it is the best candidate because a smaller k implies smaller commitments and proofs and a larger k would reduce the available search space for σ satisfying the remaining conditions defined by eqs. (4) and (5) (now it is clear the hardness is decreasing in k, and the upper bound on σ is also decreasing in k). Then we could safely choose:

$$\mathsf{best}_K(\lambda, n, q, d, B) \coloneqq \left\lceil \frac{\lambda + 2n\log_2(q)}{n\left(\log_2(q)/d - \log_2(4B-1)\right)} \right\rceil$$

Regarding B, fixing $(\lambda, n, q, d, \delta_{OL}, \delta_M)$, the size of the proofs would be smaller the smaller its size. With regard to k a larger B would make the lower bound for k derived from eq. (2) greater, effectively reducing the available search space. However, the condition imposed by eq. (5) is stricter with smaller B, therefore if B is too small it might be the case that, even with the less restrictive k defined by best_K there is no large enough σ satisfying the remaining conditions eqs. (4) and (5) that make the underlying RLWE problem sufficiently hard. For that matter, in this case we cannot explicitate a formula for B. We can implicitly define it as:

$$\begin{aligned}&\mathsf{best}_B(\lambda, n, q, d) \coloneqq \\ &= \min\left\{2^e \middle| e \in \mathbb{Z}_{>0},\, \exists k, \sigma \text{ s.t. eqs. (2) to (5) hold}\right\}.\end{aligned}$$

We will set $\mathsf{best}_B(\lambda, n, q, d) = \perp$ when the best B cannot be defined.

In order to compute it, we just need to test increasing powers of 2 verifying if with $k_e \coloneqq \mathsf{best}_K(\lambda, n, q, d, 2^e)$ we have $\sigma_e \coloneqq \mathsf{best}_\sigma(\lambda, n, q, d, 2^e, k_e)$ different from $\perp$. Condition (3) imposes an upper bound to B, and we can use it to stop the search because if we reach it then we know that no suitable parameter set exists with such $(\lambda, n, q, d, \delta_{OL}, \delta_M)$.

Once we have fixed a specific tuple of (λ, n, q, d) we notice that, provided δ_{OL} and δ_M only appear in eqs. (6) to (9), respectively, and these equations only involve λ and q, choosing them would not restrict the search space for the rest of parameters. Regarding δ_{OL} and δ_M specifically, as they denote the number of repetitions to achieve soundness, the minimum value that satisfies eqs. (6), (7) and (9) will produce the shorter proofs. Then we can directly choose:

$$\mathsf{initCand}_{\delta_{OL}}(\lambda, q) \coloneqq \left\lceil \frac{\lambda}{\log_2(2q) - \log_2(q+1)} \right\rceil$$

$$\mathsf{initCand}_{\delta_M}(\lambda, q) \coloneqq \left\lceil \frac{\lambda}{\log_2(2q^2) - \log_2(q^2+3q-2)} \right\rceil$$

and keep increasing them until they satisfy eq. (9).

The size of the seeds used to compute the challenges can be directly computed as $8\lceil(\lambda + \delta\log_2(q))/8\rceil$ to satisfy eq. (8).

Notice these two parameters are not entangled with the others and could be chosen at any other moment, but for the rest we have a clear order defined by $(\lambda, n, q, d) \to B \to k \to \sigma$.

Appendix D Additional results

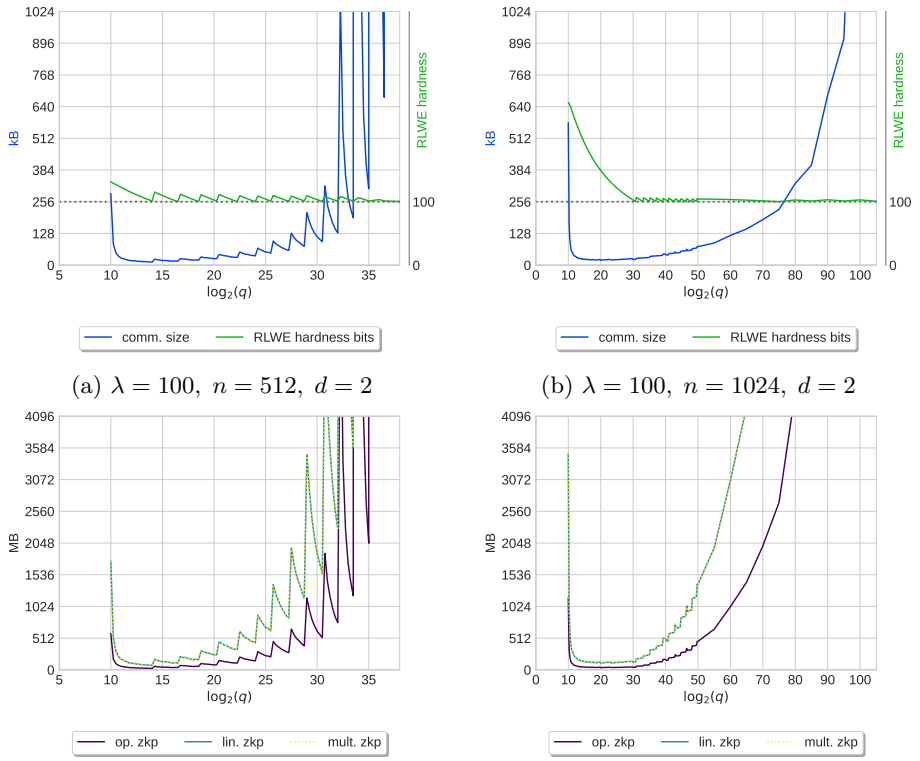
We have chosen to only present in the main article results from a selected subset of parameters. The mere existence of *worse* sets of parameters has no special theoretical interest, neither practical consequences. However, we can still extract some insights that are relevant enough to devote this last appendix to these additional parameters.

As we have already said, in order to get meaningful plots, we have only benchmarked the algorithms with sets that produce multiplicative NIZKPoK of at most 512 MB, because after some point the sizes blow up. Nevertheless, we have also computed secure sets of parameters for larger q's. Given that the sizes only depend on $\lceil q \rceil$ for these additional sets of parameters we have only tried one $2^b < q < 2^{b+1}$ for each $b \in \mathbb{N}$ (notice some restrictions are related to q itself and not its bit-size, so this is again not an exhaustive search).

We observe that, for a fixed n, we can find secure sets of parameters for many moduli q, but only until some size, from which no secure set of parameters satisfying the desired constraints seems to exist. Provided that we compute the hardness of the problem using the Lattice Estimator as a black box, we can only explain it pointing out that the $\log_2$ of the optimum B seems to grow faster than linearly in the $\log_2$ of q to preserve the hardness of the underlying problem, up until a point where Equation (3) cannot be honored. This is important because in theoretical proposals it is usual to say that some condition will be fulfilled for sufficiently large modulus, but it can be the case that, when considering all conditions at the same time no secure set of parameters with such large modulus exists satisfying all of them.

We show the sizes defined by all these additional parameters in Figure 5, again truncating the y-axis because some sets are only secure with absurdly large k, which would imply much greater sizes. These additional sets are only interesting to see the general picture of the parameter space, not because any of these are of practical interest.

We have not stopped searching for additional parameters at the first q such that no secure set existed, because we cannot ensure that no more secure sets exist for even larger q's. We have, however, discarded the existence of additional secure sets of parameters that would yield to smaller commitment sizes than the ones presented in Table 2. From Equation (2) we can see that $k \geq 4$ and this allows us to define a lower bound on the sizes of the commitments that we would obtain before explicitly computing them. We have tested different q up to the point when this lower bound (not tight at all) is already greater than the commitments we have obtained in Table 2 and stopped there. No additional secure sets have been found. We have also used a similar approach to discard the existence of better sets of parameters with $n = 256$ (for which no secure set has been found either) or $n > 1024$ (for which secure sets can be found but with



(a) $\lambda = 100,\ n = 512,\ d = 2$

(b) $\lambda = 100,\ n = 1024,\ d = 2$

(c) $\lambda = 100,\ n = 512,\ d = 2$

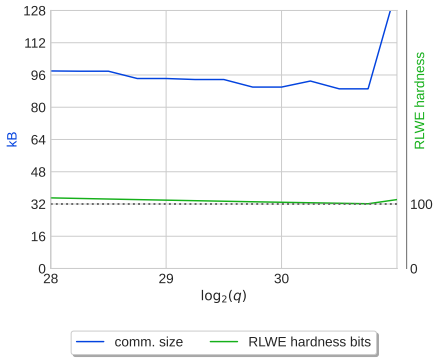
(d) $\lambda = 100,\ n = 1024,\ d = 2$

Figure 5 Additional commitment and NIZKPoK sizes

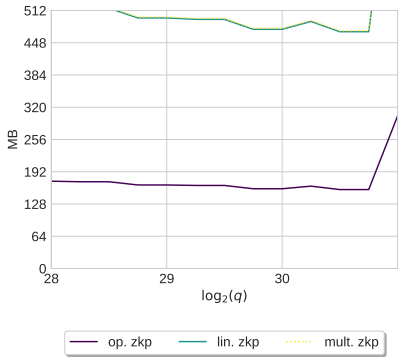
greater commitment sizes).

Besides these additional parameters regarding n and q we also explore here the existing trade-off regarding parameter d. The product of polynomials computation using the partial-FFT multiplication algorithm is more efficient the higher the d, but we can see that secure sets of parameters only exists with $d = 4$ besides the already explored $d = 2$ and the more restrictive conditions imply a greater k that ends up producing not even greater sizes (Figure 6) but also slower times (Figure 7). Therefore, this trade-off is not actually useful because of the restrictions implied by the current design of the binding property.

To ease comparisons with other schemes we present in Table 4 a version of Table 3 detailing the performance of the protocols in millions of processor cycles.

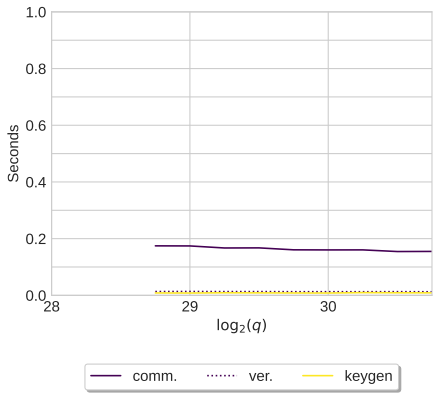


(a) $\lambda = 100,\ n = 1024,\ d = 4$

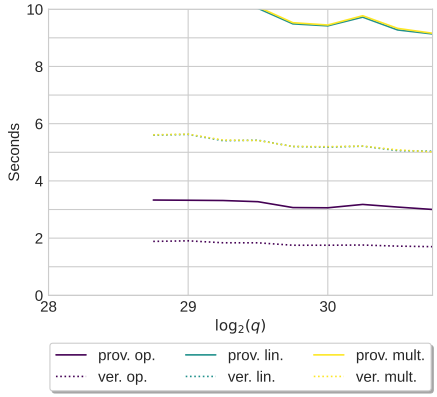


(b) $\lambda = 100,\ n = 1024,\ d = 4$

Figure 6 Commitment and NIZKPoK sizes



(a) $\lambda = 100,\ n = 1024,\ d = 4$



(b) $\lambda = 100,\ n = 1024,\ d = 4$

Figure 7 Commitment and NIZKPoK times

Table 4: Running time of the best parameters (in mill. of processor cycles)

n	q	com.	ver.	key.	$\mathcal{P}_{op}$	$\mathcal{V}_{op}$	$\mathcal{P}_{lin}$	$\mathcal{V}_{lin}$	$\mathcal{P}_{mult}$	$\mathcal{V}_{mult}$
512	16381	147.37	6.96	9.47	2368.29	1212.89	2817.98	3658.69	2580.35	3668.70
1024	1048573	168.67	8.86	19.12	2800.07	1525.85	2930.43	428.74	3988.30	342.63
1024	11863253	148.36	8.22	19.89	2499.80	1327.29	3254.21	3992.30	3094.35	4042.38
1024	16777213	148.51	8.25	19.36	2443.48	1318.44	3000.48	3966.21	2886.83	3980.50
1024	67108837	148.04	8.47	19.10	2587.04	1474.27	3564.68	423.43	3347.62	247.97
1024	1073741789	131.37	10.55	25.29	2386.08	1393.49	2868.73	3982.53	2734.25	4098.82
1024	1276901389	132.52	11.38	27.13	2492.12	1386.63	3199.94	3852.37	3102.74	3286.20
1024	1518500213	132.37	11.36	27.24	2457.95	1386.97	3053.65	3870.64	3002.06	3243.41
1024	1805811253	133.07	11.37	26.91	2439.11	1384.01	3016.04	3868.50	2903.86	3374.66