Challenges for Ring-LWE

Eric Crockett[*] Chris Peikert[†]

February 21, 2017

Abstract

As lattice cryptography becomes more widely used in practice, there is an increasing need for further cryptanalytic effort and higher-confidence security estimates for its underlying computational problems. Of particular interest is a class of problems used in many recent implementations, namely, Learning With Errors (LWE), its more efficient ring-based variant Ring-LWE, and their "deterministic error" counterparts Learning With Rounding (LWR) and Ring-LWR.

To facilitate such analysis, in this work we give a broad collection of challenges for concrete Ring-LWE and Ring-LWR instantiations over cyclotomics rings. The challenges cover a wide variety of instantiations, involving two-power and non-two-power cyclotomics; moduli of various sizes and arithmetic forms; small and large numbers of samples; and error distributions satisfying the bounds from worst-case hardness theorems related to ideal lattices, along with narrower errors that still appear to yield hard instantiations. We estimate the hardness of each challenge by giving the approximate Hermite factor and BKZ block size needed to solve it via lattice-reduction attacks.

A central issue in the creation of challenges for LWE-like problems is that dishonestly generated instances can be much harder to solve than properly generated ones, or even impossible. To address this, we devise and implement a simple, non-interactive, publicly verifiable protocol which gives reasonably convincing evidence that the challenges are properly distributed, or at least not much harder than claimed.

List of Corrections

[*]Georgia Institute of Technology and University of Michigan.

[†]Computer Science and Engineering, University of Michigan. This material is based upon work supported by the National Science Foundation under CAREER Award CCF-1054495 and CNS-1606362 and by a Google Research Award. The views expressed are those of the authors and do not necessarily reflect the official policy or position of the National Science Foundation or the Sloan Foundation.

1 Introduction

Lattice-based cryptosystems are some of the leading "post-quantum" candidates that are plausibly secure against potential large-scale quantum computers. As lattice cryptography begins a transition to widespread deployment (see, e.g., [Ste14, LS16, Bra16]), there is a pressing need for increased cryptanalytic effort and higher-confidence hardness estimates for its underlying computational problems. Of particular interest is a class of problems used in many recent implementations (e.g., [HS, GLP12, DDLL13, BCNS15, ADPS16, CP16a, BCD$^+$16]), namely:

- Learning With Errors (LWE) [Reg05],
- its more efficient ring-based variant Ring-LWE [LPR10], and
- their "deterministic error" counterparts Learning With Rounding (LWR) and Ring-LWR [BPR12].

Informally, the *search* version of the Ring-LWE problem is to find a secret ring element s given multiple random "noisy ring products" with s, while the *decision* version is to distinguish such noisy products from uniformly random ring elements. More precisely, Ring-LWE is actually a *family* of problems, with a concrete *instantiation* given by the following parameters:[1]

1. a *ring* R, which can often (but not always) be represented as a polynomial quotient ring $R = \mathbb{Z}[X]/(f(X))$ for some irreducible $f(X)$, e.g., $f(X) = X^{2^k} + 1$ or another cyclotomic polynomial;
2. a positive integer *modulus* q defining the quotient ring $R_q := R/qR = \mathbb{Z}_q[X]/(f(X))$;
3. an *error distribution* χ over R, which is typically concentrated on "short" elements (for an appropriate meaning of "short");
4. a *number of samples* provided to the attacker.

The Ring-LWE search problem is to find a uniformly random secret $s \in R_q$, given independent samples of the form

$$(a_i\, ,\, b_i = s \cdot a_i + e_i) \in R_q \times R_q,$$

where each $a_i \in R_q$ is uniformly random and each $e_i \leftarrow \chi$ is drawn from the error distribution. The decision problem is to distinguish samples of the above form from uniformly random samples over $R_q \times R_q$.

Ring-LWR is a "derandomized" variant of Ring-LWE in which the random errors are replaced by deterministic "rounding" to a smaller modulus $p < q$. Specifically, the search problem is to find a random secret $s \in R_q$ given independent samples

$$(a_i\, ,\, b_i = \lfloor s \cdot a_i \rceil_p) \in R_q \times R_p,$$

where each $a_i \in R_q$ is uniformly random, and $\lfloor \cdot \rceil_p \colon R_q \to R_p$ denotes the function that rounds each coefficient $c_j \in \mathbb{Z}_q$ of the input (with respect to an appropriate basis) to $\lfloor \frac{p}{q} \cdot c_j \rceil \in \mathbb{Z}_p$. The decision problem is to distinguish such samples from $(a_i, \lfloor u_i \rceil_p)$, where $a_i, u_i \in R_q$ are uniformly random and independent. (Notice that $\lfloor u_i \rceil_p \in R_p$ itself is uniformly random when p divides q, but otherwise is biased.)

[1] This description is of a syntactically "tweaked" form of Ring-LWE, which for convenience avoids a special ideal denoted $R^\vee$. This form is equivalent to the original "untweaked" form under a suitable change to the error distribution; see Section 2.3 for details.

Hardness. A main attraction of Ring-LWE (and Ring-LWR) is their *worst-case hardness* theorems, also known as *worst-case to average-case reductions*. Essentially, these say that solving certain instantiations is at least as hard as quantumly solving a corresponding approximate Shortest Vector Problem (approx-SVP) on *any* "ideal lattice," i.e., a lattice corresponding to an ideal of the ring. (Interestingly, the converse is unclear: it is unknown how to solve Ring-LWE using an oracle for even exact-SVP on any ideal lattice of the ring.) See [LPR10, PRS17] and [BPR12] for precise theorem statements, Section 1.2 below for further discussion, and [CDPR16, CDW17] for the status of approx-SVP on ideal lattices for quantum algorithms.[2]

As long as the underlying approx-SVP problem is actually hard in the worst case, the above-described theorems give strong evidence of cryptographic hardness, at least asymptotically (i.e., for large enough n). For practical purposes, though, the following property of (Ring-)LWE and related problems has been noticed, studied, and exploited for many years (see, e.g., [LMPR08, MR09, Lyu09, LP11, BBL$^+$14, HKM15]): even instantiations that are *not* supported by known worst-case hardness theorems, or that have too-small dimensions n to draw any meaningful conclusions from them, *can still appear very hard*—as measured against all known classes of attack. Indeed, almost every implementation of lattice cryptography to date has used considerably smaller dimensions and errors than what worst-case hardness theorems alone would recommend. However, care is needed in following this approach: e.g., some instantiations involving especially small errors turn out to be broken or seriously weakened by various attacks (see, e.g., [AG11, CLS15, Pei16]).

Given this state of affairs, and especially the common usage in practice of parameters that lack much (if any) theoretical support, we believe that a deeper understanding of how the different aspects of Ring-LWE affect concrete hardness is a critically important direction of research.

1.1 Contributions

This work provides a broad collection of cryptanalytic challenges for concrete instantiations of the search-Ring-LWE/LWR problems over *cyclotomic* rings, which are the most widely used and studied class of rings in this context. Our challenges cover a wide range and variety of parameterizations and conjectured security levels, ranging from "toy" to "very hard" (see Section 1.2 for details). We hope that these challenges will provide a focal point for theoretical and practical cryptanalytic effort on Ring-LWE/LWR, and will help to more precisely quantify the concrete security of their instantiations.[3]

A central issue in the creation of challenges for problems like (Ring-)LWE is that a dishonest challenger can publish instances that are much harder to solve than honestly generated ones—or even impossible. This is because (properly instantiated) Ring-LWE is conjectured to be pseudorandom, so it is difficult to distinguish between a correctly generated challenge and a harder one with much larger errors, or even a uniformly

[2] In brief: the fastest known quantum algorithms for the $\mathrm{poly}(n)$-approx-SVP problems underlying many cryptographic constructions, in any class of rings covered by the hardness theorems, perform essentially no better than algorithms for arbitrary lattices of the same dimension n, and take at least exponential $2^{\Omega(n)}$ time. Under plausible number-theoretic conjectures, $2^{O(\sqrt{n \log n})}$-approx-SVP is solvable in quantum polynomial time in certain rings, such as prime-power cyclotomics and their maximal totally real subrings [CDPR16, CDW17]; however, the main algorithmic technique used in these works meets a barrier at $2^{\Omega(\sqrt{n}/\log n)}$-factor approximations [CDPR16, Section 6].

[3] The challenges and their parameters can be obtained via the Ring-LWE challenges website [RLW16]. The archive `rlwe-challenges-v1.tar.gz` contains challenges for 516 different instantiations, and has a SHA-256 hash value `07cd f744 5c9d 178c 8b13 5a42 47ca a143 5320 c104 8ee8 c634 8914 a915 5757 dcef`. All our challenge-related archives are digitally signed under the PGP/GPG public key having ID `b8b2 45f5`, which has fingerprint `8126 1e02 fc1a 11c9 631a 65be b5b3 1682 b8b2 45f5`.

random one, which has no solution. A dishonest challenger could therefore publish unsolvable challenges, and point to the absence of breaks as bogus evidence of hardness.[4]

To deal with this issue, we design and implement a simple, non-interactive, and publicly verifiable "cut-and-choose" protocol that gives reasonably convincing evidence that the challenge instances are properly distributed, or at least not much harder than claimed. In short, for each Ring-LWE/LWR instantiation the challenger announces many timestamped instances. At a later time, the challenger reveals the secrets for all but a *random one* of the instances, as determined by a publicly verifiable source of randomness. (Concretely, we use the NIST randomness beacon [NIS11].) Anyone can then verify that all the revealed instances look "proper," which makes it likely that the remaining instance is proper as well—otherwise, the challenger would have had been caught with rather larger probability (as long as it cannot predict or influence the randomness source). See Section 3 for further details and discussion of potential alternatives, such as zero-knowledge proofs for lattice problems, which turn out *not* to give the kind of guarantees we desire.

Search versus decision. We stress that our challenges are for *search* versions of Ring-LWE/LWR, whereas many cryptographic applications rely on the conjectured hardness of solving *decision* with noticeable advantage. Unfortunately, it appears impractical to give meaningful challenges for the latter regime. This is because detecting a tiny advantage requires a very large number of instances, and a corresponding increase in effort by the attacker. And even for relatively large advantages, the naïve method of confirming the solutions would require the challenger to retain the correct answers and honestly compare them to the attacker's, because the attacker cannot confirm its own answers (unlike with the search problem, where it can).[5]

Nevertheless, we gain confidence in the usefulness of search challenges from the fact that the known classes of attack against decision either proceed by directly solving search, or can be adapted to do so with relatively little or no extra overhead. (See [LP11, LN13, ADPS16].) In addition, there are search-to-decision reductions [LPR10, Section 5] which provide evidence that decision cannot be much easier than search (though the known reductions incur some as-yet unoptimized overhead). Finally, we note that practical constructions of, e.g., key exchange as in [BCD$^+$16] can use "hashed" variants, for which hardness of search can be sufficient for a reductionist security analysis in the random oracle model.

Implementation. Our free and open-source challenge generator and verifier are implemented using the recent $\Lambda \circ \lambda$ (pronounced "L O L") framework for lattice- and ring-based cryptography [CP16a, CP16b]. In particular, $\Lambda \circ \lambda$ supports arbitrary cyclotomics and sampling from the theory-recommended Ring-LWE distributions we use in our instantiations (see Section 1.2 for details). We stress that while $\Lambda \circ \lambda$ is written in the functional, strongly typed language Haskell, all the challenge data is serialized using Google's platform- and language-neutral *protocol buffers* (protobuf) framework [Goo08]. This allows the challenges to be read using most popular programming languages, via parsers that are automatically generated from our protobuf message specifications. (These specifications are given in Appendix C, and with the challenges themselves.) In addition, $\Lambda \circ \lambda$ includes C++ code for cyclotomic ring operations, which can be used by alternative implementations written in other languages.

[4]This appears qualitatively different from problems like integer factorization and discrete logarithms, where deviating from the prescribed distributions seems like it can only make challenges *easier* to solve, or at least no harder.

[5]We considered more sophisticated non-interactive methods for confirming answers, like using a "fuzzy extractor" [DORS04] to encrypt a secret that can only be recovered by solving a large enough fraction of decision challenges. Such methods seem tantalizing, but are complex to implement and bandwidth-intensive in our setting, so we leave this direction to future work.

1.2 Challenge Instantiations

Our challenge instantiations cover a wide range of parameters for several aspects of the Ring-LWE/LWR problems, including: size and form of the cyclotomic *index* and corresponding dimension; *width* of the error distribution; size and arithmetic form of the *modulus*; and number of *samples*. Each of these parameters has some degree of influence on the conjectured hardness of a Ring-LWE instantiation, as we discuss below.

For each challenge instantiation we give a qualitative hardness estimate, ranging from "toy" and "easy" to "very hard," along with an approximate block size that should allow the Block Korkin-Zolotarev (BKZ) basis-reduction algorithm to solve the instantiation. The easier categories represent instantiations that should be breakable using standard lattice algorithms on desktop-class machines in somewhere between a few minutes and a few months, whereas the hardest category should be out of reach even for nation-state adversaries—based on the current state of public cryptanalysis, at least. We deduce our hardness estimates by approximating the Hermite factors and BKZ block sizes needed to solve the instantiations via lattice attacks, which usually represent the most practically efficient attacks against Ring-LWE/LWR. See Section 5 for details.

1.2.1 Cyclotomic Index

A primary parameter influencing Ring-LWE's conjectured hardness is the *degree* (or dimension) of the ring R, which in the cyclotomic case is the totient $n = \varphi(m)$ of the *index* (or conductor) m. Thus far, most implementations have used *two-power* cyclotomic rings, because they have the computationally and analytically simplest form $R \cong \mathbb{Z}[X]/(X^n + 1)$, where n is a power of two. Moreover, sampling from a spherical Gaussian in their "canonical" geometry is equivalent to sampling independent identically distributed Gaussian coefficients for the powers of X.

Nevertheless, we believe that Ring-LWE over non-two-power cyclotomics is deserving of more cryptanalytic effort. First, powers of two are rather sparse, especially in the relevant range of n in the several hundreds or more. In addition, two-power cyclotomics are incompatible with some advanced features of fully homomorphic encryption (FHE) schemes, such as "plaintext packing" [SV11] and asymptotically efficient "bootstrapping" algorithms [GHS12, AP13] for characteristic-two plaintext rings like $\mathbb{F}_{2^k}$. Finally, non-two-power cyclotomic rings lack orthogonal bases (in the canonical geometry), so sampling from recommended error distributions and error management are more subtle [LPR13], and it is interesting to consider what effect (if any) this has on concrete hardness.

Our challenges are weighted toward the popular two-power case, but they also include indices of a variety of other forms, including powers of other small primes, those that are divisible by many small primes, and moderately large primes. We are particularly interested in whether there are any cryptanalytic attacks that can take special advantage of any of these forms. Our choices of indices m correspond to dimensions n ranging from 128 to $4{,}096$ for Ring-LWE, and from 16 to 162 for Ring-LWR.

1.2.2 Error Width

The *absolute* error of a (Ring-)LWE instantiation is, very informally, the "width" of the coefficients of the error distribution, with respect to an appropriate choice of basis. The main worst-case hardness theorems for (Ring-)LWE (e.g., [Reg05, Pei09, LPR10]) apply to Gaussian-like error distributions whose widths exceed certain $\Omega(\sqrt{n})$ bounds. Conversely, there are algebraic attacks that can exploit significantly narrower errors, if enough samples are available (see, e.g., [AG11, ACFP14, EHL14, CLS15, CLS16, Pei16]). However, there is still a poorly understood gap between the theoretical bounds and parameters that plausibly fall to such attacks, especially in the low-sample regime (see Section 1.2.4 below for further details).

Following the original definition and recommended usage of Ring-LWE [LPR10, LPR13], our challenge instantiations use *spherical Gaussian* error (in the canonical geometry), relative to the "dual" fractional ideal $R^\vee$ of the ring R. More specifically, the products $s \cdot a_i$ reside in the quotient group $R^\vee/qR^\vee$, and we add Gaussian error D_r of some parameter $r > 0$. We emphasize that $R^\vee$ corresponds to a much denser lattice than $\mathbb{Z}^n$; in particular, D_r yields errors having (not necessarily independent) Gaussian coefficients of width $r\sqrt{n}$ with respect to the "decoding" basis of $R^\vee$. Therefore, our setting is closely analogous to plain LWE with Gaussian error of parameter $r\sqrt{n}$.

Our challenge instantiations use four qualitative categories of error parameter r:

Trenta corresponds to a bound from the main "worst-case hardness of decision-Ring-LWE" theorem [LPR10, Theorem 3.6], namely, $r \geq (n\ell/\ln(n\ell))^{1/4} \cdot \sqrt{\ln(2n/\varepsilon)/\pi}$, where ℓ is the number of revealed samples and (say) $\varepsilon \approx 2^{-80}$ is a bound on the statistical distance in the reduction.[6] We pose this class of challenges to give some insight into instantiations that conform to the error bounds from known worst-case hardness theorems (though not necessarily for large enough dimensions n to obtain meaningful hardness guarantees via the reductions alone).

Grande corresponds to some $r \geq c = \Theta(1)$ (i.e., coefficients of width $c\sqrt{n}$) that satisfies the lower bound from Regev's worst-case hardness theorem [Reg05] for *plain* LWE, and that also suffices for provable immunity to the class of "ring homomorphism" attacks defined in [EHL14, ELOS15, CLS15, CLS16], as shown in [Pei16, Section 5]. We note that while the theorems from [Reg05] and [Pei16] are stated for $c = 2$, an inspection of the proofs and tighter analysis reveal that the constant can be improved to nearly $1/(2\sqrt{\pi}) \approx 0.282$ in the former case [Reg16], and to $c = \sqrt{8/(\pi e)} \approx 0.968$ or better in the latter case, depending on the dimension and desired time/advantage lower bound (see Section 4.1 for details). We pose this class of challenges to give instantiations which *might* someday conform to significantly improved worst-case hardness theorems for Ring-LWE, and which in any case satisfy the bounds from known hardness theorems in the absence of ring structure.

Tall corresponds to $r \in \{6, 9\}/\sqrt{n}$, i.e., error coefficients of width 6 or 9. Errors of roughly this size have been used in prior concrete analyses of LWE instantiations (e.g., [MR09, LP11]) and in practical implementations of (Ring-)LWE cryptography (e.g., [ADPS16, BCD$^+$16]).

Short corresponds to $r \in \{1, 2\}/\sqrt{n}$, i.e., error coefficients of width 1 or 2. In light of the above-mentioned small-error and homomorphism attacks, we consider such parameters to be riskier, at least when a large number of Ring-LWE samples are available. But at present it is unclear whether the attacks are feasible when only a small or moderate number of samples are available, as is the case in our challenges and in many applications (see Section 1.2.4 below for further discussion).

Finally, for each setting of the error parameter we give challenges for both *continuous* error and its corresponding *discretized* version, where each real coefficient (with respect to the decoding basis) is rounded off to the nearest integer. Cryptographic applications almost always use discrete forms of Ring-LWE, but continuous forms are also cryptanalytically interesting. In particular, rounding yields a tight reduction from any continuous form to its corresponding discrete form, i.e., the latter is at least as hard as the former.

[6] It is very likely that the bound can be improved by a small constant factor within the same proof framework; in addition, the $(n\ell/\ln(n\ell))^{1/4}$ factor might be an artifact of the proof. However, we use the bound as stated for our challenges.

1.2.3 Modulus

Another main quantity that strongly influences Ring-LWE's apparent hardness is the *error rate*, which is, informally, the ratio of the (absolute) error width to the modulus q. There is much theoretical and practical cryptanalytic evidence that, all else being equal, Ring-LWE becomes harder as the error rate increases. E.g., there are tight reductions from smaller to larger rates; worst-case hardness theorems yield stronger conclusions for larger error rates; and lattice-based attacks perform worse in practice. Therefore, cryptographic applications typically aim to use the smallest possible modulus that can accomodate the accumulated error terms without mod-q "wraparound" (so as to avoid, e.g., incorrect decryption). However, other considerations can introduce additional subtleties in the choice of modulus.

The initial worst-case hardness theorem for *search*-Ring-LWE [LPR10, Theorem 4.1] applies to any sufficiently large modulus q and absolute error. However, the search-to-decision reduction [LPR10, Theorems 5.1 and 5.2] requires q to be a prime integer that "splits well" in R, i.e., the ideal qR factors into distinct prime ideals of small norm.[7] Subsequent work [BV11, BLP$^+$13] used the "modulus switching" technique to obtain a reduction for essentially any modulus, at the cost of an increase in the error rate. Finally, recent work [PRS17] gave a worst-case hardness theorem for *decision*-Ring-LWE for *any* modulus, which either matches or improves upon the just-described results in terms of parameters. On the cryptanalytic side, the above-mentioned homomorphism attacks of [EHL14, ELOS15, CLS15, CLS16] can take advantage of moduli q for which the ideal qR has small-norm ideal divisors, but only when the error is insufficiently "well spread" relative to those ideals. (See [Pei16] for further details.)

With these considerations in mind, our challenge instantiations include moduli of a variety of sizes and arithmetic forms. We include moduli that split completely, others that split very poorly, and some that "ramify" (e.g., two-power moduli for two-power cyclotomics). Each instantiation uses a modulus that is large enough, relative to the absolute error, to yield correct decryption with high probability in public-key encryption and key-exchange protocols following the template from [LPR10, Pei14]. See Section 4.2 for further details.

1.2.4 Number of Samples

Finally, each of our challenge instantiations consist of either a small or moderate number of samples (specifically, three or 100) for Ring-LWE, and 500 samples for Ring-LWR. These choices are motivated by the following considerations: while simple cryptographic constructions like key exchange and digital signatures reveal only a few samples (per fresh secret) to the adversary, other constructions like FHE, identity/attribute-based encryption, and pseudorandom functions can reveal a much larger (possibly even adversary-determined) number of samples.

Clearly, revealing more samples cannot increase the hardness of an instantiation, because the attacker can just ignore some of them. There is also evidence that in certain parameter regimes, such as small bounded errors, increasing the number of samples can significantly reduce concrete hardness [AG11, ACFP14]. At the same time, the main worst-case hardness theorems for Ring-LWE place mild or no conditions at all on the number of samples [LPR10, Theorem 3.6], and the same goes for plain LWE [Reg05, Pei09, BLP$^+$13]. (Worst-case hardness theorems for less-standard LWE instantiations [MP13], and for (Ring-)LWR [BPR12, AKPW13, BGM$^+$16, AA16], do have a strong dependence on the number of samples, however.) There are also standard techniques to generate fresh (Ring-)LWE samples from a fixed number of given ones, though at a cost in the error rate of the new samples [Lyu05, GPV08, ACPS09].

[7] Such moduli also enable FFT-like algorithms over $\mathbb{Z}_q$, also called Chinese Remainder Transforms, which yield fast multiplication algorithms for R/qR using just $\mathbb{Z}_q$ operations.

In summary, the practical effect of the number of samples on concrete hardness is unclear, and seems to depend heavily on the other parameters of the instantiation. Therefore, we separately consider both the small- and moderate-sample regime for our challenge instantiations.

1.3 Other Related Work

In a recent concurrent and independent work, Buchmann *et al.* [BBG$^+$16] describe a method and implementation for creating challenges for LWE (but not Ring-LWE). Both their work and ours encounter a common issue—that naïve methods of generating challenges require knowing the solutions—but their main goal is to not exclude anybody from participating in the cryptanalysis of the resulting challenges. They accomplish this by generating the challenges using a multi-party computation protocol, so that the solutions never reside with any single party. (Their implementation uses three parties, although this is not inherent to the approach.) In addition, their protocol allows for retroactively verifying the players' honest behavior *after a challenge has been solved*. However, we observe that if a majority of the parties collude, then they can obtain the solutions "semi-honestly" (i.e., without deviating from the protocol), or even maliciously create invalid instances that have no solutions. In either case, the cheating would not be detectable; in particular, the lack of a solution means that the players would never have to demonstrate honest behavior. By contrast, our protocol gives good evidence that the challenges are properly generated, although the secrets are generated in one place.

Over the years there have been many analyses of various LWE parameterizations, in both the asymptotic and concrete settings, against various kinds of attacks, e.g., [MR09, LP11, AFG13, ACFP14, ACF$^+$15, APS15, HKM15]. All of these apply equally well to Ring-LWE, which can be viewed as a specialized form of LWE, although they do not attempt to exploit the ring structure.

Cryptanalytic challenges have been provided for many other kinds of problems and cryptosystems, including integer factorization [RSA91], discrete logarithm on elliptic curve groups [Cer97], short-vector problems on ad-hoc distributions of ideal lattices [PS13], the NTRU cryptosystem [NTR15], and multivariate cryptosystems [YDH$^+$15].

1.4 Organization

The remainder of the paper is organized as follows:

Section 2 recalls the necessary mathematical background for the Ring-LWE and Ring-LWR problems.

Section 3 describes our non-interactive, publicly verifiable "cut-and-choose" protocol for giving evidence that the challenge instances are properly distributed.

Section 4 gives further details on how we choose our instantiations' parameters, specifically their Gaussian widths and moduli.

Section 5 describes how we obtain approximate hardness estimates for our challenge instantiations.

Appendix A gives some lower-level technical details about our implementation and the operational security measures we used while creating the challenges.

Appendices B and C describe the directory layouts and file formats for the challenges.

Acknowledgments. We thank Oded Regev for helpful discussions, and for initially suggesting the idea of publishing Ring-LWE challenges.

2 Background

We now recall the relevant mathematical background and definitions of the Ring-LWE and Ring-LWR problems; see [LPR10, LPR13, CP16a] for many more mathematical and computational details.

2.1 Lattices and Gaussians

In cyclotomic ring-based lattice cryptography, we use the space $H \subseteq \mathbb{C}^n$ for some even integer n, defined as

$$H := \{\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{C}^n : x_i = \overline{x_{i+n/2}},\ i \in \{1, \ldots, n/2\}\}.$$

It is easy to check that H, with the inner product $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_i x_i \overline{y_i}$ of the ambient space $\mathbb{C}^n$, is an n-dimensional real inner product space, i.e., it is isomorphic to $\mathbb{R}^n$ via an appropriate rotation. Therefore, the reader may mentally replace H with $\mathbb{R}^n$ in all that follows. We let $\mathcal{B} = \{\mathbf{x} \in H : \|\mathbf{x}\| \leq 1\}$ denote the closed unit ball in H (in the Euclidean norm).

For the purposes of this work, a *lattice* $\mathcal{L}$ is discrete additive subgroup of H that is full rank, i.e., $\operatorname{span}_{\mathbb{R}}(\mathcal{L}) = H$. A lattice is generated as the set of integer linear combinations of some linearly independent basis vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_n\}$:

$$\mathcal{L} = \mathcal{L}(\mathbf{B}) := \Big\{ \sum_i z_i \mathbf{b}_i : z_i \in \mathbb{Z} \Big\}.$$

The *volume* (or determinant) of a lattice $\mathcal{L}$ is $\operatorname{vol}(\mathcal{L}) := \operatorname{vol}(H/\mathcal{L}) = |\det(\mathbf{B})|$, where $\mathbf{B}$ denotes any basis of $\mathcal{L}$. The *minimum distance* of $\mathcal{L}$ is $\lambda_1(\mathcal{L}) := \min_{\mathbf{0} \neq \mathbf{v} \in \mathcal{L}} \|\mathbf{v}\|$, the length of a shortest nonzero lattice vector. The *dual lattice* $\mathcal{L}^\vee$ of a lattice $\mathcal{L}$ is the set of all points in H having integer inner products with every vector of the lattice: $\mathcal{L}^\vee := \{\mathbf{w} \in H : \langle \mathbf{w}, \mathcal{L} \rangle \subseteq \mathbb{Z}\}$.

Gaussians. The Gaussian function $\rho \colon H \to \mathbb{R}^+$ is defined as $\rho(\mathbf{x}) := \exp(-\pi \|\mathbf{x}\|^2)$, and is scaled to have *parameter* (or *width*) $r > 0$ by defining $\rho_r(\mathbf{x}) := \rho(\mathbf{x}/r)$. The (spherical) Gaussian probability distribution D_r over H is defined to have probability density function $r^{-n} \cdot \rho_r$. (We usually omit the subscript when $r = 1$.)

The following bounds use the function

$$f(x) = \sqrt{2\pi e} \cdot x \cdot \exp(-\pi x^2), \tag{2.1}$$

which is strictly decreasing and at most 1 for $x \geq 1/\sqrt{2\pi}$.

Lemma 2.1 ([Ban93, Lemma 1.5]). *For any $c > 1/\sqrt{2\pi}$ defining $C = f(c) < 1$, and any lattice $\mathcal{L} \subset H$,*

$$\rho(\mathcal{L} \setminus c\sqrt{n}\mathcal{B}) < C^n \cdot \rho(\mathcal{L}).$$

The analogous continuous bound $D(H \setminus c\sqrt{n}\mathcal{B}) < C^n$ follows by taking an arbitrarily dense lattice $\mathcal{L}$ and using a limiting argument. The following is a result of rearranging terms.

Corollary 2.2. *If $\pi c^2 - \ln c \geq \frac{1}{n}\ln(\frac{1}{\varepsilon}) + \frac{1}{2}\ln(2\pi e)$ for some $c > 1/\sqrt{2\pi}$ and $\varepsilon > 0$, then $D(H \setminus c\sqrt{n}\mathcal{B}) < \varepsilon$.*

The following is an immediate corollary of Lemma 2.1 and [MR04, Lemma 4.1].

Lemma 2.3. *For any lattice $\mathcal{L} \subset H$ and $r > \sqrt{n/2\pi}/\lambda_1(\mathcal{L}^\vee)$ defining $C = f(r\lambda_1(\mathcal{L}^\vee)/\sqrt{n}) < 1$, the statistical distance between $D_r \bmod \mathcal{L}$ and the uniform distribution over $H/\mathcal{L}$ is less than $\frac{1}{2}C^n/(1 - C^n)$.*

2.2 Cyclotomic Rings and Ideal Lattices

Two-power cyclotomics. As a warm-up, we start with the necessary background for *two-power* cyclotomic rings, which have especially simple representations and are widely used in practical applications of Ring-LWE. This background is sufficient to understand the remainder of the paper (from Section 2.3 onward) and our challenges in the specialized case of two-power cyclotomics.

When $m = 2^k \geq 2$ is a power of two, the mth cyclotomic polynomial is $\Phi_m(X) = X^n + 1$, where $n = \varphi(m) = 2^{k-1}$. The mth *cyclotomic field* can be represented as $K = \mathbb{Q}[X]/(X^n + 1)$, and the mth *cyclotomic ring* as $R = \mathbb{Z}[X]/(X^n + 1)$. The *power* basis (which is identical to the *powerful* basis $\vec{p}$ and the *"tweaked" decoding* basis $t \cdot \vec{d}$ of R; see below) consists of the powers $1, X, X^2, \ldots, X^{n-1}$. That is, an element of K (respectively, R) can be uniquely represented as a rational (resp., integral) polynomial in X of degree less than n.

The *canonical embedding* $\sigma\colon K \to H$ can be viewed as a linear transform from the power-basis coefficient vector in $\mathbb{Q}^n$ to $H \subset \mathbb{C}^n$, where H is as defined above in Section 2.1. Under this view, σ is just a scaling by a $\sqrt{n}$ factor, followed by a rigid rotation (an isometry). Therefore, the Gaussian distribution D_r over H (and over K, via σ^{-1}) corresponds to independent power-basis coefficients, each drawn from $D_{r/\sqrt{n}}$.

The fractional *codifferent* ideal of R is $R^\vee = n^{-1}R$. The *decoding* basis $\vec{d}$ of $R^\vee$ turns out to be the powerful basis, scaled down by the *"tweak"* factor $t = n$, i.e., $t \cdot \vec{d} = \vec{p}$. Therefore, the Gaussian distribution D_r over K corresponds to independent decoding-basis coefficients, each drawn from $D_{r\sqrt{n}}$. Tautologically, the same goes for the power-basis coefficients for the "tweaked" distribution $t \cdot D_r = D_{rn}$.

General cyclotomics. For a positive integer m, the mth *cyclotomic number field* is $K = \mathbb{Q}(\zeta_m)$, the field extension of the rationals $\mathbb{Q}$ obtained by adjoining an element ζ_m having multiplicative order m, i.e., a primitive mth root of unity. The ring of algebraic integers in K is $R = \mathbb{Z}[\zeta_m]$, the mth *cyclotomic ring*. The minimal polynomial of ζ_m has degree $n = \varphi(m)$, so $\deg(K/\mathbb{Q}) = \deg(R/\mathbb{Z}) = n$.

There are n distinct ring embeddings (i.e., injective ring homomorphisms) $\sigma_i\colon K \to \mathbb{C}$, indexed by $i \in \mathbb{Z}_m^*$, which are defined by $\sigma_i(\zeta_m) = \omega_m^i$ where $\omega_m = \exp(2\pi\sqrt{-1}/m) \in \mathbb{C}$ is the principal mth complex root of unity. These embeddings come in conjugate pairs (σ_i, σ_{m-i}), because ω_m^i is the complex conjugate of $\omega_m^{m-i} = \omega_m^{-i}$. The *canonical embedding* is the concatenation of all the embeddings (under a suitable reindexing of $\mathbb{Z}_m^*$ as $\{1, \ldots, n\}$), i.e., the injective function

$$\sigma\colon K \to H$$
$$\sigma(a) = (\sigma_i(a))_{i \in \mathbb{Z}_m^*}$$

where $H \subset \mathbb{C}^n$ is the subspace defined above in Section 2.1.

We endow K and R with a geometry using the canonical embedding σ. For example, we define the ℓ_2 norm on K as $\|x\|_2 = \|\sigma(x)\|_2 = \sqrt{\langle \sigma(x), \sigma(x) \rangle}$, and use this to define the continuous Gaussian distribution D_r over K.[8]

Representations. Often, the mth cyclotomic ring is represented as $R \cong \mathbb{Z}[X]/(\Phi_m(X))$, where $\Phi_m(X)$ is the mth *cyclotomic polynomial*, using the natural "power basis:" every element of R is uniquely represented as a $\mathbb{Z}$-linear combination of the powers $1, X, \ldots, X^{n-1}$. When $m = p$ is prime, we have $\Phi_p(X) = 1 + X + \cdots + X^{p-1}$, and when m is a power of a prime p, we have $\Phi_m(X) = \Phi_p(X^{m/p})$, but in other cases

[8]To be formal, the continuous Gaussian is defined over $K_\mathbb{R} := K \otimes_\mathbb{Q} \mathbb{R}$, which is analogous to K as the reals $\mathbb{R}$ are to the rationals $\mathbb{Q}$, and which is in bijective correspondence with H via the natural extension of σ. Because precision is always finite in any computational context, in this work we ignore the formal distinction between K and $K_\mathbb{R}$.

the mth cyclotomic polynomial need not have such a nice form, which makes computations more cumbersome. An alternative "tensored" representation, which was shown in [LPR13] to have better computational and geometric properties for cryptography, uses a multivariate polynomial ring with one variable per distinct prime divisor of m. For example, $\mathbb{Z}[X_1, X_2]/(\Phi_{m_1}(X_1), \Phi_{m_2}(X_2))$ when $m = m_1 m_2$ is the factorization of m into powers of two distinct primes. The *powerful basis* $\vec{p} \in R^n$ is the corresponding $\mathbb{Z}$-basis of monomials in this representation, i.e., the tensor product of the power bases of the individual prime-power cyclotomics. See [LPR13, Section 4] for further details. (Note that $\Lambda \circ \lambda$, which our implementation is based upon, defines the powerful basis in "digit reversed" order; see [CP16a].)

Ideal lattices. An *ideal* $\mathcal{I} \subseteq R$ is a nontrivial additive subgroup that is also closed under multiplication by R, i.e., $x \cdot r \in \mathcal{I}$ for any $x \in \mathcal{I}, r \in R$. The *norm* is defined as $\mathrm{N}(\mathcal{I}) := |R/\mathcal{I}|$, the index of $\mathcal{I}$ in R.

A *fractional ideal* $\mathcal{J} \subset K$ is a set that can be expressed as $\mathcal{J} = d^{-1} \cdot \mathcal{I}$ for some ideal $\mathcal{I} \subseteq R$ and $d \in R$. (We sometimes omit the word "fractional" when it is clear from context.) Its norm is defined as $\mathrm{N}(\mathcal{J}) := \mathrm{N}(\mathcal{I}) / \mathrm{N}(d)$. The fractional ideals form a group under multiplication (with R as the identity), where ideal multiplication is defined by $\mathcal{I}\mathcal{J} = \{\sum_i x_i y_i : x_i \in \mathcal{I}, y_i \in \mathcal{J}\}$. The norm map is then multiplicative: $\mathrm{N}(\mathcal{I}\mathcal{J}) = \mathrm{N}(\mathcal{I})\,\mathrm{N}(\mathcal{J})$.

Any (fractional) ideal $\mathcal{I}$ yields a lattice $\sigma(\mathcal{I}) \subset H$ under the canonical embedding. As usual, we often leave σ implicit and refer to $\mathcal{I}$ itself as a lattice. The following lower bound on the minimum distance of an ideal lattice is an immediate consequence of the arithmetic-mean/geometric-mean inequality.

Lemma 2.4. *For any fractional ideal $\mathcal{I} \subset K$, we have $\lambda_1(\mathcal{I}) \geq \sqrt{n} \cdot \mathrm{N}(\mathcal{I})^{1/n}$.*

Duality. Any fractional ideal $\mathcal{I} \subset K$ has a *dual* (fractional) ideal $\mathcal{I}^\vee$, which under the canonical embedding corresponds to (the complex conjugate of) the dual lattice of $\mathcal{I}$, i.e., $\sigma(\mathcal{I})$ and $\overline{\sigma(\mathcal{I}^\vee)}$ are duals. An important object in algebraic number theory and for the definition of Ring-LWE is the *codifferent* ideal $R^\vee \subset K$, the dual of the entire ring. The dual ideal is related to the inverse ideal via the codifferent: $\mathcal{I}^\vee = \mathcal{I}^{-1} R^\vee$. (See, e.g., [Con09] for further details and proofs.)

In the mth cyclotomic, $R^\vee = t^{-1}R$ for special elements $t, g \in R$ satisfying $t \cdot g = \hat{m}$, where $\hat{m} = m/2$ when m is even, and $\hat{m} = m$ otherwise. (See [LPR13, Section 2.5.4] for further details and proofs.) The *decoding basis* $\vec{d}$ is a certain $\mathbb{Z}$-basis of $R^\vee$, which is the dual of (the complex conjugate of) the powerful basis $\vec{p}$ described above. It therefore has an analogous tensorial factorization, and good geometric properties: in particular, spherical Gaussians have relatively small coefficients with respect to $\vec{d}$. Because $tR^\vee = R$, it follows that $t \cdot \vec{d}$ is a $\mathbb{Z}$-basis of R, which we call the decoding basis of R. (See [LPR13, Section 6] for further details.)

2.3 (Tweaked) Ring-LWE

Ring-LWE is a family of computational problems that was defined and analyzed in [LPR10, LPR13]. Those works use a form of Ring-LWE involving the dual ideal $R^\vee$. More specifically, the search-R-LWE$_{q,\psi}$ problem, for an integer modulus $q > 1$ defining $R_q := R/qR$ and $R_q^\vee := R^\vee/qR^\vee$, and an error distribution ψ over K, is to find a uniformly random secret $s \in R_q^\vee$ given many independent "noisy" products

$$(a_i \in R_q \ , \ b_i = s \cdot a_i + e_i \bmod qR^\vee),$$

where each a_i is uniformly random (note that $a_i \cdot s \in R_q^\vee$), and each e_i is drawn from ψ. Typically, ψ is either a continuous spherical Gaussian or its discretization to $R^\vee$; these respectively give us *continuous* (where $b_i \in K/qR^\vee$) and *discrete* (where $b_i \in R_q^\vee$) forms of the problem.

For cryptographic applications and implementations, it can be convenient to use a form of Ring-LWE that does not involve $R^\vee$. Following [AP13, CP16a], this can be done with no loss in security or efficiency by using an equivalent "tweaked" form of the problem, which is obtained by implicitly multiplying the noisy products b_i by the "tweak" factor $t = \hat{m}/g \in R$, which satisfies $t \cdot R^\vee = R$. Doing so yields new values

$$b_i' := t \cdot b_i = (t \cdot s) \cdot a_i + (t \cdot e_i) = s' \cdot a_i + e_i' \bmod qR,$$

where $a_i, s' = t \cdot s \in R_q$, and the errors $e_i' = t \cdot e_i$ come from the "tweaked" error distribution $t \cdot \psi$. Note that when ψ corresponds to a spherical Gaussian, its tweaked form $t \cdot \psi$ may be *highly non-spherical*, but this is not a problem: tweaked Ring-LWE is entirely equivalent to the above one involving $R^\vee$, because the tweak is reversible. (See [CP16a] for further details on the recommended usage of tweaked Ring-LWE in cryptographic applications.)

In this paper, our exposition primarily uses the original form of Ring-LWE involving $R^\vee$, so that we can use sharp concentration bounds on spherical Gaussians. Our implementation, however, uses the tweaked form, where equivalent bounds follow by $\|g \cdot e'\| = \|g \cdot t \cdot e\| = \hat{m} \cdot \|e\|$, where e is the original error term and $e' = t \cdot e$ is its tweaked counterpart.

3 Cut-and-Choose Protocol

A central issue in the creation of challenges for LWE-like problems is that a dishonest challenger could publish improperly generated instances that are much harder than honestly generated ones, or even impossible to solve, because they have larger error than claimed or are even uniformly random. Because both the proper and improper distributions are conjectured to be pseudorandom, such misbehavior would be very difficult to detect. This stands in contrast to other types of cryptographic challenges for, e.g., the factoring or discrete logarithm problems, where improper distributions like unbalanced factors or non-uniform exponents seem like they can only make the instances *easier* to solve (or at least no harder), so the challenger has no incentive to use them.

To deal with this issue, we use a simple, non-interactive, publicly verifiable "cut-and-choose" protocol to give reasonably convincing evidence that the challenge instances are properly distributed, or at least not much harder than claimed. The protocol uses a *timestamp service* and a *randomness beacon*. The former allows anyone to verify that a given piece of data was generated and submitted to the service before a certain point in time. The latter is a source of public, timestamped, truly random bits. Concretely, for timestamps we use the Bitcoin blockchain via the OriginStamp service [GB14], and for randomness we use the NIST beacon [NIS11].[9]

3.1 Protocol Description and Properties

At a high level, our protocol proceeds as follows:

1. For each challenge instantiation (i.e., type of problem and concrete parameter set), the challenger *commits* by generating and publishing a moderately large number N (e.g., $N = 32$) of independent

[9] The use of a centralized beacon means that verifiers must trust that the challenger cannot predict or influence the beacon values, e.g., by collusion. This is obviously suboptimal from a security standpoint. Unfortunately, there appear to be few if any decentralized and practically usable alternatives that meet our needs. For example, while the Bitcoin blockchain has been proposed and analyzed as a source of randomness, it turns out to be relatively easy and inexpensive to introduce significant bias [BCG15, PW16].

instances, along with a distinct *beacon address* indicating a time in the near future, e.g., a few days later. The challenger also *timestamps* the commitment.[10]

2. At the announced time, the challenger obtains from the beacon a random value $i \in \{0, \ldots, N-1\}$.
3. The challenger then publicly *reveals* the secrets (which also implicitly reveals the errors) underlying all the instances except for the ith one. The one unrevealed instance is then considered the "official" challenge instance for its instantiation, and the others are considered "spoiled."
4. Anyone who wishes to *verify* the challenge checks that:
   (a) the original commitment was timestamped sufficiently in advance of the beacon address (and all beacon addresses across multiple challenges are distinct);
   (b) secrets for the appropriate instances were revealed, as indicated by the beacon value; and
   (c) the revealed secrets appear "proper." For Ring-LWE, one checks that the errors are short enough, potentially along with other statistical tests, e.g., on the errors' covariance. For Ring-LWR one recomputes the rounded products with the revealed secret and compares them to the challenge instance.

Importantly, a verifier does not need to witness the challenger's initial commitment firsthand, because it can just check the timestamp. In addition, the beacon's random outputs are cryptographically signed, and can be downloaded and verified at any time, or even provided by the challenger in the reveal step (which is what our implementation does).

Under the reasonable assumptions that the challenger cannot backdate timestamps, nor predict or influence the output of the randomness beacon, the above protocol provides the following guarantee: if one or more of the instances in a particular challenge are "improper," i.e., they lack a secret that would convince the verifier, then the challenger has probability at most $1/N$ of convincing the verifier. (Moreover, if two or more of the instances are improper, then the challenger can never succeed.)

Potential cheats and countermeasures. It is important to notice that as described, the protocol does not prove that the instances were *correctly sampled* according to the claimed Ring-LWE distribution, only that the revealed errors satisfy the statistical tests (i.e., they are short enough, etc.). Below in Section 3.2 we describe a supplementary (but platform- and implementation-specific) test, which we also include in our implementation, that gives a stronger assurance of correct sampling. However, the above protocol already seems adequate for practical purposes, because there does not appear to be any significant advantage to the challenger in choosing non-uniform $a_i \in R_q$ or $s \in R_q^\vee$, nor in deviating from spherical Gaussian errors within the required error bound. In particular, spherical Gaussians are rotationally invariant, and have maximal entropy over all distributions bounded by a given covariance.

Another way the challenger might try to cheat is a variant of the "perfect prediction" stock market scam: the challenger could prepare and timestamp a large number of different initial commitments (Step 1) containing various invalid instances. The challenger's goal is for at least one of these commitments to be successfully revealable once the beacon values become available; the challenger would then publish only that (timestamped) commitment as the "official" one, and discard the rest. The more commitments it prepares in advance, the more invalid (but unrevealed) instances it can hope to sneak past the verifier. However, the number of commitments it must prepare grows exponentially with the number of invalid instances.

[10] All the challenger's public messages are cryptographically signed under a known public key. This is for the challenger's protection, so that other parties cannot publish bogus data in its name.

In order to rule out this kind of misbehavior, we prove that there is a *single* commitment by widely announcing it (or its hash value under a conjectured collision-resistant hash function) *before* the beacon values become available, in several venues where it would be hard or impossible to make multiple announcements or suppress them at a later time. For example, on the IACR ePrint archive we have created one dated submission for this paper, every version of which contains the same hash value of the commitment (in Footnote 3). Also, we announced the hash value at the IACR Crypto 2016 Rump Session, which was streamed live on the Internet and is available for replay on YouTube.

3.2 Alternative Protocols

Here we describe some potential alternative approaches for validating Ring-LWE challenges, and analyze their strengths and drawbacks.

Publishing PRG seeds. As noted above, revealing the secrets and errors does not actually prove that the instances were sampled from the claimed Ring-LWE distribution. To address this concern, the challenger could generate each instance *deterministically*, making its random choices using the output of a cryptographically secure pseudorandom generator (PRG) on a short truly random seed. Then to reveal an instance, the challenger would simply reveal the corresponding seed, which the verifier would use to regenerate the instance and check that it matches the original one. We caution that this method still does not *guarantee* that the instances are properly sampled, because the challenger could still introduce some bias by generating many instances and suppressing ones it does not like, or even choosing seeds maliciously. However, publishing PRG seeds seems to significantly constrain a dishonest challenger's options for misbehavior. (Using a public randomness beacon is not an option, because some of the PRG seeds must remain secret.)

There are a few significant practical drawbacks to this approach. First, establishing any reasonable level of assurance requires the verifier to understand and run the challenger-provided code of the instance generator, rather than just checking that its outputs appear "proper," as the above protocol does. This also makes it difficult to write an alternative verification program (e.g., in a different programming language) without specifying exactly how the PRG output bits are consumed by the instance generator, which is cumbersome for continuous distributions like Gaussians. Second, even the provided verification code might be platform-specific: using different compiler versions or CPUs could result in different outputs on the same seed, due to differences in how the PRG output bits are consumed.[11]

Despite the above drawbacks, however, using and revealing PRG seeds does not need to *replace* the above protocol, but can instead *supplement* it to provide an extra layer of assurance. Therefore, our challenger and verifier also implement this method (and allow for very small $\leq 2^{-20}$ differences in floating-point values, to account for compiler differences). A failed match does not necessarily indicate misbehavior on the challenger's part, but is output as a warning by the verifier.

Zero-knowledge proofs. Another possibility is to view a Ring-LWE instance as a Bounded Distance Decoding (BDD) problem on a lattice, and have the challenger give a non-interactive zero-knowledge proof that it knows a solution within a given error bound. This can be done reasonably efficiently via, e.g., the public-coin protocol of [MV03] or Stern-style protocols for LWE-like problems [LNSW13], using a randomness beacon to provide the public coins. While at first glance this appears to provide exactly what we

[11] We actually witnessed this phenomenon during development: different compilers yielded very small differences in the floating-point values of our continuous Ring-LWE instances, but not our discrete ones. We attribute this to the compilers producing different orders of instructions, and the non-associativity/commutativity of floating-point arithmetic.

need, it turns out *not to give any useful guarantee*, due to the *approximation gap* between the completeness and soundness properties.

In more detail, for a BDD error bound B, an honest prover can always succeed in convincing the verifier that the error is at most B. However, the soundness guarantees only prevent a dishonest prover from succeeding when the BDD error is significantly larger than B. Specifically, the protocol from [MV03] has a bound of $\approx B\sqrt{d}$ where d is the lattice dimension, and the protocol from [LNSW13] only proves that the largest *coefficient* (in some basis) of the error is bounded. For our Gaussian error distributions, this bound would need to be about 2–3 times larger than the size of a typical coefficient. In summary, these protocols can only guarantee that the error is bounded by (say) $2B$, which can correspond to a much harder Ring-LWE instance than one with error bound B. By contrast, our protocol has a gap of only 10-15%, as shown next.

3.3 Verifier and Error Bounds

Here we describe our verifier in more detail, including some relevant aspects of its implementation, and describe how we compute rather sharp error bounds for our Ring-LWE instantiations.

Recall that each of our Ring-LWE instantiations is parameterized by a cyclotomic index m defining the mth cyclotomic number field K and cyclotomic ring R, which have degree $n = \varphi(m)$; a positive integer modulus q defining $R_q := R/qR$ and $R_q^\vee := R^\vee/qR^\vee$; and a Gaussian error parameter $r > 0$. (The number of samples is also a parameter, but it plays no role in the bounds.)

Verification. To verify a (continuous) Ring-LWE instance consisting of samples $(a \in R_q, b \in K/qR^\vee)$ for a purported secret $s \in R_q^\vee$ and given error bound B, one does the following for each sample:

1. compute $\bar{e} := b - s \cdot a \in K/qR^\vee$,
2. express $\bar{e}$ with respect to the decoding basis $\vec{d} = (d_j)$ of $R^\vee$, as $\bar{e} = \sum_j \bar{e}_j d_j$ where each $\bar{e}_j \in \mathbb{Q}/q\mathbb{Z}$.
3. "lift" $\bar{e} \in K/qR^\vee$ to a representative $e \in K$, defined as $e = \sum_j e_j d_j$ where each $e_j \in \mathbb{Q} \cap [-\frac{q}{2}, \frac{q}{2})$ is the distinguished representative of $\bar{e}_j$.
4. check that $\|e\| \leq B$ (where recall that $\|e\| := \|\sigma(e)\|$, the length of the canonical embedding of e).

For a discrete instance one does the same, but with K replaced by $R^\vee$ and $\mathbb{Q}$ replaced by $\mathbb{Z}$. In either case, properly generated Ring-LWE samples for our instantiations will correctly verify (with high probability) because the original errors $e \in K$ have coefficients of magnitude smaller than $q/2$ with respect to the decoding basis, hence they are correctly recovered from $b - s \cdot a = e \bmod qR^\vee$. Moreover, we show below that they have Euclidean norms below the error bound B with high probability.

Implementation. As mentioned in Section 2.3, our $\Lambda \circ \lambda$-based implementation actually uses the "tweaked" form of Ring-LWE, in which $R^\vee$ is replaced by R by implicitly multiplying each b component, and thereby the secret s and each error term e, by the "tweak" factor t (where $tR^\vee = R$). Correspondingly, the basis $t \cdot \vec{d}$ is referred to as the decoding basis of R. Therefore, we use an equivalent verification procedure to the one above, which simply replaces $R^\vee, \vec{d}$ with $R, t \cdot \vec{d}$, and the test $\|e\| \leq B$ with $\|g \cdot e\| \leq \hat{m}B$, where $g \in R$ is the special element such that $g \cdot t = \hat{m}$. (Recall that $\hat{m} = m/2$ when m is even, and $\hat{m} = m$ otherwise.)

The $\Lambda \circ \lambda$ framework provides operations for efficiently "lifting" elements of K/qR or R/qR to K or R (respectively) using the decoding basis of R, and for computing $\|g \cdot e\|$, exactly as required. Actually, because it is computationally simpler, $\Lambda \circ \lambda$ works with *squared* norms, Gaussian parameters, error bounds, etc., so our verifier checks the equivalent condition $\|g \cdot e\|^2 \leq (\hat{m}B)^2$.

Continuous error bound. For continuous Ring-LWE instantiations with spherical Gaussian error D_r over K, we use Lemma 2.1 and Corollary 2.2 to get rather sharp tail bounds on the Euclidean norm of the error. In our actual challenge instances, the error bound we use was typically within a factor of ≈ 1.10 of the largest error in each instance, so it gives little room for misbehavior relative to the correct error distribution.

The bound is obtained as follows. For an appropriate small $\varepsilon > 0$ we compute the minimal $c > 1/\sqrt{2\pi}$ (up to $\approx 10^{-4}$ precision) such that

$$\pi c^2 - \ln c \geq \frac{1}{n} \ln(1/\varepsilon) + \frac{1}{2} \ln(2\pi e).$$

Then by Corollary 2.2, we have $\Pr_{x \sim D_r}[\|x\| > B] < \varepsilon$, where $B := cr\sqrt{n}$. Concretely, we set $\varepsilon = 2^{-25}$ to get a rather strict bound that is still not too likely to be violated over the tens of thousands of error terms across all the instances.

Discrete error bound. For Ring-LWE instantiations with spherical Gaussian error D_r over K, discretized (i.e., rounded off) to $R^\vee$ using the decoding basis $\vec{d}$, we need to use a high-probability bound on the norm of the discretized error. For this we use a combination of Corollary 2.2 and a (partially heuristic) analysis of the round-off term. In our actual challenge instances, the ultimate bound was typically within a factor of ≈ 1.15 of the largest error in each instance.

Our discrete bound is obtained as follows. We first compute the same bound $B = cr\sqrt{n}$ on D_r as above. Now, because D_r is above or near the "smoothing parameter" of $R^\vee$, the fractional part $\mathbf{f} \in [-\frac{1}{2}, \frac{1}{2})^n$ of its coefficient vector with respect to $\vec{d}$ is close to uniformly random; henceforth we model it as such. The discretization error is $f = \langle \vec{d}, \mathbf{f} \rangle \in K$, which corresponds to $\mathbf{Df}$ in the canonical embedding, where $\mathbf{D} = \sigma(\vec{d}) = (\sigma_i(d_j))_{i,j}$. Observe that

$$\|f\|^2 = \langle \mathbf{Df}, \mathbf{Df} \rangle = \mathbf{f}^t \mathbf{G} \mathbf{f},$$

where $\mathbf{G} = \mathbf{D}^* \cdot \mathbf{D}$ is the positive definite Gram matrix of $\mathbf{D}$.

We now analyze the trace $\mathrm{Tr}(\mathbf{G})$, and use this to obtain a high-probability tail bound on $\|f\|$. Note that by definition of the decoding basis, $\mathbf{G} = \mathbf{H}^{-1}$ is the inverse of the Gram matrix $\mathbf{H}$ of the powerful basis $\vec{p}$. When m is a prime p, the proof of [LPR13, Lemma 4.3] shows that $\mathbf{H} = p\mathbf{I}_{p-1} - \mathbf{1}$, so $\mathbf{G} = p^{-1}(\mathbf{I}_{p-1} + \mathbf{1})$, which has trace $\mathrm{Tr}(\mathbf{G}) = 2(p-1)/p = 2n/m$. By the tensorial decomposition of the powerful and decoding bases, this immediately generalizes for arbitrary m to

$$\mathrm{Tr}(\mathbf{G}) = \frac{2^k n}{m},$$

where k is the number of distinct primes dividing m.

Recalling that we model $\mathbf{f} \in [-\frac{1}{2}, \frac{1}{2})^n$ as uniformly random, by independence of f_i, f_j for $i \neq j$ and linearity of expectation we have

$$\mathop{\mathbb{E}}_{f}[\|f\|^2] = \mathop{\mathbb{E}}_{\mathbf{f}}[\mathbf{f}^t \mathbf{G} \mathbf{f}] = \frac{1}{12} \mathrm{Tr}(\mathbf{G}) = \frac{2^k n}{12m}.$$

We heuristically assume that $\sigma(f) = \mathbf{Df}$ obeys essentially the same concentration bound (Lemma 2.1) as a spherical Gaussian having the above expected squared norm, times a small constant factor to account for the somewhat heavier tails (due to the non-spherical, non-Gaussian distribution). Our ultimate bound is $\sqrt{B^2 + F^2}$, where $B = cr\sqrt{n}$ and $F = c\sqrt{2^k n/m}$ are the high-probability bounds on the norms of D_r and the rounding term f, respectively.

4 Parameters

Here we give further details on how we choose the parameters of our instantiations, particularly the Gaussian error parameters r (Section 4.1) and modulus q (Section 4.2).

4.1 Error Parameter

As already mentioned in Section 1.2.2, we consider four categories of parameter r for the Gaussian error distribution D_r over K: "Trenta," "Grande," "Tall," and "Short." For all categories except Grande, the descriptions in Section 1.2.2 give the exact Gaussian parameter, or range of parameters, that we use in our instantiations.

For the Grande category, we use parameters that in particular have provable immunity to the "homomorphism" attack explored in [EHL14, ELOS15, CLS15, CLS16]. In [Pei16] it was shown that $r \geq 2$ is a sufficient condition for such immunity (in rings of cryptographically relevant dimensions). Here we generalize and tighten the analysis to obtain better bounds, which we use in our Grande instantiations.

The homomorphism attack on the original (non-"tweaked") definition of decision-Ring-LWE is as follows. (This is for the continuous form; it adapts immediately to the discrete form by replacing K with $R^\vee$.) Let ψ be an arbitrary error distribution over K, and let $\mathcal{I} \subseteq R$ be any ideal divisor of qR. We are given independent samples $(a_i, b_i) \in R_q \times K/qR^\vee$, which are distributed either uniformly or according to the Ring-LWE distribution for some secret $s \in R_q^\vee$. We first reduce the samples to

$$(a'_i = a_i \bmod \mathcal{I}\,,\ b'_i = b_i \bmod \mathcal{I}R^\vee) \in R/\mathcal{I} \times K/(\mathcal{I}R^\vee).$$

Then for each of the $\mathrm{N}(\mathcal{I})$ candidate (reduced) secrets $s' \in R^\vee/\mathcal{I}R^\vee$, we try to distinguish the $d'_i := b'_i - s' \cdot a'_i \in K/\mathcal{I}R^\vee$ from uniform. (How this is done does not matter for the present discussion.) Observe that if the samples come from the Ring-LWE distribution, i.e., $b_i = s \cdot a_i + e_i \bmod qR^\vee$ for $e_i \leftarrow \psi$, then for the correct candidate $s' = s \bmod \mathcal{I}R^\vee$ we have $d'_i = e_i \bmod \mathcal{I}R^\vee$.

Observe that the above attack takes time at least $\mathrm{N}(\mathcal{I})$ times the number of samples consumed, and that it can work *only if* the reduced error distribution $\psi \bmod \mathcal{I}R^\vee$ has noticeable statistical distance from uniform over $K/\mathcal{I}R^\vee$. Otherwise, the d'_i are statistically indistinguishable from uniform for any candidate s', regardless of the form of the original samples (uniform or Ring-LWE), and the attack fails.

Immunity to homomorphism attack. The following lemma gives a sufficient condition on the parameter of Gaussian error $\psi = D_r$ to ensure that the homomorphism attack has exponentially large time/advantage ratio t^n, for any desired $t > 1$. (Note that the proof never uses the fact that $\mathcal{I}$ divides qR.) For simplicity, in our Grande instantiations we always use $t = 2$ and hence $r = \sqrt{8/(\pi e)} \approx 0.968$. For dimensions (say) $n > 256$ one could take $t = 2^{256/n}$ to obtain an even smaller r.

Lemma 4.1. *For any $n \geq 17$, $t > 1$, and $r \geq t\sqrt{2/(\pi e)} \approx 0.484t$, the time/advantage ratio of the homomorphism attack (for any choice of the ideal $\mathcal{I}$) is at least t^n.*

Proof. Let $s = \mathrm{N}(\mathcal{I})^{1/n}$, and note that the running time of the attack is at least $\mathrm{N}(\mathcal{I}) = s^n$, so we may assume without loss of generality that $s \leq t$.

The dual ideal of $\mathcal{I}R^\vee$ is $(\mathcal{I}R^\vee)^{-1} \cdot R^\vee = \mathcal{I}^{-1}$, which has norm $\mathrm{N}(\mathcal{I})^{-1}$, so by Lemma 2.4 its minimum distance is $\lambda_1(\mathcal{I}^{-1}) \geq \sqrt{n}/s$. Letting $f(x) = \sqrt{2\pi e} \cdot x \cdot \exp(\pi x^2)$ be as in Equation (2.1), define

$$
\begin{aligned}
c &:= \frac{r\lambda_1(\mathcal{I}^{-1})}{\sqrt{n}} \geq \frac{r}{s} \geq \frac{r}{t} \geq \sqrt{2/(\pi e)} > 1/\sqrt{2\pi}, \\
C &:= f(c) \leq 2\exp(-2/e) < 2^{-1/17},
\end{aligned}
$$

where the penultimate inequality follows by $c \geq \sqrt{2/(\pi e)}$ and the fact that f is decreasing for $x \geq 1/\sqrt{2\pi}$.

By Lemma 2.3, the statistical distance between $D_r \bmod \mathcal{I}R^\vee$ and the uniform distribution over $K/\mathcal{I}R^\vee$ is at most $\frac{1}{2}C^n/(1-C^n)$. Then because $n \geq 17$, the time/advantage ratio of the attack is

$$
\frac{2(1-C^n)\,\mathrm{N}(\mathcal{I})}{C^n} \geq \frac{\mathrm{N}(\mathcal{I})}{C^n} = (s/C)^n,
$$

so it remains to show that $s/C \geq t$. By the previous observation on $f(x)$ and the fact that $c \geq r/s > 1/\sqrt{2\pi}$,

$$
s/C = s/f(c) \geq s/f(r/s) = \frac{r}{\sqrt{2\pi e} \cdot (r/s)^2 \cdot \exp(-\pi(r/s)^2)}.
$$

A straightforward calculation shows that the denominator (as a function of s) has a global maximum when $r/s = 1/\sqrt{\pi}$, so as desired, $s/C \geq r\sqrt{\pi e/2} \geq t$. $\square$

4.2 Modulus

For a given Gaussian error parameter r, we choose moduli q to reflect a typical Ring-LWE public-key encryption or key-exchange application following the basic template from [LPR10, Pei14]. Essentially, this means that q must be large enough to accomodate the ultimate error term, which is a combination of the original errors, without any "wraparound." A bit more precisely, we need that with sufficiently high probability, the ultimate error has coefficients (with respect to an appropriate choice of basis) in the interval $(-\frac{q}{4}, \frac{q}{4})$. The precise meaning of "high probability" depends on the low-level details of the application. For example, wraparound of a few coefficients might be acceptable if error-correcting codes are used, or a final key-confirmation step may handle the rare case when wraparound does occur.

The Ring-LWE "toolkit" [LPR13] provides general techniques and reasonably sharp concentration bounds for analyzing the coefficients of sums and products of (discretized) error terms in arbitrary cyclotomics (see, e.g., [LPR13, Lemma 6.6]). However, their generality makes them a bit pessimistic, so they do not capture the strongest possible concentration properties for concrete cases of interest.

In this work we take a combined empirical and theoretical approach to more tightly bound the ultimate error in encryption/key-exchange applications, and thereby obtain smaller values of the modulus and larger error rates. Our empirical approach is as follows:

1. We simulate thousands of ultimate error terms $E := \hat{m}(e \cdot e' + f \cdot f') \in R^\vee$, where $e, e', f, f' \in R^\vee$ are independent samples from D_r, discretized to $R^\vee$ using the decoding basis.[12]
2. We compute the largest magnitude B among all the coefficients of all the Es (again with respect to the decoding basis), and use $4B$ as a heuristic "very high probability" bound on the coefficients.

[12] Depending on the primes dividing the cyclotomic index m, replacing the $\hat{m}$ factor by t in the expression for E can sometimes yield smaller coefficients. We use the best of the two choices in our simulation.

3. Using $4B$ as a lower bound on $q/4$, we choose moduli q of different arithmetic forms (e.g., completely split, power of two, ramified) that all conform to this bound.

The theoretical (though heuristic) basis for this approach is as follows: in the canonical embedding, the coordinates of D_r are i.i.d. Gaussians over $\mathbb{C}$ (up to conjugate symmetry), and the same *nearly* holds for the discretization to $R^\vee$ when D_r is "well-spread" relative to $R^\vee$ (as it is in our instantiations). Because multiplication is coordinate-wise in the canonical embedding, the products $e \cdot e', f \cdot f'$ have nearly i.i.d. subexponential coordinates. (The multiplication by $\hat{m}$ simply scales them all by the same factor.) Finally, each coefficient of E with respect to the decoding basis is by definition the inner product of $\sigma(E)$ with a vector consisting of various roots of unity. Bernstein's inequality says that such inner products have subgaussian $\exp(-\Theta(k^2))$ tail probabilities in the "near zone," which in our setting goes all the way out to $k = O(\sqrt{n})$ standard deviations. In the "far zone" beyond that, the tails are still subexponential $\exp(-\Theta(k))$.

Because the near zone is so wide, the largest coefficient among the tens or hundreds of thousands in our simulation should be not much smaller than a true high-probability bound. Concretely, the largest empirical coefficient B should have a tail probability of no more than, say, 2^{-13}. Under the subgaussian model, the probability of obtaining a coefficient of magnitude more than $4B$ is therefore less than $(2^{-13})^{4^2} = 2^{-208}$. Even under the weaker subexponential model, the probability is at most $(2^{-13})^4 = 2^{-52}$.

5 Hardness Estimates

In this section we describe how we obtain hardness estimates for our challenges. There are many different algorithmic approaches for attacking lattice problems like the approximate Shortest Vector Problem (SVP) and the Bounded Distance Decoding (BDD) problem, of which Ring-LWE/LWR are special cases. These include lattice-basis reduction (e.g., [LLL82, Sch87, GNR10, CN11, MW16]), exponential-time and -space sieving or Voronoi-based algorithms (e.g., [AKS01, NV08, MV10b, MV10a, Laa15, ADRS15]), combinatorial and algebraic attacks [BKW03, AG11, ACFP14], and combinations thereof (e.g., [How07]).

Because all the above approaches represent active areas of research and can be difficult to compare directly—especially because some require enormous memory—we do not attempt to give precise estimates of "bits of security." Instead, we follow the analysis approach of [MR09, LP11, LN13, ADPS16] for (Ring-)LWE to derive two kinds of hardness estimates. First, we give the approximate *root-Hermite factor* $\delta > 1$ needed to solve each challenge via lattice attacks. We use δ to classify each challenge into one of a few broad categories, ranging from "toy" (very easy) to "very hard" (likely out of reach for nation-state attackers using the best publicly known algorithms). Second, we estimate the smallest *block size* that is sufficient to solve the challenge using the BKZ algorithm [SE94, CN11].

In Appendix D, Table 1 and Table 2 give the hardness estimates for our Ring-LWE/LWR challenges, using the methods described below (specifically, Equations (5.1) and (5.2)).

5.1 Ring-LWE/LWR as BDD

A standard attack on Ring-LWE casts it as a Bounded Distance Decoding (BDD) problem on a random lattice from a certain class. For a collection of ℓ Ring-LWE samples $(a_i \in R_q, b_i = s \cdot a_i + e_i \bmod qR^\vee)$ defining $\vec{a} = (a_1, \dots, a_\ell)$, we consider the corresponding "q-ary" lattice

$$\mathcal{L}(\vec{a}) := \{\vec{v} \in (R^\vee)^\ell : \exists\, z \in R^\vee \text{ such that } \vec{v} = z \cdot \vec{a} \pmod{qR^\vee}\}.$$

The vector $\vec{b} = (b_1, \dots, b_\ell) \approx s \cdot \vec{a} \bmod qR^\vee$ is then a BDD target that is close to an element of $\mathcal{L}(\vec{a})$, and the BDD error is $\vec{e} = (e_1, \dots, e_\ell)$, where each e_i is distributed as the spherical Gaussian D_r.

The difficulty of BDD is primarily determined by the lattice dimension, and the width of the error relative to the (dimension-normalized) lattice determinant. Because $R^\vee$ is isomorphic as a group to $\mathbb{Z}^n$, we have that $\mathcal{L}(\vec{a})$ is an ℓn-dimensional lattice; however, by ignoring some coordinates we can view it as a d-dimensional lattice for any desired $d \in [n, \ell n]$. In order to most easily adapt the prior analyses for attacks on (Ring-)LWE, we also implicitly rescale the canonical embedding (thereby rescaling both the lattice and the error) by a factor of $\delta_R := \mathrm{vol}(\sigma(R))^{1/n}$, so that the rescaled $R^\vee$ has unit volume, just like $\mathbb{Z}^n$. The determinant of the lattice is then q^{d-n}—the same as for a d-dimensional LWE lattice—and the error is distributed as a spherical Gaussian of parameter $r' := \delta_R \cdot r$.

For Ring-LWR we proceed similarly, but because the rounding is done with respect the decoding basis of $R^\vee$—which in general is not orthogonal in the canonical embedding—we instead use the geometry given by identifying the decoding basis with the standard basis of $\mathbb{Z}^n$, and we model the rounding error in each coordinate as uniform in the interval $(-\frac{q}{2p}, \frac{q}{2p})$. This makes the rounding error isotropic and gives $R^\vee$ unit volume, and therefore yields the smallest ratio of error width to dimension-normalized determinant. Specifically, the lattice determinant is again q^{d-n}, and the error has standard deviation $\frac{q}{p}/\sqrt{12}$ in each coordinate, so we heuristically model it as a spherical Gaussian with parameter $r' := \frac{q}{p}\sqrt{\pi/6}$.

5.2 Root-Hermite Factor

The quality of lattice vectors, and the concrete hardness of obtaining them, is often measured by the *Hermite factor*: for a d-dimensional lattice $\mathcal{L}$, vector $\mathbf{v} \in \mathcal{L}$ has Hermite factor δ^d given by $\|\mathbf{v}\| = \delta^d \cdot \mathrm{vol}(\mathcal{L})^{1/d}$; we call δ the *root-Hermite factor*. Experiments on random lattices indicate that δ is a very good indicator of hardness in cryptographically relevant dimensions. For example, $\delta \approx 1.022$ and $\delta \approx 1.011$ are efficiently obtainable by the LLL and BKZ-28 algorithms (respectively) [GN08], whereas $\delta = 1.005$ is considered far out of practical reach for $d \geq 500$ [CN11]. To our knowledge, the best publicly demonstrated root-Hermite factors for cryptographic dimensions are $\delta \approx 1.00955$ or more, on the Darmstadt lattice challenges [LRBN10].

Assuming that the error is sufficiently "smooth" over the integers, which is the case for all our challenges, the analyses of [MR09, LP11, LN13] show that one can solve LWE/BDD with some not-too-small probability by obtaining a root-Hermite factor δ given by

$$\lg \delta = \frac{\lg^2(Cq/r')}{4n \lg q}. \tag{5.1}$$

Here the factor C influences the success probability: larger values correspond to smaller chance of success. For example, extrapolating from [LN13, Table 2] for $n \leq 256$, taking $C \in [1.7, 2.5]$ can yield probability ≈ 1 (depending on the exact dimension); $C \approx 3.0$ corresponds to probability $\approx 2^{-32}$; and $C \approx 4.0$ corresponds to probability $\approx 2^{-64}$. (These are only rough estimates, and can be affected by the number of iterations, choice of pruning strategy, etc.) In our estimates, for simplicity we always use $C = 2.0$.

We use our root-Hermite factor estimates to classify each challenge into one of several qualitative hardness categories. The category thresholds are given in Figure 1.

5.3 BKZ Block Size

Another very good indication of hardness for a BDD instance is the smallest *block size* needed for the success of the BKZ lattice-basis reduction algorithm [SE94, CN11]. This parameter is a useful proxy for hardness because the runtime for BKZ is at least exponential in the block size.

Class	$\delta >$
Toy	1.011
Easy	1.0095
Moderate	1.0075
Hard	1.005
Very Hard	1.0

Figure 1: Root-Hermite factor thresholds for our qualitative hardness estimates. Each challenge is classified according the largest applicable threshold (i.e., the weakest category.)

Heuristic algorithms exist to approximate the runtime of BKZ [CN11, Che13], but they focus on the runtime of an SVP subroutine. This subroutine is called many times by the BKZ algorithm, but there are no precise estimates for the number of calls, and hence no very precise estimates for the total runtime of BKZ. Furthermore, the heuristic estimates are for sufficiently large block sizes in high dimensions, while some of our challenges have low dimension or can be attacked with a relatively small block size. Therefore, rather than provide an imprecise "bits of security" estimate, we instead give the approximate block size needed for the BKZ algorithm to successfully solve each challenge.

The "primal" form of the BKZ attack on LWE/BDD is most easily explained using Kannan's embedding technique, which converts a d-dimensional BDD instance with error $\vec{e}$ to a $(d+1)$-dimensional SVP instance with a "planted" shortest vector $(\vec{e}, 1)$.[13] When BKZ is run with a large enough block size b, it successfully finds the planted shortest vector. More specifically, by modeling the behavior of BKZ using the geometric series assumption (GSA) [Sch03], and assuming the error is Gaussian with parameter r', the analysis of [ADPS16] shows that the attack succeeds when

$$r'\sqrt{b/(2\pi)} \leq \kappa^{2b-d-1} \cdot q^{1-n/d}, \tag{5.2}$$

where $\kappa = ((\pi b)^{1/b} \cdot b/(2\pi e))^{1/(2b-2)}$ is the GSA factor. We optimize our choice of $d \in [n, \ell n]$ to minimize the block size needed for each challenge.

References

[AA16] J. Alperin-Sheriff and D. Apon. Dimension-preserving reductions from LWE to LWR. Cryptology ePrint Archive, Report 2016/589, 2016. `http://eprint.iacr.org/2016/589`.

[ACF+15] M. R. Albrecht, C. Cid, J. Faugère, R. Fitzpatrick, and L. Perret. On the complexity of the BKW algorithm on LWE. *Designs, Codes and Cryptography*, 74(2):325–354, 2015.

[ACFP14] M. R. Albrecht, C. Cid, J.-C. Faugère, and L. Perret. Algebraic algorithms for LWE. Cryptology ePrint Archive, Report 2014/1018, 2014. `http://eprint.iacr.org/2014/1018`.

[ACPS09] B. Applebaum, D. Cash, C. Peikert, and A. Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In *CRYPTO*, pages 595–618. 2009.

[13] Alkim *et al.* [ADPS16], found that by adjusting the parameters appropriately, the best "dual" attack required an almost identical block size as the primal attack, so we do not consider it here.

[ADPS16] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe. Post-quantum key exchange - a new hope. In *USENIX Security Symposium*, pages ??–?? 2016.

[ADRS15] D. Aggarwal, D. Dadush, O. Regev, and N. Stephens-Davidowitz. Solving the shortest vector problem in 2^n time using discrete Gaussian sampling. In *STOC*, pages 733–742. 2015.

[AFG13] M. R. Albrecht, R. Fitzpatrick, and F. Göpfert. On the efficacy of solving LWE by reduction to unique-SVP. In *ICISC*, pages 293–310. 2013.

[AG11] S. Arora and R. Ge. New algorithms for learning in presence of errors. In *ICALP (1)*, pages 403–415. 2011.

[AKPW13] J. Alwen, S. Krenn, K. Pietrzak, and D. Wichs. Learning with rounding, revisited - new reduction, properties and applications. In *CRYPTO*, pages 57–74. 2013.

[AKS01] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *STOC*, pages 601–610. 2001.

[AP13] J. Alperin-Sheriff and C. Peikert. Practical bootstrapping in quasilinear time. In *CRYPTO*, pages 1–20. 2013.

[APS15] M. R. Albrecht, R. Player, and S. Scott. On the concrete hardness of learning with errors. *J. Mathematical Cryptology*, 9(3):169–203, 2015.

[Ban93] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(4):625–635, 1993.

[BBG$^+$16] J. A. Buchmann, N. Büscher, F. Göpfert, S. Katzenbeisser, J. Krämer, D. Micciancio, S. Siim, C. van Vredendaal, and M. Walter. Creating cryptographic challenges using multi-party computation: The LWE challenge. In *AsiaPKC*, pages 11–20. 2016.

[BBL$^+$14] A. Banerjee, H. Brenner, G. Leurent, C. Peikert, and A. Rosen. SPRING: Fast pseudorandom functions from rounded ring products. In *FSE*, pages 38–57. 2014.

[BCD$^+$16] J. W. Bos, C. Costello, L. Ducas, I. Mironov, M. Naehrig, V. Nikolaenko, A. Raghunathan, and D. Stebila. Frodo: Take off the ring! practical, quantum-secure key exchange from LWE. In *CCS*, pages 1006–1018. 2016.

[BCG15] J. Bonneau, J. Clark, and S. Goldfeder. On Bitcoin as a public randomness source. Cryptology ePrint Archive, Report 2015/1015, 2015. `http://eprint.iacr.org/2015/1015`.

[BCNS15] J. W. Bos, C. Costello, M. Naehrig, and D. Stebila. Post-quantum key exchange for the TLS protocol from the ring learning with errors problem. In *IEEE Symposium on Security and Privacy*, pages 553–570. 2015.

[BGM$^+$16] A. Bogdanov, S. Guo, D. Masny, S. Richelson, and A. Rosen. On the hardness of learning with rounding over small modulus. In *TCC*, pages 209–224. 2016.

[BK15] E. Barker and J. Kelsey. Recommendation for random number generation using deterministic random bit generators, June 2015. NIST Special Publication 800-90A, revision 1.

[BKW03] A. Blum, A. Kalai, and H. Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *J. ACM*, 50(4):506–519, 2003.

[BLP$^+$13] Z. Brakerski, A. Langlois, C. Peikert, O. Regev, and D. Stehlé. Classical hardness of learning with errors. In *STOC*, pages 575–584. 2013.

[BPR12] A. Banerjee, C. Peikert, and A. Rosen. Pseudorandom functions and lattices. In *EUROCRYPT*, pages 719–737. 2012.

[Bra16] M. Braithwaite. Experimenting with post-quantum cryptography, July 2016. `https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html`, last retrieved Aug 2016.

[BV11] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM J. Comput.*, 43(2):831–871, 2014. Preliminary version in FOCS 2011.

[CDPR16] R. Cramer, L. Ducas, C. Peikert, and O. Regev. Recovering short generators of principal ideals in cyclotomic rings. In *EUROCRYPT*, pages 559–585. 2016.

[CDW17] R. Cramer, L. Ducas, and B. Wesolowski. Short Stickelberger class relations and application to Ideal-SVP. In *EUROCRYPT*. 2017. To appear.

[Cer97] Certicom ECC challenge, November 1997. `https://www.certicom.com/images/pdfs/challenge-2009.pdf`, last retrieved Aug 2016.

[Che13] Y. Chen. *Lattice Reduction and Concrete Security of Fully Homomorphic Encryption*. Ph.D. thesis, Paris Diderot University, 2013.

[CLS15] H. Chen, K. Lauter, and K. E. Stange. Attacks on search RLWE. Cryptology ePrint Archive, Report 2015/971, 2015. `http://eprint.iacr.org/`.

[CLS16] H. Chen, K. Lauter, and K. E. Stange. Vulnerable Galois RLWE families and improved attacks. Cryptology ePrint Archive, Report 2016/193, 2016. `http://eprint.iacr.org/`.

[CN11] Y. Chen and P. Q. Nguyen. BKZ 2.0: Better lattice security estimates. In *ASIACRYPT*, pages 1–20. 2011.

[Con09] K. Conrad. The different ideal, 2009. Available at `http://www.math.uconn.edu/~kconrad/blurbs/`, last accessed 12 Oct 2009.

[CP16a] E. Crockett and C. Peikert. $\Lambda \circ \lambda$: Functional lattice cryptography. In *ACM CCS*, pages 993–1005. 2016. Full version at `http://eprint.iacr.org/2015/1134`.

[CP16b] E. Crockett and C. Peikert. $\Lambda \circ \lambda$ source code repository, 2016. `https://github.com/cpeikert/Lol`.

[DDLL13] L. Ducas, A. Durmus, T. Lepoint, and V. Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO*, pages 40–56. 2013.

[DORS04] Y. Dodis, R. Ostrovsky, L. Reyzin, and A. Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. *SIAM J. Comput.*, 38(1):97–139, 2008. Preliminary version in EUROCRYPT 2004.

[DuB15] T. DuBuisson. DRBG haskell package, Nov 2015. `https://hackage.haskell.org/package/DRBG`.

[EHL14] K. Eisenträger, S. Hallgren, and K. E. Lauter. Weak instances of PLWE. In *SAC*, pages 183–194. 2014.

[ELOS15] Y. Elias, K. E. Lauter, E. Ozman, and K. E. Stange. Provably weak instances of Ring-LWE. In *CRYPTO*, pages 63–92. 2015.

[GB14] A. Gernandt and B. Bipp. OriginStamp, 2014. `https://www.originstamp.org/`, last retrieved Aug 2016.

[GHS12] C. Gentry, S. Halevi, and N. P. Smart. Better bootstrapping in fully homomorphic encryption. In *Public Key Cryptography*, pages 1–16. 2012.

[GLP12] T. Güneysu, V. Lyubashevsky, and T. Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *CHES*, pages 530–547. 2012.

[GN08] N. Gama and P. Q. Nguyen. Predicting lattice reduction. In *EUROCRYPT*, pages 31–51. 2008.

[GNR10] N. Gama, P. Q. Nguyen, and O. Regev. Lattice enumeration using extreme pruning. In *EUROCRYPT*, pages 257–278. 2010.

[Goo08] Google. Protocol buffers (version 2), Jul 2008. `https://developers.google.com/protocol-buffers/`, last retrieved Aug 2016.

[GPV08] C. Gentry, C. Peikert, and V. Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In *STOC*, pages 197–206. 2008.

[HKM15] G. Herold, E. Kirshanova, and A. May. On the asymptotic complexity of solving LWE. Cryptology ePrint Archive, Report 2015/1222, 2015. `http://eprint.iacr.org/2015/1222`.

[How07] N. Howgrave-Graham. A hybrid lattice-reduction and meet-in-the-middle attack against NTRU. In *CRYPTO*, pages 150–169. 2007.

[HS] S. Halevi and V. Shoup. HElib: an implementation of homomorphic encryption. `https://github.com/shaih/HElib`, last retrieved May 2016.

[Laa15] T. Laarhoven. Sieving for shortest vectors in lattices using angular locality-sensitive hashing. In *CRYPTO*, pages 3–22. 2015.

[LLL82] A. K. Lenstra, H. W. Lenstra, Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, December 1982.

[LMPR08] V. Lyubashevsky, D. Micciancio, C. Peikert, and A. Rosen. SWIFFT: A modest proposal for FFT hashing. In *FSE*, pages 54–72. 2008.

[LN13] M. Liu and P. Q. Nguyen. Solving BDD by enumeration: An update. In *CT-RSA*, pages 293–309. 2013.

[LNSW13] S. Ling, K. Nguyen, D. Stehlé, and H. Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In *PKC*, pages 107–124. 2013.

[LP11] R. Lindner and C. Peikert. Better key sizes (and attacks) for LWE-based encryption. In *CT-RSA*, pages 319–339. 2011.

[LPR10] V. Lyubashevsky, C. Peikert, and O. Regev. On ideal lattices and learning with errors over rings. *Journal of the ACM*, 60(6):43:1–43:35, November 2013. Preliminary version in Eurocrypt 2010.

[LPR13] V. Lyubashevsky, C. Peikert, and O. Regev. A toolkit for ring-LWE cryptography. In *EUROCRYPT*, pages 35–54. 2013.

[LRBN10] R. Lindner, M. Rückert, P. Baumann, and L. Nobach. TU Darmstadt lattice challenge, 2010. `https://www.latticechallenge.org/`.

[LS16] I. Lovecruft and P. Schwabe. RebelAlliance: A post-quantum secure hybrid handshake based on NewHope, May 2016. `https://gitweb.torproject.org/user/isis/torspec.git/tree/proposals/XXX-newhope-hybrid-handshake.txt?h=draft/newhope`, last retrieved Aug 2016.

[Lyu05] V. Lyubashevsky. The parity problem in the presence of noise, decoding random linear codes, and the subset sum problem. In *APPROX-RANDOM*, pages 378–389. 2005.

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

[MP13] D. Micciancio and C. Peikert. Hardness of SIS and LWE with small parameters. In *CRYPTO*, pages 21–39. 2013.

[MR04] D. Micciancio and O. Regev. Worst-case to average-case reductions based on Gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007. Preliminary version in FOCS 2004.

[MR09] D. Micciancio and O. Regev. Lattice-based cryptography. In *Post Quantum Cryptography*, pages 147–191. Springer, February 2009.

[MV03] D. Micciancio and S. P. Vadhan. Statistical zero-knowledge proofs with efficient provers: Lattice problems and more. In *CRYPTO*, pages 282–298. 2003.

[MV10a] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In *STOC*, pages 351–358. 2010.

[MV10b] D. Micciancio and P. Voulgaris. Faster exponential time algorithms for the shortest vector problem. In *SODA*, pages 1468–1480. 2010.

[MW16] D. Micciancio and M. Walter. Practical, predictable lattice basis reduction. In *EUROCRYPT*, pages 820–849. 2016.

[NIS11] NIST randomness beacon, Sep 2011. `http://www.nist.gov/itl/csd/ct/nist_beacon.cfm`, last retrieved Aug 2016.

[NTR15] NTRU challenge, 2015. `https://www.securityinnovation.com/products/ntru-crypto/ntru-challenge`, last retrieved Aug 2016.

[NV08] P. Q. Nguyen and T. Vidick. Sieve algorithms for the shortest vector problem are practical. *J. Mathematical Cryptology*, 2(2):181–207, 2008.

[Pei09] C. Peikert. Public-key cryptosystems from the worst-case shortest vector problem. In *STOC*, pages 333–342. 2009.

[Pei14] C. Peikert. Lattice cryptography for the Internet. In *PQCrypto*, pages 197–219. 2014.

[Pei16] C. Peikert. How (not) to instantiate Ring-LWE. In *SCN*, pages 411–430. 2016.

[PRS17] C. Peikert, O. Regev, and N. Stephens-Davidowitz. Pseudorandomness of Ring-LWE for any ring and modulus. In *STOC*. 2017. To appear.

[PS13] T. Plantard and M. Schneider. Creating a challenge for ideal lattices. Cryptology ePrint Archive, Report 2013/039, 2013. `http://eprint.iacr.org/2013/039`.

[PW16] C. Pierrot and B. Wesolowski. Malleability of the blockchain's entropy. Cryptology ePrint Archive, Report 2016/370, 2016. `http://eprint.iacr.org/2016/370`.

[Reg05] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6):1–40, 2009. Preliminary version in STOC 2005.

[Reg16] O. Regev. Personal communication, 3 August 2016.

[RLW16] Ring-LWE challenges website, 2016. `https://web.eecs.umich.edu/~cpeikert/rlwe-challenges`.

[RSA91] RSA factoring challenge, March 1991. `http://www.emc.com/emc-plus/rsa-labs/historical/the-rsa-factoring-challenge-faq.htm`, last retrieved Aug 2016.

[Sch87] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987.

[Sch03] C.-P. Schnorr. Lattice reduction by random sampling and birthday methods. In *STACS*, pages 145–156. 2003.

[SE94] C.-P. Schnorr and M. Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Mathmatical Programming*, 66:181–199, 1994.

[Ste14] A. Steffen. strongSwan BLISS implementation, Dec 2014. `https://wiki.strongswan.org/projects/strongswan/wiki/BLISS`, updated Nov 2015, last retrieved Aug 2016.

[SV11] N. P. Smart and F. Vercauteren. Fully homomorphic SIMD operations. *Designs, Codes and Cryptography*, 71(1):57–81, 2014. Preliminary version in ePrint Report 2011/133.

[YDH+15] T. Yasuda, X. Dahan, Y.-J. Huang, T. Takagi, and K. Sakurai. MQ challenge: Hardness evaluation of solving multivariate quadratic problems. Cryptology ePrint Archive, Report 2015/275, 2015. `http://eprint.iacr.org/2015/275`.

A Implementation Notes

In this section we describe some of the lower-level technical details of our challenges, and the operational security measures we used when generating them.

Beacon addresses. Every 60 seconds the NIST randomness beacon [NIS11] announces a 512-bit string, which is identified by the corresponding *(Unix) epoch*, i.e., the number of seconds elapsed since 1 January 1970 00:00:00 UTC. (The beacon epochs are always divisible by 60.) For our cut-and-choose protocol, a *beacon address* is a pair (s, i) consisting of an epoch s and a zero-indexed offset $i \in \{0, \ldots, 63 = 512/8-1\}$, which indexes the ith byte of the beacon's output string for epoch s.

Each of our challenges is associated with a distinct beacon address, which is used to determine which of its $N = 32$ instances will become the "official" one; the remainder will have their secrets revealed in the cut-and-choose protocol (see Section 3 for details). A beacon address of (s, i) means that the official instance will be the one indexed by the ith byte of the beacon value for epoch s, interpreted as an unsigned 8-bit integer and reduced modulo 32. That is, we use the least-significant 5 bits of the ith byte, and ignore the rest.

To ensure distinct beacon addresses, we generated our challenges to have sequentially increasing addresses starting from epoch $1{,}471{,}449{,}600$ (corresponding to 17 August 2016 12:00:00 EDT) and index zero. "Sequentially increasing" means that the index increments from 0 to 63, after which the epoch increments (by 60) and the index is reset to zero.[14]

Randomness. As the source of randomness for generating each instance of our challenges, we used the Haskell `DRBG` implementation [DuB15] of the NIST standard CTR-DRBG-AES-128 [BK15] pseudorandom generator, with a 256-bit seed ("input entropy"). The seeds themselves were derived using the Hash-DRBG-SHA-512 generator [BK15], seeded with 512 bits of system entropy. We would have preferred to use Hash-DRBG-SHA-512 for all pseudorandomness, but its implementation in `DRBG` is much slower, and pseudorandom bit generation is currently the main bottleneck in our implementation.

Operational security. A primary goal when generating our challenges and executing the cut-and-choose protocol was to reduce the risk of unauthorized exfiltration of the underlying secrets, e.g., by malware or hacking.

We generated the challenges on a 2010 MacBook Pro laptop with a freshly installed operating system, which was never connected to any network and had all network interfaces disabled. We exclusively used write-once CD and DVD media for copying the challenge-generator executable to the laptop, and the challenges and revealed secrets from the laptop.[15]

We enabled FileVault encryption for the user account storage. As an extra layer of protection, we also created and stored the challenges and their secrets in a separately encrypted volume (within user storage), which was kept unmounted except when the challenges were being created or operated upon. The random passphrases for the user account and encrypted volume were generated and stored non-electronically, and were destroyed with fire once the cut-and-choose protocol was completed. Finally, we wiped the storage media with all-zeros. Therefore, we believe that the non-revealed secrets should be completely unrecoverable (even by us), except by solving the corresponding challenges.

[14] Actually, there are two non-sequential "jumps" in the beacon addresses of our challenges, corresponding to batches we created with different runs of the generator. However, all beacon addresses are distinct across all our challenges.

[15] Because our executable requires compilers and external libraries to build, it was produced on a networked machine. It is conceivable, but seems highly unlikely, that the resulting executable could contain malicious code that manages to exfiltrate secrets via the external media when we export the challenges and revealed secrets. Unfortunately, this risk is inherent to our setup, because we must copy data from the laptop at some point.

B Directory Structure and File Contents

B.1 Commitment Phase

The commitment phase corresponds to Step 1 of the cut-and-choose protocol from Section 3: we timestamp and publish all the challenge parameters, instances, and beacon addresses, but none of the underlying secrets.

We publish the commitment phase as a single archive named `rlwe-challenges-v1.tar.gz`, which contains many directories, each corresponding to a different challenge. For convenience, the Ring-LWE challenge directories are named according to the template[16]

`chall-id`*challID*`-`*type*`-m`*mval*`-q`*qval*`-l`*lval*`-`*annotation*

where

- *challID* is a globally unique non-negative integer (in decimal);
- *type* $\in \{\texttt{rlwec}, \texttt{rlwed}\}$ respectively indicates continuous or discretized Ring-LWE;
- *mval* is the cyclotomic index m;
- *qval* is the modulus q;
- *lval* is the number of Ring-LWE samples l;
- *annotation* is a descriptive string indicating the categories of the error width and estimated hardness (e.g., `grande-moderate`);

For example, the (hypothetical) directory `chall-id0003-rlwed-m128-q257-l100-short-easy` would contain challenge number 3, which is for discretized Ring-LWE over the 128th cyclotomic with modulus $q = 257$ and $l = 100$ samples, for a Gaussian parameter r from the "short" category, which we expect to be "easy" to solve.

Similarly, the Ring-LWR challenge directories are named according to the template

`chall-id`*challID*`-rlwr-m`*mval*`-q`*qval*`-p`*pval*`-l`*lval*`-`*annotation*

where *challID*, *mval*, *qval*, and *lval* are as above, and

- *pval* is the target rounding modulus p;
- *annotation* is a descriptive string indicating the estimated hardness category (e.g., `veryhard`)

Each challenge directory named *dirName* contains the following:

- A file *dirName*`.challenge`, which consists of a serialized **`message Challenge`** containing the parameters of the instantiation, the computed error bound (for Ring-LWE instantiations), the number of instances in the challenge, the beacon address for the cut-and-choose protocol, etc. (See Figure 2a.)

- Several files *dirName*`-`*instID*`.instance`, where *instID* is two upper-case hexadecimal digits uniquely identifying the instance within the challenge, starting from `00`. Each such file consists of a serialized **`message Instance`***Type*, where *Type* is as indicated by the challenge file. (See Figure 2b.)

See Appendix C for further details on the formats of the `.challenge` and `.instance` files.

[16] We stress that the file *contents* define the actual challenge data; the names are only for convenience and human readability.

B.2 Reveal Phase

The reveal phase corresponds to Step 3 of the cut-and-choose protocol from Section 3: for each challenge, we publish the secrets and PRG seeds underlying all but one of the instances, as indicated by the value of the randomness beacon at the "address" (i.e., beacon epoch and byte offset) specified in the challenge.

We publish a single archive having the same directory structure as in the commitment phase. For each instance file *instName*`.instance` whose secret should be revealed, we include a file *instName*`.secret` in the same directory, which consists of a serialized **`message Secret`**. (See Figure 2b.)

In addition to the instance secrets, for convenience the archive includes some additional files at the top level of the directory tree (i.e., not in any challenge folder):

- We include the original XML files for all the needed NIST beacon values; their format is detailed at `https://beacon.nist.gov/record/0.1/beacon-0.1.0.xsd`.
- We include the NIST certificate containing the public verification key under which the beacon values are digitally signed. This certificate is available at `https://beacon.nist.gov/certificate/beacon.cer`.

We remark that all these files are publicly available from the NIST beacon web site; we include them in our archives so that the challenges can be verified offline, or in the event that the NIST beacon becomes unavailable.

C Protocol Buffers Message Specifications

Our challenges are serialized using Google's language- and platform-neutral *protocol buffers* framework [Goo08]. Figure 2 gives the specifications for all the message types, which are available in the `.proto` files on the Ring-LWE challenges website [RLW16] and the $\Lambda\circ\lambda$ GitHub repository [CP16b]. These message specifications can be used to automatically generate parsers for our challenge files in most popular programming languages.

We point out that in the `Rq` and `Kq` message types, the coefficient arrays correspond to the *ordered* bases as implemented in $\Lambda\circ\lambda$, which are in "digit reversed" order. E.g., the powerful and tweaked decoding basis of the 16th cyclotomic ring is $1, X^4, X^2, X^6, X, X^5, X^3, X^7$. See [CP16a, Section C.1.1] for further details.

Figure 2: Protocol buffers message types.

```
message Challenge {
  required int32 challengeID  = 1;  // unique identifier of challenge
  required int32 numInstances = 2;  // number of instances in challenge
  required int64 beaconEpoch  = 3;  // NIST beacon epoch
  required int32 beaconOffset = 4;  // byte position of beacon value
  oneof params {                    // challenge type and parameters
    ContParams cparams        = 5;
    DiscParams dparams        = 6;
    RLWRParams rparams        = 7;
  }
}

message ContParams {                // continuous Ring-LWE parameters
  required int32 m            = 1;  // cyclotomic index m
  required int64 q            = 2;  // modulus q
  required double svar        = 3;  // squared Gaussian param v = r^2 (pre-tweak)
  required double bound       = 4;  // ||g · e||^2 bound (post-tweak)
  required int32 numSamples   = 5;  // number of samples per instance
}

message DiscParams {                // discrete Ring-LWE parameters; similar to ContParams
  required int32 m            = 1;
  required int64 q            = 2;
  required double svar        = 3;
  required int64 bound        = 4;
  required int32 numSamples   = 5;
}

message RLWRParams {                // Ring-LWR parameters; similar to ContParams
  required int32 m            = 1;
  required int64 q            = 2;
  required int64 p            = 3;  // rounding modulus p < q
  required int32 numSamples   = 4;
}
```

(a) Message types for challenges and their parameters.

```
message InstanceCont {                  // continuous Ring-LWE instance
  required int32 challengeID  = 1;  // ID of challenge this instance belongs to
  required int32 instanceID   = 2;  // ID of instance within the challenge
  required ContParams params  = 3;  // challenge params (for self-containment; should match)
  repeated SampleCont samples = 4;  // the Ring-LWE samples
}

message InstanceDisc {                  // discrete Ring-LWE instance; similar to InstanceCont
  required int32 challengeID  = 1;
  required int32 instanceID   = 2;
  required DiscParams params  = 3;
  repeated SampleDisc samples = 4;
}

message InstanceRLWR {                  // Ring-LWR instance; similar to InstanceCont
  required int32 challengeID  = 1;
  required int32 instanceID   = 2;
  required RLWRParams params  = 3;
  repeated SampleRLWR samples = 4;
}

message SampleCont {                    // continuous Ring-LWE sample
  required Rq a                 = 1;  // a ∈ R_q
  required Kq b                 = 2;  // b = s·a + e ∈ K_q for tweaked error e
}

message SampleDisc {                    // discrete Ring-LWE sample
  required Rq a                 = 1;  // a ∈ R_q
  required Rq b                 = 2;  // b = s·a + ⌊e⌉ ∈ R_q for tweaked e, discretized in dec. basis of R
}

message SampleRLWR {                    // Ring-LWR sample
  required Rq a                 = 1;  // a ∈ R_q
  required Rq b                 = 2;  // b = ⌊s·a⌉_p ∈ R_p, rounded in decoding basis of R
}

message Secret {                        // a secret for an Ring-LWE/LWR instance
  required int32 challengeID    = 1;  // ID of challenge this secret applies to
  required int32 instanceID     = 2;  // ID of instance this secret applies to
  required int32 m              = 3;  // cyclotomic index m of R
  required int64 q              = 4;  // modulus q
  required bytes seed           = 5;  // 256-bit CTR-DRBG-AES-128 entropy seed used to generate instance
  required Rq s                 = 6;  // the secret s ∈ R_q
}
```

(b) Message types for Ring-LWE/LWR samples and instances.

```
message Rq {                            // an element of R_q = R/qR
  required uint32 m             = 1;  // cyclotomic index m of R
  required uint64 q             = 2;  // modulus q
  repeated sint64 xs            = 3;  // n = φ(m) integral coefficients in decoding basis of R
}

message Kq {                            // an element of K_q = K/qR
  required uint32 m             = 1;  // cyclotomic index m of K
  required uint64 q             = 2;  // modulus q
  repeated double xs            = 3;  // n = φ(m) real coefficients in decoding basis of R
}
```

(c) Message types for ring and field elements modulo qR.

D Hardness Estimates

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
0	256	128	1.000	769	1.0160	toy	1.0124	96	$\leq$ 30
2	256	128	1.000	512	1.0152	toy	1.0125	121	42
4	256	128	2.000	3,329	1.0160	toy	1.0124	105	$\leq$ 30
6	256	128	2.000	1,024	1.0136	toy	1.0116	135	57
8	256	128	6.000	7,681	1.0135	toy	1.0117	162	56
10	256	128	6.000	8,192	1.0137	toy	1.0118	172	54
12	256	128	9.000	17,921	1.0138	toy	1.0119	177	52
14	256	128	9.000	32,768	1.0150	toy	1.0126	195	34
16	256	128	10.950	25,601	1.0138	toy	1.0120	187	51
18	256	128	10.950	32,768	1.0143	toy	1.0123	198	45
20	256	128	1.000	769	1.0160	toy	1.0124	96	$\leq$ 30
22	256	128	1.000	512	1.0152	toy	1.0125	121	42
24	256	128	2.000	3,329	1.0160	toy	1.0124	105	$\leq$ 30
26	256	128	2.000	1,024	1.0136	toy	1.0116	135	57
28	256	128	6.000	7,681	1.0135	toy	1.0117	162	56
30	256	128	6.000	8,192	1.0137	toy	1.0118	172	54
32	256	128	9.000	17,921	1.0138	toy	1.0119	177	52
34	256	128	9.000	32,768	1.0150	toy	1.0126	195	34
36	256	128	10.950	25,601	1.0138	toy	1.0120	187	51
38	256	128	10.950	32,768	1.0143	toy	1.0123	198	45
40	512	256	1.000	7,681	1.0102	easy	1.0095	201	94
42	512	256	1.000	512	1.0075	moderate	1.0074	196	152
44	512	256	2.000	7,681	1.0088	moderate	1.0084	234	121
46	512	256	2.000	2,048	1.0075	hard	1.0073	242	154
48	512	256	6.000	10,753	1.0071	hard	1.0070	295	167
50	512	256	6.000	16,384	1.0075	hard	1.0073	293	154
52	512	256	9.000	25,601	1.0072	hard	1.0071	318	162
54	512	256	9.000	32,768	1.0075	hard	1.0073	332	154
56	512	256	15.486	70,657	1.0073	hard	1.0072	352	159
58	512	256	15.486	131,072	1.0079	moderate	1.0077	334	142
60	512	256	1.000	7,681	1.0102	easy	1.0095	201	94
62	512	256	1.000	512	1.0075	moderate	1.0074	196	152
64	512	256	2.000	7,681	1.0088	moderate	1.0084	234	121
66	512	256	2.000	2,048	1.0075	hard	1.0073	242	154
68	512	256	6.000	10,753	1.0071	hard	1.0070	295	167
70	512	256	6.000	16,384	1.0075	hard	1.0073	293	154
72	512	256	9.000	25,601	1.0072	hard	1.0071	318	162
74	512	256	9.000	32,768	1.0075	hard	1.0073	332	154
76	512	256	15.486	70,657	1.0073	hard	1.0072	352	159
78	512	256	15.486	131,072	1.0079	moderate	1.0077	334	142
80	1,024	512	9.000	37,889	1.0038	very hard	1.0041	620	382
82	1,024	512	21.901	202,753	1.0039	very hard	1.0042	682	376
84	2,048	1,024	9.000	59,393	1.0020	very hard	1.0023	1,121	838

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
86	2,048	1,024	30.972	638,977	1.0021	very hard	1.0024	1,321	823
88	4,096	2,048	9.000	86,017	1.0010	very hard	1.0013	2,121	1,795
90	4,096	2,048	43.801	1,720,321	1.0011	very hard	1.0013	2,621	1,779
92	8,192	4,096	9.000	114,689	1.0005	very hard	1.0007	4,017	3,799
94	8,192	4,096	61.945	5,234,689	1.0006	very hard	1.0007	5,141	3,727
96	243	162	0.931	487	1.0121	toy	1.0107	132	72
98	243	162	0.931	512	1.0122	toy	1.0108	133	71
100	243	162	0.931	729	1.0128	toy	1.0111	124	65
102	243	162	1.861	1,459	1.0115	toy	1.0104	167	78
104	243	162	1.861	1,024	1.0110	easy	1.0099	160	86
106	243	162	1.861	2,187	1.0122	toy	1.0108	161	70
108	243	162	5.584	8,263	1.0110	easy	1.0100	207	84
110	243	162	5.584	8,192	1.0110	easy	1.0100	216	84
112	243	162	5.584	19,683	1.0123	toy	1.0111	205	66
114	243	162	8.375	17,011	1.0110	easy	1.0100	208	84
116	243	162	8.375	32,768	1.0120	toy	1.0108	210	70
118	243	162	8.375	19,683	1.0112	toy	1.0103	231	80
120	243	162	11.464	32,563	1.0112	toy	1.0102	223	81
122	243	162	11.464	32,768	1.0112	toy	1.0102	220	81
124	243	162	11.464	59,049	1.0121	toy	1.0109	216	69
126	243	162	0.931	487	1.0121	toy	1.0107	132	72
128	243	162	0.931	512	1.0122	toy	1.0108	133	71
130	243	162	0.931	729	1.0128	toy	1.0111	124	65
132	243	162	1.861	1,459	1.0115	toy	1.0104	167	78
134	243	162	1.861	1,024	1.0110	easy	1.0099	160	86
136	243	162	1.861	2,187	1.0122	toy	1.0108	161	70
138	243	162	5.584	8,263	1.0110	easy	1.0100	207	84
140	243	162	5.584	8,192	1.0110	easy	1.0100	216	84
142	243	162	5.584	19,683	1.0123	toy	1.0111	205	66
144	243	162	8.375	17,011	1.0110	easy	1.0100	208	84
146	243	162	8.375	32,768	1.0120	toy	1.0108	210	70
148	243	162	8.375	19,683	1.0112	toy	1.0103	231	80
150	243	162	11.464	32,563	1.0112	toy	1.0102	223	81
152	243	162	11.464	32,768	1.0112	toy	1.0102	220	81
154	243	162	11.464	59,049	1.0121	toy	1.0109	216	69
156	625	500	8.229	28,751	1.0038	very hard	1.0041	611	377
158	625	500	19.788	191,251	1.0040	very hard	1.0043	644	359
160	3,360	768	7.033	30,241	1.0026	very hard	1.0030	853	610
162	3,360	768	20.960	305,761	1.0027	very hard	1.0030	988	584
164	500	200	0.914	3,001	1.0121	toy	1.0110	179	68
166	500	200	0.914	512	1.0099	easy	1.0092	165	101
168	500	200	0.914	500	1.0099	easy	1.0092	149	102
170	500	200	1.829	3,001	1.0103	easy	1.0095	193	94
172	500	200	1.829	2,048	1.0098	easy	1.0092	206	102
174	500	200	1.829	1,600	1.0095	moderate	1.0089	193	108
176	500	200	5.486	9,001	1.0090	moderate	1.0086	241	116

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
178	500	200	5.486	16,384	1.0098	easy	1.0092	229	102
180	500	200	5.486	10,000	1.0092	moderate	1.0087	251	113
182	500	200	8.229	19,501	1.0091	moderate	1.0087	269	114
184	500	200	8.229	32,768	1.0097	easy	1.0092	252	102
186	500	200	8.229	20,000	1.0091	moderate	1.0087	250	114
188	500	200	12.515	44,501	1.0092	moderate	1.0087	263	112
190	500	200	12.515	65,536	1.0097	easy	1.0092	285	102
192	500	200	12.515	50,000	1.0094	moderate	1.0089	265	109
194	500	200	0.914	3,001	1.0121	toy	1.0110	179	68
196	500	200	0.914	512	1.0099	easy	1.0092	165	101
198	500	200	0.914	500	1.0099	easy	1.0092	149	102
200	500	200	1.829	3,001	1.0103	easy	1.0095	193	94
202	500	200	1.829	2,048	1.0098	easy	1.0092	206	102
204	500	200	1.829	1,600	1.0095	moderate	1.0089	193	108
206	500	200	5.486	9,001	1.0090	moderate	1.0086	241	116
208	500	200	5.486	16,384	1.0098	easy	1.0092	229	102
210	500	200	5.486	10,000	1.0092	moderate	1.0087	251	113
212	500	200	8.229	19,501	1.0091	moderate	1.0087	269	114
214	500	200	8.229	32,768	1.0097	easy	1.0092	252	102
216	500	200	8.229	20,000	1.0091	moderate	1.0087	250	114
218	500	200	12.515	44,501	1.0092	moderate	1.0087	263	112
220	500	200	12.515	65,536	1.0097	easy	1.0092	285	102
222	500	200	12.515	50,000	1.0094	moderate	1.0089	265	109
224	1,155	480	0.727	2,311	1.0052	hard	1.0054	311	256
226	1,155	480	0.727	4,096	1.0055	hard	1.0056	333	238
228	1,155	480	0.727	2,401	1.0052	hard	1.0054	335	254
230	1,155	480	1.454	4,621	1.0047	very hard	1.0050	393	286
232	1,155	480	1.454	4,096	1.0047	very hard	1.0049	383	291
234	1,155	480	1.454	3,465	1.0046	very hard	1.0049	375	298
236	1,155	480	4.362	18,481	1.0043	very hard	1.0046	509	321
238	1,155	480	4.362	16,384	1.0043	very hard	1.0046	496	327
240	1,155	480	4.362	12,705	1.0042	very hard	1.0045	521	339
242	1,155	480	6.543	32,341	1.0043	very hard	1.0045	530	331
244	1,155	480	6.543	32,768	1.0043	very hard	1.0045	543	330
246	1,155	480	6.543	27,783	1.0042	very hard	1.0045	551	338
248	1,155	480	15.416	164,011	1.0043	very hard	1.0046	597	327
250	1,155	480	15.416	262,144	1.0046	very hard	1.0048	632	305
252	1,155	480	15.416	164,025	1.0043	very hard	1.0046	597	327
254	1,155	480	0.727	2,311	1.0052	hard	1.0054	311	256
256	1,155	480	0.727	4,096	1.0055	hard	1.0056	333	238
258	1,155	480	0.727	2,401	1.0052	hard	1.0054	335	254
260	1,155	480	1.454	4,621	1.0047	very hard	1.0050	393	286
262	1,155	480	1.454	4,096	1.0047	very hard	1.0049	383	291
264	1,155	480	1.454	3,465	1.0046	very hard	1.0049	375	298
266	1,155	480	4.362	18,481	1.0043	very hard	1.0046	509	321
268	1,155	480	4.362	16,384	1.0043	very hard	1.0046	496	327

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
270	1,155	480	4.362	12,705	1.0042	very hard	1.0045	521	339
272	1,155	480	6.543	32,341	1.0043	very hard	1.0045	530	331
274	1,155	480	6.543	32,768	1.0043	very hard	1.0045	543	330
276	1,155	480	6.543	27,783	1.0042	very hard	1.0045	551	338
278	1,155	480	15.416	164,011	1.0043	very hard	1.0046	597	327
280	1,155	480	15.416	262,144	1.0046	very hard	1.0048	632	305
282	1,155	480	15.416	164,025	1.0043	very hard	1.0046	597	327
284	179	178	0.988	3,581	1.0137	toy	1.0119	149	52
286	179	178	0.988	2,048	1.0129	toy	1.0114	154	61
288	179	178	0.988	32,041	1.0168	toy	1.0124	83	$\leq$ 30
290	179	178	1.977	3,581	1.0116	toy	1.0105	176	76
292	179	178	1.977	4,096	1.0118	toy	1.0107	191	73
294	179	178	1.977	32,041	1.0147	toy	1.0124	171	$\leq$ 30
296	179	178	5.930	8,951	1.0100	easy	1.0093	212	100
298	179	178	5.930	16,384	1.0108	easy	1.0099	215	86
300	179	178	5.930	32,041	1.0117	toy	1.0107	235	72
302	179	178	8.895	20,407	1.0101	easy	1.0094	226	97
304	179	178	8.895	32,768	1.0108	easy	1.0099	231	86
306	179	178	8.895	32,041	1.0107	easy	1.0099	250	86
308	179	178	12.762	40,813	1.0102	easy	1.0095	238	95
310	179	178	12.762	65,536	1.0109	easy	1.0100	249	84
312	179	178	12.762	5,735,339	1.0171	toy	1.0124	123	$\leq$ 30
314	179	178	0.988	3,581	1.0137	toy	1.0119	149	52
316	179	178	0.988	2,048	1.0129	toy	1.0114	154	61
318	179	178	0.988	32,041	1.0168	toy	1.0124	83	$\leq$ 30
320	179	178	1.977	3,581	1.0116	toy	1.0105	176	76
322	179	178	1.977	4,096	1.0118	toy	1.0107	191	73
324	179	178	1.977	32,041	1.0147	toy	1.0124	171	$\leq$ 30
326	179	178	5.930	8,951	1.0100	easy	1.0093	212	100
328	179	178	5.930	16,384	1.0108	easy	1.0099	215	86
330	179	178	5.930	32,041	1.0117	toy	1.0107	235	72
332	179	178	8.895	20,407	1.0101	easy	1.0094	226	97
334	179	178	8.895	32,768	1.0108	easy	1.0099	231	86
336	179	178	8.895	32,041	1.0107	easy	1.0099	250	86
338	179	178	12.762	40,813	1.0102	easy	1.0095	238	95
340	179	178	12.762	65,536	1.0109	easy	1.0100	249	84
342	179	178	12.762	5,735,339	1.0171	toy	1.0124	123	$\leq$ 30
344	257	256	0.991	9,767	1.0104	easy	1.0098	222	89
346	257	256	0.991	4,096	1.0096	easy	1.0091	218	104
348	257	256	0.991	66,049	1.0123	toy	1.0113	225	62
350	257	256	1.982	9,767	1.0090	moderate	1.0086	244	115
352	257	256	1.982	4,096	1.0082	moderate	1.0079	238	135
354	257	256	1.982	66,049	1.0109	easy	1.0102	255	81
356	257	256	5.947	13,879	1.0073	hard	1.0072	305	158
358	257	256	5.947	16,384	1.0075	hard	1.0074	306	153
360	257	256	5.947	66,049	1.0089	moderate	1.0085	301	118

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
362	257	256	8.920	23,131	1.0071	hard	1.0071	312	165
364	257	256	8.920	32,768	1.0075	hard	1.0073	317	154
366	257	256	8.920	66,049	1.0081	moderate	1.0079	314	135
368	257	256	15.349	74,017	1.0074	hard	1.0073	356	157
370	257	256	15.349	131,072	1.0079	moderate	1.0077	351	141
372	257	256	0.991	9,767	1.0104	easy	1.0098	222	89
374	257	256	0.991	4,096	1.0096	easy	1.0091	218	104
376	257	256	0.991	66,049	1.0123	toy	1.0113	225	62
378	257	256	1.982	9,767	1.0090	moderate	1.0086	244	115
380	257	256	1.982	4,096	1.0082	moderate	1.0079	238	135
382	257	256	1.982	66,049	1.0109	easy	1.0102	255	81
384	257	256	5.947	13,879	1.0073	hard	1.0072	305	158
386	257	256	5.947	16,384	1.0075	hard	1.0074	306	153
388	257	256	5.947	66,049	1.0089	moderate	1.0085	301	118
390	257	256	8.920	23,131	1.0071	hard	1.0071	312	165
392	257	256	8.920	32,768	1.0075	hard	1.0073	317	154
394	257	256	8.920	66,049	1.0081	moderate	1.0079	314	135
396	257	256	15.349	74,017	1.0074	hard	1.0073	356	157
398	257	256	15.349	131,072	1.0079	moderate	1.0077	351	141
400	797	796	8.968	44,633	1.0025	very hard	1.0028	886	643
402	797	796	27.210	401,689	1.0026	very hard	1.0029	1,042	625
404	256	128	141.295	3,754,241	1.0154	toy	1.0124	182	$\leq$ 30
406	256	128	141.295	4,194,304	1.0156	toy	1.0124	174	$\leq$ 30
408	256	128	302.375	18,684,161	1.0162	toy	1.0124	166	$\leq$ 30
410	512	256	232.482	16,470,529	1.0083	moderate	1.0081	438	130
412	512	256	502.450	74,613,761	1.0086	moderate	1.0083	449	123
414	1,024	512	835.832	289,001,473	1.0045	very hard	1.0047	933	311
416	2,048	1,024	1,391.758	1,159,182,337	1.0024	very hard	1.0026	1,740	712
418	243	162	155.683	6,112,423	1.0126	toy	1.0115	275	60
420	243	162	155.683	8,388,608	1.0131	toy	1.0118	279	54
422	243	162	155.683	6,112,422	1.0126	toy	1.0115	275	60
424	243	162	334.353	25,218,541	1.0130	toy	1.0118	296	55
426	625	500	750.988	241,965,001	1.0046	very hard	1.0048	874	302
428	3,360	768	880.048	476,757,121	1.0032	very hard	1.0034	1,337	504
430	500	200	177.953	8,794,501	1.0104	easy	1.0098	343	89
432	500	200	177.953	8,791,500	1.0104	easy	1.0098	343	89
434	500	200	383.329	37,996,001	1.0107	easy	1.0100	349	84
436	1,155	480	266.103	41,817,931	1.0048	very hard	1.0049	777	291
438	1,155	480	579.489	212,466,871	1.0050	very hard	1.0051	810	276
440	179	178	176.904	8,382,929	1.0116	toy	1.0108	325	71
442	179	178	176.904	8,388,608	1.0116	toy	1.0108	325	71
444	179	178	176.904	8,382,033	1.0116	toy	1.0108	325	71
446	179	178	380.444	37,250,617	1.0120	toy	1.0111	316	66
448	257	256	230.425	15,802,417	1.0083	moderate	1.0080	428	131
450	257	256	230.425	15,792,907	1.0083	moderate	1.0080	428	131
452	257	256	498.003	72,720,721	1.0086	moderate	1.0083	457	123

Table 1: Hardness estimates for our *continuous* Ring-LWE challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: r' is the rescaled error parameter (Section 5.1), δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3). Hardness estimates for our *discrete* Ring-LWE challenges (odd challenge IDs, with parameters identical to the preceding even challenge ID) are essentially the same, but may be slightly larger due to the extra round-off error.

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	r'	q	δ	Qualitative	κ	Dimension d	Block size
454	797	796	1,152.130	741,587,779	1.0030	very hard	1.0033	1,360	527

Table 2: Hardness estimates for our Ring-LWR challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3).

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	q	p	δ	Qualitative	κ	Dimension d	Block size
456	32	16	97	2	1.0100	easy	1.0081	75	$\leq$ 30
457	32	16	32	2	1.0133	toy	1.0092	60	101
458	32	16	105	7	1.0299	toy	1.0124	33	$\leq$ 30
459	64	32	193	2	1.0043	very hard	1.0053	141	263
460	64	32	16	2	1.0083	moderate	1.0075	72	150
461	64	32	105	7	1.0148	toy	1.0108	82	71
462	128	64	257	2	1.0021	very hard	1.0034	250	497
463	128	64	16	2	1.0041	very hard	1.0052	112	272
464	128	64	105	7	1.0074	hard	1.0071	128	162
465	256	128	257	2	1.0010	very hard	1.0022	427	904
466	256	128	16	2	1.0021	very hard	1.0034	189	493
467	256	128	105	7	1.0037	very hard	1.0045	232	335
468	27	18	109	2	1.0087	moderate	1.0075	82	147
469	27	18	32	2	1.0118	toy	1.0087	63	112
470	27	18	105	7	1.0265	toy	1.0124	41	$\leq$ 30
471	27	18	81	3	1.0142	toy	1.0098	66	88
472	81	54	163	2	1.0027	very hard	1.0040	200	399
473	81	54	16	2	1.0049	very hard	1.0057	101	235
474	81	54	105	7	1.0088	moderate	1.0079	114	134
475	81	54	27	3	1.0063	hard	1.0065	106	190
476	243	162	487	2	1.0007	very hard	1.0018	578	1,221
477	243	162	16	2	1.0016	very hard	1.0030	215	605
478	243	162	105	7	1.0029	very hard	1.0038	274	426
479	243	162	27	3	1.0021	very hard	1.0033	227	527
480	25	20	101	2	1.0079	moderate	1.0072	88	158
481	25	20	32	2	1.0106	easy	1.0083	65	123
482	25	20	105	7	1.0239	toy	1.0124	57	$\leq$ 30
483	25	20	125	5	1.0180	toy	1.0115	65	60
484	125	100	251	2	1.0013	very hard	1.0026	353	727
485	125	100	16	2	1.0026	very hard	1.0040	153	399
486	125	100	105	7	1.0047	very hard	1.0053	184	260
487	125	100	25	5	1.0054	hard	1.0058	121	225
488	49	42	197	2	1.0033	very hard	1.0045	177	332
489	49	42	16	2	1.0063	hard	1.0065	90	189

Table 2: Hardness estimates for our Ring-LWR challenges, in terms of approximate root-Hermite factors and smallest BKZ block size required to solve them: δ is the root-Hermite factor (Section 5.2), and κ is the GSA factor (Section 5.3).

					Hermite Factor		BKZ		
ID	m	$\varphi(m)$	q	p	δ	Qualitative	κ	Dimension d	Block size
490	49	42	105	7	1.0113	toy	1.0093	97	100
491	49	42	49	7	1.0135	toy	1.0103	77	80
492	84	24	337	2	1.0052	hard	1.0058	131	225
493	84	24	32	2	1.0088	moderate	1.0077	76	143
494	84	24	105	7	1.0198	toy	1.0123	70	46
495	84	24	42	2	1.0082	moderate	1.0074	81	153
496	105	48	211	2	1.0028	very hard	1.0041	190	377
497	105	48	16	2	1.0055	hard	1.0061	97	212
498	105	48	105	7	1.0099	easy	1.0085	107	117
499	105	48	105	3	1.0050	hard	1.0056	141	237
500	60	16	61	2	1.0112	toy	1.0085	69	118
501	60	16	32	2	1.0133	toy	1.0092	60	101
502	60	16	105	7	1.0299	toy	1.0124	33	$\leq$ 30
503	60	16	900	2	1.0067	hard	1.0066	113	184
504	100	40	101	2	1.0040	very hard	1.0050	151	283
505	100	40	16	2	1.0066	hard	1.0067	81	182
506	100	40	105	7	1.0119	toy	1.0095	100	94
507	100	40	100	2	1.0040	very hard	1.0050	144	283
508	29	28	59	2	1.0064	hard	1.0065	99	188
509	29	28	32	2	1.0076	moderate	1.0071	84	163
510	29	28	105	7	1.0170	toy	1.0115	75	59
511	29	28	841	29	1.0258	toy	1.0124	42	$\leq$ 30
512	23	22	47	2	1.0087	moderate	1.0076	78	146
513	23	22	32	2	1.0096	easy	1.0080	70	133
514	23	22	105	7	1.0217	toy	1.0126	64	39
515	23	22	529	23	1.0317	toy	1.0124	30	$\leq$ 30