Variants of LWE: Attacks, Reductions, and a Construction

Amit Deo

Thesis submitted to the University of London
for the degree of Doctor of Philosophy

Information Security Group
Royal Holloway, University of London

2019

Declaration

These doctoral studies were conducted under the supervision of Professor Martin R. Albrecht and Professor Kenneth G. Paterson.

The work presented in this thesis is the result of original research conducted by myself in collaboration with others, whilst enrolled in the School of Mathematics and Information Security as a candidate for the degree of Doctor of Philosophy. This work has not been submitted for any other degree of award in any other university or educational establishment.

Amit Deo
October, 2019

Acknowledgements

First of all I'd like to thank my supervisors Martin Albrecht and Kenny Paterson. Without them, the writing of this thesis would not have been possible. In particular, I'd like to thank Martin for giving me so much of his time in the last three years and for his seemingly limitless patience, positivity and cryptographic insight. I can safely say that I have learnt a lot and have a much more positive attitude towards the challenges faced in research thanks to Martin. As for Kenny, I thank him for the belief he has shown in me throughout this PhD process and his willingness to share research ideas. I also thank my other coauthors Lorenzo Cavallaro, Santanu Dash, Alex Davidson, Ela Lee, Keith M. Martin, Nigel P. Smart, Guillermo Suarez-Tangil, Volodya Vovk and the entire "Estimate all the {LWE, NTRU schemes!" team. In addition, I thank the EPSRC and the UK government for their financial support as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/K035584/1).

During the last four years, I have met a lot of wonderful people who have made my time at Royal Holloway a pleasure. In particular, I'd like to thank all of my former "253" office-mates for making the department a friendly place to work in and for the much-needed daily coffee breaks. A special thanks goes to all of the CDT/PhD friends I have made on this journey, especially Ben, for the kindness and good humour that they've shown to me from start to finish. To Lydia, I thank you for always being a source of love and happiness in my life. And finally, to my family (and in particular my parents without whom I would have never been able to go down this path): thank you for your unconditional support and guidance through the ups and downs of this PhD process and my entire life.

Publications

The contents of this thesis are based on the following publications, the last of which is currently in submission.

1. Martin R. Albrecht, Amit Deo, and Kenneth G. Paterson. Cold Boot Attacks on Ring and Module LWE Keys Under the NTT. In *IACR TCHES,* 2018(3):173-213, 2019.

2. Martin R. Albrecht and Amit Deo. Large Modulus Ring-LWE $\geq$ Module-LWE. In *ASIACRYPT 2017 Part I,* volume 10624 of *LNCS,* pages 267-296. Springer, Heidelberg.

3. Martin R. Albrecht, Alex Davidson, Amit Deo, and Nigel P. Smart. Round-Optimal Verifiable Pseudorandom Functions from Ideal Lattices. *In submission.*

Abstract

In this thesis, we cover various topics in lattice-based cryptography. This means that the scope of this thesis is wider than it is deep, with contributions in the areas of cryptanalysis, hardness assumptions and cryptographic constructions. For the cryptanalytic contribution, we present techniques that allow an attacker to recover a secret key when given a leakage profile practically achievable by means of a cold boot attack. In particular, we focus on the case where a number theoretic transform (NTT) is used for key storage. This is a common choice made in efficient implementations of lattice-based cryptography. In addition to describing the attack in detail, we run experiments attesting to the practicality and efficiency of our methods using realistic parameters. Our techniques rely heavily on the divide and conquer structure of the NTT and lattice basis reduction.

The second main contribution considers reductions between variants of the standard learning with errors (LWE) problem. The variants of interest are Ring-LWE (RLWE) and Module-LWE (MLWE). The content presented in this thesis improves on our original paper published at Asiacrypt 2017 by way of an improved analysis of the reduction. In particular, a main result in our original paper states that for power-of-two cyclotomic rings of dimension n, there is a reduction *from* the MLWE problem in module rank d, modulus q and error rate α *to* the RLWE problem in modulus q^d and error rate $\alpha' = n^2\sqrt{d} \cdot \alpha$. However, this thesis shows that a smaller error rate of $\alpha' = n^{1/2+c}\sqrt{d} \cdot \alpha$ is possible for any constant $c > 0$. In addition, the original RLWE to RLWE dimensions that halve the ring dimension while squaring the modulus are improved upon by way of shrinking the growth factor in error rate.

Our final contribution is to construct a verifiable oblivious pseudorandom function (VOPRF) protocol whose security is based on lattice assumptions. To our knowledge, this is the first construction of a post-quantum secure VOPRF. A VOPRF allows a client to obtain a pseudorandom function evaluation on a point of its choice using a server's secret key. Importantly, the server does not learn anything about the client's input and the client does not learn anything about the server's key. Our protocol manages to achieve security against adversaries that may deviate arbitrarily from the protocol and consists of a single round of interaction. Our protocol should be interpreted as showing the feasibility of round-optimal VOPRFs from lattice assumptions rather than being a practical construction.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Contents

As the development of large-scale quantum computation progresses, new solutions to cryptographic problems are being developed. In particular, these new solutions are intended to resist attacks carried out on a quantum computer. Unfortunately, all cryptographic constructions that are secure assuming the hardness of the RSA [120] or discrete logarithm problems are insecure when considering an adversary that can execute Shor's quantum algorithm [129]. Since these problems are central in the design of a large proportion of cryptography, the cryptographic community has been investigating alternative constructions/hardness assumptions that remain secure /valid with respect to quantum adversaries. In addition to Shor's algorithm, quantum algorithms such as Grover's algorithm [63], Simon's algorithm [130], and the HHL algorithm [92] have all been considered for quantum cryptanalysis [80, 28, 39]. Cryptography that remains secure against quantum adversaries is often called *post-quantum cryptography*. The importance of designing post-quantum cryptography is highlighted by the efforts of NIST to standardise post-quantum public-key encryption, key encapsulation mechanisms and signature schemes [104].

Research areas in post-quantum cryptography are numerous. Amongst these areas are lattice-based cryptography, multi-variate cryptography, code-based cryptography, hash-based cryptography and isogeny-based cryptography. In this thesis, we will focus solely on lattice-based cryptography, which appears to be amongst the most popular research areas in post-quantum cryptography. In fact, this thesis is fairly broad in its content and covers different topics within lattice-based cryptography. Therefore, this introduction will contain brief overviews of:

- popular hardness assumptions,
- cryptanalysis of lattice-based schemes,
- and advanced cryptographic constructions.

1.1 LWE and its Variants

The learning with errors problem (LWE) [118] is perhaps the most fundamental assumption in lattice-based cryptography. Informally, the (decisional) LWE problem asks an adversary to distinguish between $(\boldsymbol{a}_i, b_i := \boldsymbol{a}_i \cdot \boldsymbol{s} + e_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ and $(\boldsymbol{a}_i, u_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ for $i = 1, \ldots, m$ where $\boldsymbol{a}_i$ is uniform over $\mathbb{Z}_q^n$; u_i is uniform over $\mathbb{Z}_q$; the "noise" terms $e_i \in \mathbb{Z}$ are integers small in absolute value; and $\boldsymbol{s} \in \mathbb{Z}_q^n$ is a secret drawn from a uniform distribution. Typically, the noise terms are sampled from a discretised Gaussian distribution with standard deviation $\alpha \cdot q$. We refer to the parameter α as the error rate. It is believed that the LWE problem cannot be solved efficiently by a quantum algorithm. This belief is *partially* backed up by the result of Regev stating that the existence of an efficient quantum algorithm solving LWE implies the existence of an efficient quantum algorithm solving certain lattice problems *in the worst-case*. Therefore, according to the hypothesis that these worst-case lattice problems are not efficiently solvable with respect to quantum algorithms, LWE is not efficiently solvable; even for quantum adversaries. For formal statements, see Section 2.

One can view the LWE problem as the task of distinguishing random *noisy* inner products from uniform. From this informal interpretation, one can straightforwardly imagine replacing the finite set of vectors $\mathbb{Z}_q^n$ with other finite spaces and swapping the inner product/addition operations with alternative analogous operations to obtain *variants* of the LWE problem. One example would be to consider the ring of integers R of some algebraic number field. We might then swap the vectors in $\mathbb{Z}_q^n$ with $R_q := R/qR$ and replace inner-products and additions with multiplication and addition in R_q. More concretely, this description gives rise to the Ring-LWE (RLWE) [97] and informally asks an adversary to distinguish between $(a_i, b_i := a_i \cdot s + e_i) \in R_q \times R_q$ and $(a_i, u_i) \in R_q \times R_q$ for $i = 1, \ldots, m$ where a_i and u_i are uniform over R_q; the "noise" terms $e_i \in R$ are polynomials with small

coefficients; and $s \in R_q$ is a secret drawn from a uniform distribution. The RLWE problem can be interpreted as asking an adversary to distinguish random noisy ring products from uniform. The advantage of using RLWE over LWE is that a RLWE sample essentially contains n structured LWE samples (one in each coefficient) assuming R is of degree n. Roughly speaking, this results in RLWE public keys being a factor of n smaller than LWE keys at the cost of introducing algebraic structure.

The above recipe for producing variants of the LWE problem has lead to a proliferation of LWE-like problems. Amongst these are Module-LWE (MLWE) [81], Polynomial-LWE (PLWE) [134], Order-LWE (OLWE) [17] and Middle-Product-LWE (MPLWE) [121]. Similarly to RLWE, each of these variants have their pros and cons. For example, MLWE asks an adversary to distinguish between $(\boldsymbol{a}_i, b_i := \boldsymbol{a}_i \cdot \boldsymbol{s} + e_i) \in R_q^d \times R_q$ and $(\boldsymbol{a}_i, u_i) \in R_q^d \times R_q$ for $i = 1, \ldots, m$ where $\boldsymbol{a}_i \in R_q^d$ and $u_i \in R_q$ are uniform; the "noise" terms $e_i \in R$ are polynomials with small coefficients; and $\boldsymbol{s} \in R_q^d$ is a secret drawn from a known distribution. Note that the multiplication here is analogous to an inner product of vectors. The advantage of MLWE is that one can very easily increase security in an implementation by changing the value of d. Note that the ring R (or q) need not change as is the case for increasing security for RLWE. Therefore, we can stick to a fixed ring (perhaps one that allows efficient multiplication) once and for all, even if security levels need to be increased.

Often, whenever a new LWE variant is introduced it is accompanied by a reduction from a presumed hard lattice problem over a particular class of lattices (e.g. ideal lattices [97] or module lattices [81]). Additionally, direct reductions between the variants also allow for a better understanding of the relative hardness of different variants under certain parameter settings. For example, there are results showing that MPLWE is at least as hard as PLWE [121], and that PLWE is at least as hard as RLWE [122] in particular parameter settings. In addition, recent research simplifies and improves on some reductions from RLWE to other variants by presenting a very general algebraic framework [109]. For a more technical summary of reductions between variants of LWE, see Section 4.5.

Contribution: Reductions between MLWE and RLWE One of the contributions of this thesis (Chapter 4) relates to reductions from MLWE to RLWE. The

results presented in this thesis are an improvement over our original publication that appeared at Asiacrypt 2017 [6]. The improvements are a consequence of a better analysis of the reduction. All of our reductions apply to MLWE/RLWE in normal form where the secret is drawn according to the error distribution. Note that it has been shown that this method of sampling the secret makes MLWE/RLWE no easier [98, 81] than the case where secrets are chosen uniformly. Consider R to be a cyclotomic ring of integers with a power-of-two dimension n. Put informally, our main result states that the MLWE problem in rank d and modulus q over ring R is at least as hard as the RLWE problem in modulus q^d and ring R. It should be noted that the RLWE error rate is a factor of $n^{\frac{1}{2}+c} \cdot \sqrt{d}$ larger than the MLWE error rate for *any* constant $c > 0$. This blow-up factor improves on the factor of $n^2 \cdot \sqrt{d}$ reported in our original publication. The methodology of the reduction and analysis is heavily influenced by that of [32]. Major emphasis in the analysis is put on ensuring that the resulting error distributions remain statistically close to a Gaussian (which is usually considered as the standard error distribution as it enables reductions from lattice problems). Our reduction provides a clear account of the relative hardness of MLWE and RLWE, stating that RLWE in modulus q^d is at least as hard as MLWE in rank d and modulus q keeping the ring fixed.

We can also combine our aforementioned reduction from MLWE to RLWE with the recent results of Peikert and Pepin [109] to obtain a result relating the hardness of RLWE in different dimensions. We emphasise now that the following discussion holds for power-of-two cyclotomic rings. In particular, the work of Peikert and Pepin shows the existence of a reduction from RLWE in dimension n to MLWE in module rank 2 with underlying ring dimension $n/2$. Combining this reduction with our MLWE to RLWE reduction implies a dimension reducing reduction from RLWE to RLWE. A more precise account of the parameters involved follows. Let R' be a cyclotomic ring with power-of-two dimension $n' < n$ such that $n' > 1$. Using this notation, we combine our reduction with the aforementioned recent work to show existence of a reduction from RLWE in ring R, modulus q to RLWE in ring R', modulus $q^{n/n'}$. The accompanying growth factor in the error rate turns out to be $(n/n')^{3/2} \cdot (n')^{\frac{1}{2}+c}$ for any constant $c > 0$. As an example, taking $n' = n/2$ yields a growth in modulus from q to q^2 and a polynomial growth in error rate of roughly $n^{1/2+c}$ (ignoring constants). For $n' = n/2$, our original publication only managed to show a growth factor $n^{9/4}$ for *search* variants of RLWE. Therefore, the result in this

thesis extends our original result by allowing for consideration of *decisional* RLWE *and* providing a smaller growth in the error rate. Our RLWE to RLWE reduction implies the hardness of RLWE for very small ring dimensions and polynomial error rates provided that the modulus is chosen large enough.

1.2 Cryptanalysis

Above, we have mentioned reductions from lattice problems to LWE. These are often cited as providing theoretical evidence for the hardness of LWE. Unfortunately, these reductions cannot be used for concrete parameter selection as the complexity of solving *worst-case* lattice problems is not well understood. A seemingly more reasonable view is to consider LWE problems as hardness assumptions themselves, rather than relying on the worst-case hardness of lattice problems. The disadvantage of doing this is that LWE is a relatively new problem, whereas lattice problems have been around for many more years. Nonetheless, cryptanalysts are yet to contradict the hardness of LWE. Interestingly, the best attacks for standard cryptographic parameters on LWE correspond to the best RLWE and MLWE attacks with standard parameters. Put differently, the best way to attack RLWE/MLWE with standard parameters is to consider samples as multiple LWE samples (ignoring structure) and run the best known attacks on LWE.

The most effective algorithms for solving LWE include the primal [88, 15, 10, 8] and dual attacks [102, 4]. Essentially, both of these attacks formulate a reduction from LWE to lattice problems and then attempt to solve the resulting lattice problem via concrete methods. For example, one form of the primal attack transforms solving LWE to finding the shortest vector in a particular lattice. According to current knowledge, the best way to find this short vector is to run the BKZ [40] algorithm. In turn, the BKZ algorithm runs by making *multiple* calls to an oracle that finds short vectors in lattices of relatively small dimension (known as the block size). We note that the block size is carefully chosen to be large enough to ensure a successful attack on LWE, while bearing in mind that choosing larger block sizes leads to longer running times. In order to estimate the running time of the BKZ algorithm, we need to calculate both the number of oracle calls and the running time of each oracle call at the chosen block size. Unfortunately, neither of these quantities are

easily bounded and estimates vary between different cryptanalyses. For example, the number of oracle calls is sometimes conservatively assumed to be one [10] and in other cases assumed to be 8 times the dimension of the larger lattice [4]. In addition, the cost of an oracle call depends on the methods used to instantiate the oracle. The two main methods are sieving and enumeration. Once again, the costs of sieving/enumeration are not easily predicted so heuristics/empirical data-fitting is used to provide security estimates [40, 9, 103, 68]. All models predict a running time exponential in the block size. In addition, there have been many works investigating the quantum cost of the oracle call in BKZ [79, 80, 11, 78]. For a complete list of the attack cost models used in the lattice-based submissions to NIST's post-quantum standardisation process, see [5].

The cryptanalysis described above is used to set concrete parameters, and usually arises from a standard security notion (e.g. CPA, CCA, EUF-CMA security). However, one may consider a more general adversarial setting where partial leakage of secret key material occurs by way of a side-channel. This alternative setting may lead to unforeseen attacks not captured by standard security proofs. As we describe next, cold boot attacks are one such class of side-channel attack based on a particular leakage scenario.

Cold boot attacks were introduced and studied in the seminal work of Halderman et al. [65]. Briefly, cold boot attacks rely on the fact that bits in RAM retain their value for some time after power is cut. In order to preserve the value for longer, memory can be cooled to extreme temperatures ($-50°$C) in order to retain a $\rho_0 = 1\%$ bit-flip rate even after a time period of ten minutes. Halderman et al. also noted that bit-flip rates as low as $\rho_0 = 0.17\%$ are possible when liquid nitrogen is used for cooling. Another key observation was that memory has a ground state that the bits decay to over time, i.e. the noise introduced is very biased. However, it was also noticed that there is a very small but non-zero probability of retrograde bit-flips away from the ground state. It was estimated that these retrograde bit-flips occur at a rate of $\rho_1 \in [0.05 - 0.1\%]$. In a cold boot attack, then, the attacker is assumed to have physical access to a machine shortly after a power down cycle. The attacker proceeds by extracting from memory a noisy version of a scheme's secret key, where a small number of bits have been flipped. This can be achieved by performing a cold boot and loading a malicious operating system or alternatively, plugging the

victim's RAM chip into an external device. The attacker then recovers the key by applying bespoke error correction algorithms. To date, cold boot attacks have received a significant amount of attention across a range of cryptographic primitives including a variety of symmetric ciphers [65, 137, 75, 3], RSA [67, 66, 106], discrete log [113] and, most recently, NTRU [107] systems. We note that NIST considers resistance to side-channel attacks as a worthwhile, albeit secondary security feature in their post-quantum standardisation process [104]: "schemes that can be made resistant to side-channel attacks at minimal cost are more desirable than those whose performance is severely hampered by any attempt to resist side-channel attacks".

Contribution: Cold Boot Attacks on RLWE/MLWE Using the NTT The full details of this contribution can be found in Chapter 3. This contribution is based on our original publication in "*IACR Transactions on Cryptographic Hardware and Embedded Systems 2018*" (CHES 2018) [7]. The accompanying proof of concept code is publicly available [1]. We consider the resistance of RLWE- and MLWE-based schemes to cold boot attacks. Note that other classes of side-channel attacks have been considered on lattice-based schemes [52, 117, 35, 29], but to our knowledge, the aforementioned publication was the first to give a detailed consideration of cold boot attacks on secret key stored using an NTT. In light of the leakage resilience of (R)LWE [61, 44], we investigate how *cold boot leakage* of secrets stored as polynomial coefficients affects the hardness of the LWE problem. We show that for moderate cold boot error rates the resulting problem is considerably easier to solve than the side-channel-free RLWE/MLWE instances from which it is derived; for this analysis, we simply apply standard security estimates. However, we note that this analysis does not apply to many schemes as specified and implemented in practice. In particular, many schemes, e.g. [115, 131, 93, 142, 123, 128, 48, 127, 96, 38], make use of a power-of-two cyclotomic ring $\mathbb{Z}[x]/(x^n + 1)$. This ring is amenable to performing efficient multiplications with complexity $\mathcal{O}(n \log_2 n)$ using a (negacyclic) number theoretic transform (NTT). Adopting Fourier transform terminology, a polynomial that is the result of an NTT is said to be in the *frequency domain* whereas polynomials that have not been transformed are said to be in the *time domain*. In order to utilise the efficiency gains of the NTT, it is beneficial to store intermediate values in the frequency domain as is done in the Kyber specification [127]. This implementation

[1] at `https://bitbucket.org/Amit_Deo/coldboot-ntt/`

Table 1.1: Estimated costs of cold boot attacks on Kyber KEM keys stored in the non-NTT/NTT domain with ρ_0, ρ_1 cold boot bit-flip rates. The non-NTT success rate is always expected to be close to 100%. For more details, see the caption of Table 3.10.

bit-flip rates		NTT		non-NTT
ρ_0	ρ_1	cost	rate	cost
0.2%	0.1%	$3 \cdot 2^{21.1}$	95%	$2^{38.7}$
0.5%	0.1%	$3 \cdot 2^{33.1}$	87%	$2^{51.6}$
1.0%	0.1%	$3 \cdot 2^{43.3}$	91%	$2^{70.3}$
1.4%	0.1%	$3 \cdot 2^{53.6}$	91%	$2^{89.2}$
1.7%	0.1%	$3 \cdot 2^{62.8}$	89%	$2^{100.1}$

detail dramatically alters the landscape for cold boot attacks on RLWE/MLWE-based schemes using an NTT. Therefore, our contribution is the design of a practical cold boot attack for schemes storing secret polynomials in the frequency domain. The attack is described in Section 3.5.

While our attack in principle applies to all RLWE/MLWE schemes using an NTT to store secret keys, we focus on the example of the default Kyber parameters [127] for concreteness. We next summarise our findings. We establish the decoding cost for cold boot attacks when the NTT is not used for secret key storage, and obtain a solving cost of 2^{70} operations for $\rho_0 = 1\%, \rho_1 = 0.1\%$ bit-flip rates. We then develop a practical attack with a cost of roughly 2^{43} operations when the key is stored in the frequency domain for the aforementioned bit-flip rates. This attack relies heavily on the structure of the NTT. We present our findings in Table 3.10. In addition to the example of Kyber, we also analyse New Hope KEM [115] to give an idea of the attack performance on a RLWE-based scheme. The results for New Hope are slightly different to the Kyber results and are summarised in Table 3.15. In particular, for the bit-flip rates considered, the attack complexities on New Hope when using the NTT for key storage are comparable to the case where the NTT is not used.

For Kyber KEM, our results suggest that vulnerability to cold boot attacks can be mitigated by storing the secret in the time domain instead of the frequency domain. This counter measure would increase decryption time in a typical IND-CCA setting by a factor of at most two as such a conversion from the time to frequency domain must take place already due to the re-encryption step[2]. However,

[2]Note that the NTT is typically not the most expensive operation in RLWE/MLWE-based

Table 1.2: Estimated costs of cold boot attacks on New Hope KEM keys stored in the non-NTT/NTT domain with ρ_0, ρ_1 cold boot bit-flip rates. The non-NTT success rate is always expected to be close to 100%. For more details, see the caption of Table 3.15.

bit-flip rates		NTT		non-NTT
ρ_0	ρ_1	cost	rate	cost
0.17%	0.1%	$2^{48.7}$	84%	$2^{53.7}$
0.25%	0.1%	$2^{60.6}$	81%	$2^{60.0}$
0.32%	0.1%	$2^{70.2}$	81%	$2^{66.1}$

such a counter measure would not completely rule out cold boot attacks: for bit-flip rates of $\rho_0 = 0.2\%$ the resulting MLWE instance is still relatively easy to solve using the standard methods/security estimates (see Section 3.3). This countermeasure does not appear to be relevant in the case of New Hope according to Table 3.15 where the complexity of attacking a New Hope key remains comparable whether the NTT is used for key storage or not. However, future work may propose better algorithms for solving the cold boot NTT decoding problem.

A secondary contribution is an alternative cold boot attack on RLWE/MLWE secrets stored in the frequency domain based on the efficient Berlekamp-Massey algorithm [99] and Blahut's Theorem [26] (Section 3.6). Informally put, Blahut's theorem says that an infinite periodic sequence has linear complexity equal to the Hamming weight of a single period whereas the Berlekamp-Massey algorithm enables the efficient calculation of linear complexities. Unlike our main attack which works for standard secret distributions, the success of this attack depends on the use of low Hamming weight secrets i.e. secrets with a small number of non-zero coefficients. Note that low Hamming weight secret distributions have been considered for practical constructions [41, 13]. As a starting point, we show that for secret polynomials with a total of w non-zero coefficients, leakage of a consecutive block of $2w$ NTT entries leads to an extremely efficient key recovery attack. The intuition behind this observation is that the structure of the Berlekamp-Massey algorithm implies that a sequence with linear complexity w is fully defined by $2w$ consecutive coordinates. This base aspect of the attack depends on the Berlekamp-Massey algorithm and Blahut's theorem. Unfortunately, the likelihood of a cold boot attacker

schemes, thus the factor of two is conservative. There is also a conversion from the frequency to time domain during decryption. However, in an MLWE setting this operates on one ring element as opposed to k ring elements, which is the dimension of the secret s.

obtaining $2w$ consecutive error-less entries will typically be low for practical choices of q. To remedy this, one can try to guess which bit flips to correct within a block of $2w$ NTT coordinates and then run the base attack on each guess. This leads to a more realistic cold boot attack. For example, taking the New Hope parameters with low Hamming weight secrets i.e. $w = 64$, $n = 1024$, $q = 12289$, we obtain the following complexity estimates (ignoring the trivial cost of the base Berlekamp-Massey algorithm). For bit-flip rates of $(\rho_0, \rho_1) = (1\%, 0.1\%)$, we estimate an attack with complexity roughly 2^{80}, whereas for bit-flip rates of $(\rho_0, \rho_1) = (0.17\%, 0.1\%)$ the complexity is roughly 2^{28}. Additionally, the complexities associated to the same parameters but with $w = 128$ are 2^{163} and 2^{50} respectively.

1.3 Constructions

Next we briefly overview some interesting constructions that can be built from LWE/RLWE/MLWE etc. The very first construction based on LWE was public key encryption [118, 88]. Many signature schemes have also been constructed [58, 95, 1, 51] in addition to the constructions submitted to the NIST post-quantum standardisation process. One of the more fundamental primitives that can be built from lattices are pseudorandom functions (PRFs). These were first instantiated from lattice assumptions in [19] and later improved in [27, 18] to be approximately key homomorphic. In addition to standard PRFs, there has been much successful work on constructing PRFs with advanced properties. A selection of such PRFs include constrained-key PRFs [33], constrained key, programmable PRFs [111] and constraint-hiding constrained key PRFs [36]. Intuitively, constrained-key PRFs enable the production of constrained keys that provide PRF evaluations on points satisfying some constraint whilst returning random values on other points. Other advanced cryptographic primitives such as identity-based encryption [2] and attribute-based encryption [62, 34] have also been constructed from LWE. In addition to the advanced constructions above, fully homomorphic encryption can be built from LWE (and its variants) [53, 31, 59]. More recent advances in the area of FHE are beginning to show the practicality of FHE based on LWE assumptions [42]. In addition, one of the long-standing open problems of instantiating zero-knowledge proofs for all of NP from lattice-based assumptions was recently solved [112]. This small subset of constructions shows the constructive power and flexibility of LWE assump-

tions. Having said this, there are still some classically secure constructions without post-quantum alternatives, one of which is presented below.

Contribution: A Post-Quantum VOPRF Construction Our final contribution is to construct a secure verifiable oblivious pseudorandom function (VOPRF) protocol that is secure based on lattice assumptions. We stress that our construction is by no means practical but is (online) round-optimal. To our knowledge, our construction is the first post-quantum construction of a VOPRF.

A VOPRF is an interactive protocol between two parties; a client and a server. Intuitively, this protocol allows for a server with key k, to provide a client with an evaluation of a PRF on an input x that the client chooses using k as the key. Informally, the security of a VOPRF, from the server's perspective, guarantees that the client learns nothing more than the PRF evaluated at x using k as the key. Security from the perspective of the client guarantees the two conditions below:

1. the server learns nothing about the input x;
2. the client output is indeed the evaluation on input x and key k.

VOPRFs have numerous applications including secure keyword search [57], private set intersection [73], secure data de-duplication [77], password-protected secret sharing [70, 71], password-authenticated key exchange (PAKE) [72] and privacy-preserving lightweight authentication mechanisms [46]. A post-quantum VOPRF is required to afford these applications security against quantum adversaries.

The underlying (post-quantum) PRF we use is the ring version of the PRF from [18] (referred to as the BP14 PRF). Our basic VOPRF design and proof assumes certain non-interactive zero knowledge proofs of knowledge (NIZKPoKs). With the goal of creating an example instantiation of the required NIZKPoKs, we adapt the usage of Stern's [136, 86] protocol to argue knowledge of the input and (small) key to a BP14 PRF evaluation. We note that our analysis implies that we may also use the comparatively efficient framework of Beullens [24], but we focus on Stern proof formulations for simplicity. In addition, we use the Fiat-Shamir transform [54] to

obtain non-interactivity, meaning that our zero knowledge instantiations are secure in the *quantum* random oracle model by the results of [50, 91].

It should be said that we do not achieve a standard multi-party computation (MPC) notion of a VOPRF where security holds for *every* pair of inputs/secret keys (see [87] for standard MPC security definitions). Instead, our security definition for malicious clients asks for average case security over the sampling of a key from the key distribution dictated by the PRF. We argue that this is a reasonable concession since an honest server will always be sampling its key from the aforementioned distribution as it would want evaluations of the PRF that it provides to appear pseudorandom. On the other hand, security against malicious servers follows the standard notions from MPC and holds for any input/secret key pair. For more details of our definition, see Section 5.2.3.

To summarise, we show the existence of a post-quantum secure VOPRF in the quantum random oracle model whose security relies on the hardness of the RLWE and 1D-SIS problems. Note that the 1D-SIS problem is fairly non-standard but does have a reduction from a presumably hard lattice problem [33].

Road Map

We begin with a preliminary section (Chapter 2) where general lattice-based and traditional cryptographic preliminaries are presented along with the general notation that will be used. The three chapters following this are the *main* chapters listed next.

- Chapter 3: Cold boot attacks on RLWE/MLWE keys stored using an NTT.
- Chapter 4: Reductions between the MLWE and RLWE problems.
- Chapter 5: A post-quantum VOPRF construction.

The three main chapters each begin with a short synopsis highlighting the objectives, high-level techniques, and conclusions of that chapter followed by chapter-specific preliminaries. Note that each of the main chapters may be read independently of the others.

Chapter 2

Preliminaries

Contents

2.1 General Mathematical Notation

We let $\mathbb{R}$ denote the reals, $\mathbb{Q}$ denote the rationals, $\mathbb{Z}$ denote the integers, and $\mathbb{C}$ denote the complex numbers. For positive real y, we write $\lfloor y \rfloor$ to denote the integer part of y, $\lceil y \rceil$ to denote the smallest positive integer larger than y and $\lfloor y \rceil$ to denote the rounding of y to the nearest integer (rounding down in the case of a tie). Note that vectors and polynomials are rounded component-wise. We denote the integers modulo q as $\mathbb{Z}_q$. For $x \in \mathbb{Z}_q$, we define the rounding operation to $\mathbb{Z}_p$ as $\lfloor x \rceil_p :=$

$\left\lfloor \frac{p}{q} \cdot x \right\rceil$. For real numbers a and b where $b > a$, $[a, b] \subset \mathbb{R}$ denotes the closed interval $\{x \in \mathbb{R} : a \leq x \leq b\}$.

We denote vectors using bold font and use indices along with non-bold font to reference the entries of a vector. For example, the i^{th} entry of a vector $\boldsymbol{b}$ is denoted b_i. In general, p-norms of vectors will be denoted as $\|\cdot\|_p$ and $\|\cdot\|$ will be used to denote a Euclidean norm of a vector. We sometimes count indices from 0 and sometimes count from 1 depending on the context. We denote matrices analogously. For example, the $(i, j)^{th}$ entry of a matrix $\mathbf{B}$ is denoted as $B_{i,j}$. In addition, $\mathbf{B}^T$ denotes the transpose of $\mathbf{B}$ and $\mathbf{B}^{-T}$ represents the inverse of $\mathbf{B}^T$. The matrix $\mathbf{I}$ denotes the identity matrix. In cases where there are two types of vectors, we use an over-arrow to denote vectors, e.g. $\vec{b}$. to allow for easy differentiation between the classes of vector.

For a full-rank matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$, we denote by $\tilde{\mathbf{B}}$ the result of running the Gram-Schmidt orthogonalisation procedure on its columns. Considering the columns of a matrix $\mathbf{B} \in \mathbb{R}^{n \times n}$ as vectors, $\|\mathbf{B}\|$ will denote the Euclidean norm of the *longest* column of $\mathbf{B}$.

For a probability distribution $\mathcal{D}$, we write $s \leftarrow \mathcal{D}$ to denote that s is an element sampled from the distribution $\mathcal{D}$. If $\boldsymbol{s}$ is a k-dimensional vector, then $\boldsymbol{s} \leftarrow (\mathcal{D})^k$ denotes that each entry in $\boldsymbol{s}$ is drawn independently from the distribution $\mathcal{D}$. If $\mathcal{S}$ is a finite set, $s \leftarrow \mathcal{S}$ denotes that s is an element sampled from the uniform distribution over $\mathcal{S}$. Alternatively, we write $U(\mathcal{S})$ to denote the uniform distribution over the set $\mathcal{S}$.

We will denote the security parameter as λ. We use standard asymptotic notation ($\mathcal{O}, \omega, \Omega$ etc.). We let $\mathsf{poly}(\lambda)$ denote the set of polynomial functions in λ. With respect to the security parameter, a probabilistic algorithm $\mathcal{A}$ is said to be polynomial time if its running time can be bounded (in the worst case) by a function in $\mathsf{poly}(\lambda)$. Such an algorithm is said to be a probabilistic polynomial time (PPT) algorithm. A function μ is said to be negligible if for every $c > 0$, $\exists \lambda^*$ such that for all $\lambda > \lambda^*$, $|\mu(\lambda)| < 1/\lambda^c$. The set of negligible functions is denoted by $\mathsf{negl}(\lambda)$. By abuse of notation, we sometimes write $f(\lambda) = \mathsf{negl}(\lambda)$ to denote that f is a negligible function. We also write $a(\lambda) \gg b(\lambda)$ to denote that $a(\lambda) = \lambda^{\omega(1)} \cdot b(\lambda)$.

For an algorithm $\mathcal{A}$ outputting either 0 or 1, the advantage of $\mathcal{A}$ in distinguishing between two distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ is defined to be

$$\left| \Pr_{x \leftarrow \mathcal{D}_0}[\mathcal{A}(x) = 1] - \Pr_{x' \leftarrow \mathcal{D}_1}[\mathcal{A}(x') = 1] \right|.$$

If the advantage of any PPT algorithm in distinguishing distributions $\mathcal{D}_0$ and $\mathcal{D}_1$ is negligible (where the sizes of samples from $\mathcal{D}_i$ are in $\mathsf{poly}(\lambda)$), then we say that $\mathcal{D}_0$ and $\mathcal{D}_1$ are computationally indistinguishable.

2.1.1 Statistical Distance and Rényi Divergence

It is common to write security proofs/reductions that rely on the fact that certain distributions are close to each other. There are multiple ways of doing this, the most well-known being to rely on the notion of statistical distance.

Definition 1 (Statistical Distance)**.** *Let P and Q be distributions over some discrete domain X. The statistical distance between P and Q is defined as $\Delta(P, Q) := \sum_{i \in X} |P(i) - Q(i)|/2$. For continuous distributions, replace the sum by the appropriate integral.*

Claim 1. *If P and Q are two probability distributions such that $P(i) \geq (1 - \epsilon)Q(i)$ for all i, then $\Delta(P, Q) \leq \epsilon$.*

Importantly, the statistical distance between two distributions gives an upper bound on the distinguishing advantage of *any* distinguisher; even an unbounded one. The following two properties of statistical distance are simple to verify, given the definition and previous observation.

- ***Triangle Inequality:*** For any distributions P, Q, R, we have $\Delta(P, R) \leq \Delta(P, Q) + \Delta(Q, R)$.
- ***Data Processing Inequality:*** Let $f(X)$ denote the distribution induced by sampling $x \leftarrow X$ and applying a probabilistic algorithm/function f. Then for any distributions P, Q we have $\Delta(f(P), f(Q)) \geq \Delta(P, Q)$.

We can also make use of the Rényi divergence as an alternative to the statistical distance to measure the similarity between two distributions.

Definition 2. *(Rényi Divergence) For any distributions P and Q such that* $\mathrm{Supp}(P) \subseteq \mathrm{Supp}(Q)$*, the Rényi divergence of P and Q of order $a \in [1, \infty]$ is given by*

$$R_a(P\|Q) = \begin{cases} \exp\left(\sum_{x \in Supp(P)} P(x) \log \frac{P(x)}{Q(x)}\right) & \text{for } a = 1, \\ \left(\sum_{x \in Supp(P)} \frac{P(x)^a}{Q(x)^{a-1}}\right)^{\frac{1}{a-1}} & \text{for } a \in (1, \infty), \\ \max_{x \in Supp(P)} \frac{P(x)}{Q(x)} & \text{for } a = \infty. \end{cases}$$

For the case where P and Q are continuous distributions, we replace the sums by integrals and let $P(x)$ and $Q(x)$ denote probability densities. We also give a collection of well-known results on the Rényi divergence (cf. [82]), many of which can be seen as multiplicative analogues of standard results for statistical distance. The proof of this lemma is given in [138] and [82].

Lemma 1 (Useful facts on Rényi divergence)**.** *Let $a \in [1, +\infty]$. Also let P and Q be distributions such that $Supp(P) \subseteq Supp(Q)$. Then we have:*

- ***Increasing Function of the Order:*** *The function $a \mapsto R_a(P\|Q)$ is non-decreasing, continuous and tends to $R_\infty(P\|Q)$ as $a \to \infty$.*
- ***Log Positivity:*** *$R_a(P\|Q) \geq R_a(P\|P) = 1$.*
- ***Data Processing Inequality:*** *$R_a\left(P^f\|Q^f\right) \leq R_a(P\|Q)$ for any function f where P^f and Q^f denote the distributions induced by performing the function f on a sample from P and Q respectively.*
- ***Multiplicativity:*** *Let P and Q be distributions on a pair of random variables (Y_1, Y_2). Let $P_{2|1}(\cdot|y_1)$ and $Q_{2|1}(\cdot|y_1)$ denote the distributions of Y_2 under P and Q respectively given that $Y_1 = y_1$. Also, for $i \in \{1, 2\}$ denote the marginal distribution of Y_i under P resp. Q as P_i resp. Q_i. Then*
  - *$R_a(P\|Q) = R_a(P_1\|Q_1) \cdot R_a(P_2\|Q_2)$.*
  - *$R_a(P\|Q) = R_\infty(P_1\|Q_1) \cdot \max_{y_1 \in Supp(P_1)} R_a\left(P_{2|1}(\cdot|y_1)\|Q_{2|1}(\cdot|y_1)\right)$.*
- ***Probability Preservation:*** *Let $E \subseteq Supp(Q)$ be an arbitrary event. If $a \in (1, \infty)$, then $Q(E) \geq P(E)^{\frac{a}{a-1}} / R_a(P\|Q)$. Furthermore, we have $Q(E) \geq P(E)/R_\infty(P\|Q)$.*

- ***Weak Triangle Inequality:*** *Let P_1, P_2 and P_3 be three probability distributions such that $Supp(P_1) \subseteq Supp(P_2) \subseteq Supp(P_3)$. Then*

$$R_a\left(P_1 \| P_3\right) \leq \begin{cases} R_a\left(P_1 \| P_2\right) \cdot R_\infty\left(P_2 \| P_3\right), \\ R_\infty\left(P_1 \| P_2\right)^{\frac{a}{a-1}} \cdot R_a\left(P_2 \| P_3\right) \ \textit{if } a \in (1, +\infty). \end{cases}$$

2.2 Lattices

An n-dimensional lattice is a discrete additive subgroup of $\mathbb{R}^n$. A rank d lattice Λ can be written in terms of a set of linearly independent vectors $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ (where $\boldsymbol{b}_i \in \mathbb{R}^n$) as

$$\Lambda := \{\boldsymbol{x} \in \mathbb{R}^n : \boldsymbol{x} = \sum_{i=0}^{d-1} z_i \boldsymbol{b}_i, z_i \in \mathbb{Z}\}. \tag{2.1}$$

The set $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ is called a *basis* of the lattice. We can also represent this basis as a matrix $\mathbf{B} \in \mathbb{R}^{n \times d}$ where the j^{th} column of $\mathbf{B}$ is given by $\boldsymbol{b}_j$. The lattice generated by a basis $\mathbf{B}$ as in Equation (2.1), will be denoted as $\mathcal{L}(\mathbf{B})$. The determinant of a lattice with basis matrix $\mathbf{B}$ is defined to be $\sqrt{\det(\mathbf{B}^T\mathbf{B})}$. A lattice is *full-rank* if $n = d$ in which case, the determinant is given by $\det(\mathbf{B})$. Unless stated otherwise, all lattices should be assumed to be full-rank. Note that $\mathcal{L}(\mathbf{B}) = \mathcal{L}(\mathbf{C})$ if and only if $\mathbf{B} = \mathbf{CU}$ for some unimodular matrix $\mathbf{U} \in \mathbb{Z}_q^{d \times d}$. For any lattice Λ, the *dual lattice* Λ^* is defined to be

$$\Lambda^* := \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} \in \text{span}(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}), \langle \boldsymbol{y}, \Lambda \rangle \in \mathbb{Z}\}. \tag{2.2}$$

It can be shown that if $\mathbf{B} \in \mathbb{R}^{n \times d}$ is a basis for Λ, then $\mathbf{B} \cdot (\mathbf{B}^T\mathbf{B})^{-T}$ is a basis for Λ^*. For full rank lattices, the basis of the dual can simply be written as $\mathbf{B}^{-T}$. In either case, $\det(\Lambda) \cdot \det(\Lambda^*) = 1$.

We denote the length of the shortest non-zero vector of a lattice Λ as $\lambda_1(\Lambda)$. The k^{th} successive minima (where $k = 1, \ldots, d$) for a lattice Λ is defined to be the smallest positive real number r such that there exists at least k linearly independent lattice vectors of Euclidean norm at most r. We use $\lambda_k(\Lambda)$ to denote the k^{th} successive minima for lattice Λ. In addition, for any point $\boldsymbol{t} \in \mathbb{R}^n$, $\text{dist}(\Lambda, \boldsymbol{t})$ denotes the minimum value of $\|\boldsymbol{v} - \boldsymbol{t}\|$ over all $\boldsymbol{v} \in \Lambda$.

2.2.1 Lattice Problems

Below, we present definitions for *approximate* lattice problems where the approximation factor is denoted by γ. For the definitions of *exact* lattice problems, set $\gamma = 1$. As written below, we consider lattices as inputs to the various problems. In order to ensure that inputs can be expressed finitely, we may consider canonical finite representations of lattices as the inputs e.g. bases in Hermite normal form. For more information, see [100].

Definition 3 (γ-Approximate Shortest Vector Problem (SVP_γ))**.** *Given as input a lattice Λ, output a non-zero vector $\boldsymbol{v} \in \Lambda$ such that $\|\boldsymbol{v}\| \leq \gamma \cdot \lambda_1(\Lambda)$.*

Definition 4 (γ-Approximate Shortest Independent Vectors Problem (SIVP_γ))**.** *Given as input a lattice Λ, output a set of linearly independent vectors $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_d \in \Lambda$ such that $\max_i \|\boldsymbol{v}_i\| \leq \gamma \cdot \lambda_n(\Lambda)$.*

Definition 5 (γ-Approximate Closest Vector Problem (CVP_γ))**.** *Given as input a lattice Λ and a target point $\boldsymbol{t} \in \mathbb{R}^n$, output a lattice vector $\boldsymbol{v} \in \Lambda$ such that $\|\boldsymbol{v} - \boldsymbol{t}\| \leq \gamma \cdot dist(\Lambda, \boldsymbol{t})$.*

We can also consider a variant of CVP where it is guaranteed that the target point is very close to the lattice (relative to the length of the shortest vector in the lattice).

Definition 6 (γ-Approximate Bounded Distance Decoding Problem (γ-BDD))**.** *Given as input a lattice Λ and a target point $\boldsymbol{t} \in \mathbb{R}^n$ such that $dist(\Lambda, \boldsymbol{t}) \leq \gamma \cdot \lambda_1(\Lambda)$, output a lattice vector $\boldsymbol{v} \in \Lambda$ such that $\|\boldsymbol{v} - \boldsymbol{t}\| \leq \gamma \cdot \lambda_1(\Lambda)$.*

2.3 Gaussian Distributions (Over Lattices)

Definition 7 (Continuous Gaussian function/distribution)**.** *The Gaussian function of parameter r and centre c is defined as*

$$\rho_{r,c}(x) = \exp\left(-\pi(x-c)^2/r^2\right)$$

and the Gaussian distribution $D_{r,c}$ is the probability distribution whose probability density function is given by $\frac{1}{r}\rho_{r,c}$. Further, we define D_r to be the distribution $D_{r,0}$.

Definition 8 (Multivariate continuous Gaussian function/distribution)**.** *Let* $\Sigma = \boldsymbol{S}^T\boldsymbol{S}$ *for some rank-n matrix* $\boldsymbol{S} \in \mathbb{R}^{m\times n}$*. The multivariate Gaussian function with (scaled) covariance matrix* Σ *centred on* $\boldsymbol{c} \in \mathbb{R}^n$ *is defined as*

$$\rho_{\boldsymbol{S},\boldsymbol{c}}(\boldsymbol{x}) = \exp\left(-\pi(\boldsymbol{x}-\boldsymbol{c})^T(\boldsymbol{S}^T\boldsymbol{S})^{-1}(\boldsymbol{x}-\boldsymbol{c})\right)$$

and the corresponding multivariate Gaussian distribution denoted $D_{\boldsymbol{S},\boldsymbol{c}}$ *is defined by the density function* $\frac{1}{\sqrt{\det(\Sigma)}}\rho_{\boldsymbol{S},\boldsymbol{c}}$*.*

Remark 1. *In the above definition, the scaled covariance matrix differs to the standard covariance matrix by a factor of* 2π*. Throughout, we will ignore this factor of* 2π *and simply refer to scaled covariance matrices as covariance matrices.*

Remark 2. *As notation, if* $\boldsymbol{S}$ *in the above definition is diagonal with the entries of a vector* $\boldsymbol{s}$ *along the diagonal (i.e.* $\boldsymbol{S} = diag(\boldsymbol{s})$*), we write* $D_{\boldsymbol{s},c}$ *to represent the multivariate continuous Gaussian distribution.*

Definition 9 (Bounded width continuous Gaussian family)**.** *The bounded width Gaussian error family with parameter* $\alpha \geq 0$ *is defined to be the set* $\Psi_{\leq\alpha} := \{D_{\boldsymbol{s}} : |s_i| \leq \alpha\}$*.*

Definition 10 (Discrete Gaussian sums/distribution)**.** *The discrete Gaussian distribution over some n-dimensional lattice* Λ *and coset vector* $\boldsymbol{u} \in \mathbb{R}^n$ *with parameter* r*, denoted* $D_{\Lambda+\boldsymbol{u},r}$*, is the distribution with probability mass function* $\frac{1}{\rho_r(\Lambda+\boldsymbol{u})}\rho_r$*, where* $\rho_r(\Lambda+\boldsymbol{u}) := \sum_{\boldsymbol{x}\in\Lambda+\boldsymbol{u}}\rho_r(\boldsymbol{x})$ *is a discrete Gaussian sum over* $\Lambda+\boldsymbol{u}$*.*

We use the following shorthand conventions for Gaussian functions (and similarly for D_Λ):

- $\rho(\boldsymbol{x}) = \rho_{\mathbf{I},\mathbf{0}}(\boldsymbol{x}) = \exp(-\pi\|x\|^2)$
- $\rho_{\mathbf{S}}(\boldsymbol{x}) = \rho_{\mathbf{S},\mathbf{0}}(\boldsymbol{x})$
- $\rho_s(\boldsymbol{x}) = \rho_{s\mathbf{I},\mathbf{0}}(\boldsymbol{x})$
- $\rho_{\boldsymbol{s}}(\boldsymbol{x}) = \rho_{\mathrm{diag}(\boldsymbol{s}),\mathbf{0}}(\boldsymbol{x})$

We use the convention that the Fourier transform of $f : \mathbb{R}^n \to \mathbb{C}$ is given by $\hat{f}(\boldsymbol{z}) = \int_{\boldsymbol{x}\in\mathbb{R}^n} f(\boldsymbol{x})\cdot e^{-2\pi i\langle\boldsymbol{x},\boldsymbol{z}\rangle}d\boldsymbol{x}$. The Poisson summation formula over lattices is a useful

tool when proving results about discrete Gaussian distributions over lattices. It states that for a function $f : \mathbb{R}^n \to \mathbb{C}$ with Fourier transform $\hat{f}$, $\sum_{\boldsymbol{x}\in\Lambda} f(\boldsymbol{x}) = \det(\Lambda^*) \sum_{\boldsymbol{y}\in\Lambda^*} \hat{f}(\boldsymbol{y})$. After noting that the Fourier transform of ρ is precisely ρ, the Poisson summation formula states that for any lattice Λ, $\rho(\Lambda) = \det(\Lambda^*) \cdot \rho(\Lambda^*)$. After noting that $(\Lambda/r)^* = r \cdot \Lambda^*$, the Poisson summation formula for ρ_r is $\rho_r(\Lambda) = r^n \det(\Lambda^*)\rho_{1/r}(\Lambda^*)$. The Poisson summation formula can be straightforwardly applied to prove the following two facts for any lattice Λ:

1. For any $s \geq 1, \rho_s(\Lambda) = \rho(\Lambda/s) \leq s^n \rho(\Lambda)$.
2. For any coset $\Lambda + \boldsymbol{u}$, $\rho(\Lambda + \boldsymbol{u}) \leq \rho(\Lambda)$.[1]

These two facts are used to prove the following results, bounding the infinity and Euclidean norm of discrete Gaussians (with standard deviation 1). For a full proof of the below result (and of Lemma 5) see Daniele Micciancio's lecture notes on the Gaussian distribution. [2]

Lemma 2 ([16])**.** *For any n-dimensional lattice Λ and $\alpha \geq 1$, if $\boldsymbol{x} \leftarrow D_\Lambda$, then*

$$\Pr\left[\|\boldsymbol{x}\|_\infty \geq t\right] \leq 2n \exp(-\pi t^2)$$

and

$$\Pr\left[\|\boldsymbol{x}\| \geq \alpha \cdot \sqrt{\frac{n}{2\pi}}\right] \leq \left(\frac{\alpha^2}{\exp(\alpha^2 - 1)}\right)^{n/2}.$$

To get a more general result, we can use the fact that $D_{\Lambda,\sigma}$ can be sampled by sampling from $D_{\Lambda/\sigma}$ and then multiplying by σ.

Corollary 1. *For any n-dimensional lattice Λ, $\sigma > 0$, if $\boldsymbol{x} \leftarrow D_{\Lambda,\sigma}$, then*

$$\Pr\left[\|\boldsymbol{x}\|_\infty \geq t\right] \leq 2n \exp\left(-\frac{\pi t^2}{\sigma^2}\right)$$

and

$$\Pr\left[\|\boldsymbol{x}\| \geq \sigma\sqrt{n}\right] \leq c^{-n}$$

for some universal constant $c > 1$.

[1] The following fact is useful: $f(\boldsymbol{x}) = g(\boldsymbol{x} + \boldsymbol{c}) \implies \hat{f}(\boldsymbol{z}) = e^{2\pi i \langle \boldsymbol{z}, \boldsymbol{c}\rangle} \cdot \hat{g}(\boldsymbol{z})$

[2] available at `https://cseweb.ucsd.edu/classes/wi16/cse206A-a/LecGaussian.pdf`

In addition, there is a "drowning/smudging" lemma for discrete Gaussians over the integers. This is proved following the same reasoning as in [49].

Lemma 3. *Let $\sigma > 0$ and $y \in \mathbb{Z}$. The statistical distance between $D_{\mathbb{Z},\sigma}$ and $D_{\mathbb{Z},\sigma} + y$ is at most $|y|/\sigma$.*

A useful quantity when analysing properties of discrete Gaussian distributions over a lattice Λ is the smoothing parameter of Λ defined next. When discussing intuitive properties of the smoothing parameter η_ϵ, we will assume that the parameter ϵ is small (e.g. $\epsilon = 2^{-\lambda}$ for security parameter λ). The definition below says that the smoothing parameter is the smallest scaling factor s such that $\rho(s\Lambda^*)$ has *almost* all its weight on $\mathbf{0}$. Following the definition is a result bounding the size of the smoothing parameter.

Definition 11 (Smoothing parameter [101])**.** *For a lattice Λ and any $\epsilon > 0$, the* smoothing parameter $\eta_\epsilon(\Lambda)$ *is defined as the smallest $s > 0$ s.t. $\rho_{1/s}(\Lambda^*) = \rho(s\Lambda^*) \leq 1 + \epsilon$.*

Lemma 4 ([58])**.** *For any $\epsilon > 0$, and n-dimensional lattice Λ with basis $\mathbf{B}$,*

$$\eta_\epsilon(\Lambda) \leq \|\tilde{\mathbf{B}}\| \sqrt{\ln(2n(1 + 1/\epsilon))/\pi}.$$

It is clear from the definition that $\eta_\epsilon(c\Lambda) = c\eta_\epsilon(\Lambda)$. Once again, the Poisson summation formula is a handy tool when used in conjunction with the smoothing parameter. This is the case for the proof of the following lemma that says the continuous Gaussian distribution with any width larger than the smoothing parameter assigns an almost constant weight on any coset of Λ.

Lemma 5. *For any lattice Λ and coset $\Lambda + \boldsymbol{u}$, if $\eta_\epsilon(\Lambda) \leq 1$, then*

$$\rho(\Lambda + \boldsymbol{u}) \in [1 - \epsilon, 1 + \epsilon] \det(\Lambda^*).$$

Alternatively, for any $r \geq \eta_\epsilon(\Lambda)$,

$$\rho_r(\Lambda + \boldsymbol{u}) \in [1 - \epsilon, 1 + \epsilon] \det(r\Lambda^*).$$

As a corollary, combining the above lemma, and the fact that $\rho_r(\Lambda + \boldsymbol{u}) \leq \rho_r(\Lambda)$, we have the following.

Lemma 6 (Claim 3.8 in [119], Sums of Gaussians over cosets)**.** *For any lattice Λ, $\epsilon > 0$, $r \geq \eta_\epsilon(\Lambda)$ and $\boldsymbol{c} \in \mathbb{R}^n$, we have*

$$\rho_r(\Lambda + \boldsymbol{c}) \in \left[\frac{1-\epsilon}{1+\epsilon}, 1\right] \cdot \rho_r(\Lambda).$$

The following lemma states that we can efficiently sample from a Gaussian distribution over a lattice provided that the deviation of the Gaussian distribution is sufficiently large compared to the Gram-Schmidt length of a known basis.

Lemma 7 (Lemma 2.3 in [32], Sampling discrete Gaussians)**.** *There is a probabilistic polynomial-time algorithm that, given a basis $\boldsymbol{B}$ of an n-dimensional lattice $\Lambda = \mathcal{L}(\boldsymbol{B})$, $\boldsymbol{c} \in \mathbb{R}^n$ and parameter $r \geq \|\tilde{\boldsymbol{B}}\| \cdot \sqrt{\ln(2n+4)/\pi}$ outputs a sample distributed according to $D_{\Lambda+\boldsymbol{c},r}$.*

2.4 Lattice Basis Reduction

Here we briefly recall some of the foundational results/techniques in solving lattice problems. More techniques and details will be given in the relevant sections of this thesis. For any basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$, we denote the Gram-Schmidt orthogonalisation of this basis as $\{\boldsymbol{b}_0^*, \ldots, \boldsymbol{b}_{d-1}^*\}$. Concretely, for $i = 0, \ldots, d-1$, we have that

$$\boldsymbol{b}_i^* = \boldsymbol{b}_i - \sum_{j=0}^{i-1} \mu_{i,j} \boldsymbol{b}_j^*, \quad \mu_{i,j} := \frac{\boldsymbol{b}_i \cdot \boldsymbol{b}_j^*}{\|\boldsymbol{b}_j^*\|^2}. \tag{2.3}$$

Intuitively, $\boldsymbol{b}_i^*$ is the projection of $\boldsymbol{b}_i$ onto the space *orthogonal* to $\mathrm{span}(\{\boldsymbol{b}_0^*, \ldots, \boldsymbol{b}_{i-1}^*\}) = \mathrm{span}(\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{i-1}\})$. Alternatively, $\boldsymbol{b}_i^*$ is the projection of $\boldsymbol{b}_i$ onto $\mathrm{span}(\{\boldsymbol{b}_i^*, \ldots, \boldsymbol{b}_{d-1}^*\})$.

Definition 12 (δ-LLL reduced basis)**.** *A basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ is δ-LLL reduced for some value $\delta \in (1/4, 1)$ if*

1. *(Size reduced) for $0 \leq j < i \leq d-1$, $|\mu_{i,j}| \leq 1/2$,*
2. *(Lovász condition) for $0 \leq i < d-1$, $\delta\|\boldsymbol{b}_i^*\|^2 \leq \|\boldsymbol{b}_{i+1}^*\|^2 + \mu_{i+1,i}^2\|\boldsymbol{b}_i^*\|^2$.*

Setting $\alpha := 1/(\delta - 1/4)$, it can be shown that the following properties hold for δ-LLL reduced bases:

- $\|\boldsymbol{b}_1\| \leq \alpha^{(n-1)/2} \cdot \lambda_1$
- $\max_i \|\boldsymbol{b}_i\| \leq \alpha^{(n-1)/2} \cdot \lambda_n$

Given any basis of a lattice, the LLL algorithm [83] computes a δ-LLL reduced basis in time polynomial in the number of bits required to represent a basis. Very informally, the LLL algorithm runs through a sequence of iterations, terminating only when an iteration leads to an LLL reduced basis. Each iteration runs Babai's nearest plane algorithm [14] repeatedly to obtain a size reduced basis (see Algorithm 1) before performing a swap if the Lovász condition is violated. To see that Babai's nearest plane allows for a size reduced basis, one can use a geometric interpretation of the algorithm: on input $\boldsymbol{t}$ and lattice $\mathcal{L}(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1})$, Babai's nearest plane returns the unique lattice vector $\boldsymbol{v}$ such that $\boldsymbol{t} \in \left\{\boldsymbol{v} + \sum_{i=0}^{d-1} \mu_i \boldsymbol{b}_i^* : -\frac{1}{2} \leq \mu_i < \frac{1}{2}\right\}$.

Algorithm 1: Size-reduction sub-routine in LLL

Input: Basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$, Babai's nearest plane oracle $\mathsf{NearestPlane}$
Output: Basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ that is size reduced

```
i ← 1
while i ≤ d − 1 do
    b_i ← b_i − NearestPlane(b_i − b_i^*, L(b_0, ..., b_i))
    i ← i + 1
```

Using the properties stated above, the LLL algorithm therefore solves the SVP_γ and SIVP_γ problems for approximation factors $\gamma = \alpha^{(n-1)/2}$ in polynomial time for any $\alpha > 4/3$. In other words, there is a polynomial time algorithm solving SVP_γ and SIVP_γ for exponential approximation factors. In addition to this, it is a well-known fact that applying the LLL algorithm to a basis and then using Babai's nearest plane algorithm leads to a polynomial time algorithm solving CVP for exponential approximation factors [14]. Therefore, lattice problems with exponential approximation factors have been shown to be classically solvable in time polynomial in the lattice dimension.

2.5 Fields, Modules, Rings

We now present some relevant algebraic number theoretic preliminaries. The concepts recalled here are fairly standard. As a detailed reference, see the book of Neukirch [105]. For fields E and F, we denote that F is an extension field of E by writing E/F. Let K be an algebraic number field i.e. $K = K/\mathbb{Q}$. The degree of K is equal to the dimension of K as a vector space over $\mathbb{Q}$. The trace of a field element $x \in K$ is defined to be the trace of the linear map (acting on K when viewed as a vector space over $\mathbb{Q}$) corresponding to multiplication by x. An element $x \in K$ is said to be integral if it is the root of a monic polynomial with integer coefficients. The set of all integral elements forms the ring of integers of K denoted by $\mathcal{O}_K$. An order $\mathcal{O}$ of an algebraic number field K of degree n is a subring containing 1 that is also a rank n $\mathbb{Z}$-module. The simplest example is the ring of integers $\mathcal{O}_K$ which corresponds to the *maximal* order i.e. for any order of K, $\mathcal{O}$, we have that $\mathcal{O} \subseteq \mathcal{O}_K$. Suppose that $\zeta \in \mathbb{C}$ is an algebraic number (i.e. the root of some polynomial with rational coefficients) and let $f(X)$ denote the minimal polynomial of ζ with coefficients in $\mathbb{Q}$. The field extension $\mathbb{Q}(\zeta) \supset \mathbb{Q}$ is isomorphic to $\mathbb{Q}(X)/\langle f(X)\rangle$, so we can view field elements in $\mathbb{Q}(\zeta)$ as polynomials with degree at most $\deg(f) - 1$, and consider operations as polynomial multiplication/addition modulo the polynomial $f(X)$. An integral ideal of a ring R is an additive subgroup of R that is closed under multiplication by all ring elements.

Cyclotomic Fields/Rings A common example of an algebraic number field used in lattice based cryptography is the n^{th} cyclotomic field. The n^{th} cyclotomic field is obtained by adjoining a primitive n^{th} root of unity $\zeta_n = e^{2\pi\sqrt{-1}/n} \in \mathbb{C}$ to the rationals. It turns out that the ring of integers of a cyclotomic field $\mathbb{Q}(\zeta_n)$ is simply $\mathbb{Z}(\zeta_n)$ and we refer to this as a *cyclotomic ring*. The minimal polynomial of ζ_n is called the n^{th} cyclotomic polynomial. For power-of-two n, the $(2n)^{th}$ cyclotomic polynomial has the form X^n+1. Therefore, for power-of-two n, the $(2n)^{th}$ cyclotomic field (resp.ring of integers) can be represented as $\mathbb{Q}(X)/\langle X^n+1\rangle$ (resp. $\mathbb{Z}(X)/\langle X^n+1\rangle$). This is the representation commonly used in lattice-based cryptography. We sometimes informally refer to the $(2n)^{th}$ cyclotomic field for power-of-two n as the

cyclotomic field with power-of-two degree n [3].

2.5.1 Coefficient Embedding

We have seen that we can represent field elements in $\mathbb{Q}(\zeta)$ using polynomials of degree strictly less than $d = \deg(f)$ where f is the minimal polynomial of ζ. We can naturally view these polynomials as d-dimensional vectors with rational entries using a coefficient embedding:

$$\sum_{i=0}^{d-1} c_i X^i \longleftrightarrow (c_0, \ldots, c_{d-1})^T.$$

Moreover, we can view polynomial multiplication modulo $f(X)$ as a linear operation taking degree $d-1$ polynomials to degree $d-1$ polynomials or coefficient embeddings to coefficient embeddings. In particular, suppose we represent $\alpha \in \mathbb{Q}(\zeta)$ as a polynomial $\alpha(X)$. Then the matrix that acts on coefficient embeddings representing multiplication by $\alpha(X)$ is a $d \times d$ matrix whose i^{th} column is the coefficient embedding of $\alpha(X) \cdot X^i \bmod f(X)$.

Example 1. *Take the $(2n)^{th}$ cyclotomic field for power-of-two n, $K = \mathbb{Q}(\zeta_{2n}) \approxeq \mathbb{Q}(X)/\langle X^n + 1\rangle$. The multiplication matrix of $s = \sum_{i=0}^{n-1} s_i \zeta_{2n}^i$ is given by*

$$\mathrm{rot}(s) = \begin{bmatrix} s_0 & -s_{n-1} & -s_{n-2} & \cdots & \cdots & -s_1 \\ s_1 & s_0 & -s_{n-1} & \ddots & \ddots & -s_2 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \ddots & \vdots \\ s_{n-1} & s_{n-2} & \cdots & \cdots & \cdots & s_0 \end{bmatrix}. \tag{2.4}$$

Note that it is often the case that we work with polynomials whose coefficients are integers modulo some prime $q \in \mathbb{N}$. For example, it is common to work with the ring $\mathbb{Z}_q(X)/\langle X^n + 1\rangle$. In such cases, we define coefficient embeddings and multiplication matrices for $\mathbb{Z}_q(X)/\langle X^n + 1\rangle$ analogous to those for a field. In particular, $c(X) = \sum_{i=0}^{n-1} c_i X^i \in \mathbb{Z}_q(X)/\langle X^n + 1\rangle$ has $(c_0, \ldots, c_{n-1}) \in \mathbb{Z}_q^n$ as a coefficient embedding and a multiplication matrix whose i^{th} column corresponds to the coefficient embedding of $c(X) \cdot X^i \bmod X^n + 1$ where all coefficients are reduced modulo q.

[3] Formally, this is ambiguous as there are many cyclotomic polynomials of degree n, but we always mean the $(2n)^{th}$ cyclotomic polynomial of the form $X^n + 1$

2.5.2 Canonical Embedding

We can also use canonical embeddings to endow field elements with a geometry. A general number field $K = \mathbb{Q}(\zeta)$ has $r_1 + 2r_2$ field homomorphisms $\sigma_i : K \to \mathbb{C}$ fixing each element of $\mathbb{Q}$[4]. Let $\sigma_1, \ldots, \sigma_{r_1}$ be the real embeddings and $\sigma_{r_1+1}, \ldots, \sigma_{r_1+2r_2}$ be complex. The complex embeddings come in conjugate pairs, so we have $\sigma_i = \overline{\sigma_{i+r_2}}$ for $i = r_1 + 1, \ldots, r_1 + r_2$ using an appropriate ordering of the embeddings. Define

$$H \coloneqq \{\boldsymbol{x} \in \mathbb{R}^{r_1} \times \mathbb{C}^{2r_2} : x_i = \overline{x_{i+r_2}}, i = r_1 + 1, \ldots, r_1 + r_2\} \subset \mathbb{C}^{r_1+2r_2}$$

and let $(\hat{\boldsymbol{e}}_i)_{i=1}^n$ be the standard (orthonormal) basis of $\mathbb{R}^{r_1} \times \mathbb{C}^{2r_2}$ assumed in the above definition of H. We can easily change to the orthonormal basis $(\hat{\boldsymbol{h}}_i)_{i=1}^n$ defined by

- $\hat{\boldsymbol{h}}_i = \hat{\boldsymbol{e}}_i$ for $i = 1, \ldots, r_1$
- $\hat{\boldsymbol{h}}_i = \frac{1}{\sqrt{2}}(\hat{\boldsymbol{e}}_i + \hat{\boldsymbol{e}}_{i+r_2})$ for $i = r_1 + 1, \ldots, r_1 + r_2$
- $\hat{\boldsymbol{h}}_i = \frac{\sqrt{-1}}{2}(\hat{\boldsymbol{e}}_i - \hat{\boldsymbol{e}}_{i+r_2})$ for $i = r_1 + r_2 + 1, \ldots, r_1 + 2r_2$

to see that $H \simeq \mathbb{R}^n$ as an inner product space. The *canonical embedding* is defined as $\sigma_C : K \to \mathbb{R}^{r_1} \times \mathbb{C}^{2r_2}$ where

$$\sigma_C(x) \coloneqq (\sigma_1(x), \ldots, \sigma_n(x)).$$

The image of any field element under the canonical embedding lies in the space H, so we can always represent $\sigma_C(x)$ via the real vector $\sigma_H(x) \in \mathbb{R}^n$ through the change of basis described above. For any $x \in K$, $\sigma_H(x) = \mathbf{U}_H^\dagger \cdot \sigma_C(x)$ where the unitary matrix is given by

$$\mathbf{U}_H = \begin{bmatrix} \mathbb{I}_{r_1} & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}}\mathbb{I}_{r_2} & \frac{i}{\sqrt{2}}\mathbb{I}_{r_2} \\ 0 & \frac{1}{\sqrt{2}}\mathbb{I}_{r_2} & \frac{-i}{\sqrt{2}}\mathbb{I}_{r_2} \end{bmatrix} \in \mathbb{C}^{n \times n}. \tag{2.5}$$

Addition and multiplication of field elements is carried out component-wise in the canonical embedding, i.e. for any $x, y \in K$, $\sigma_C(xy)_i = \sigma_C(x)_i \cdot \sigma_C(y)_i$ and $\sigma_C(x+y) =$

[4] For the n^{th} cyclotomic field for any n, $r_1 = 0$ and $r_2 = \phi(n)/2$ where ϕ is the Euler totient function

$\sigma_C(x)+\sigma_C(y)$. Multiplication is not component-wise for σ_H. Specifically, in the basis $(\hat{\boldsymbol{e}}_i)_{i=1}^n$, we have that multiplication by $x \in K$ can be written as left multiplication by the matrix $X_{ij} = \sigma_i(x)\delta_{ij}$ where δ_{ij} is the Kronecker delta. Therefore, in the basis $(\hat{\boldsymbol{h}}_i)_{i=1}^n$, the corresponding matrix is $\mathbf{X}_H = \mathbf{U}_H^\dagger \mathbf{X} \mathbf{U}_H \in \mathbb{R}^{n\times n}$ which is not diagonal in general. However, for any $\mathbf{X}_H$, we have $\mathbf{X}_H \cdot \mathbf{X}_H^T = \mathbf{X}_H \cdot \mathbf{X}_H^\dagger = \mathbf{U}_H^\dagger \mathbf{X}\mathbf{X}^\dagger \mathbf{U}_H$. Explicitly, $(\mathbf{X}_H \cdot \mathbf{X}_H^T)_{ij} = |\sigma_i(x)|^2\delta_{ij}$ i.e. $\mathbf{X}_H \cdot \mathbf{X}_H^T$ is a diagonal matrix. Likewise for $\mathbf{X}_H^T \cdot \mathbf{X}_H$. Therefore, the singular values of $\mathbf{X}_H$ are precisely given by $|\sigma_i(x)|$ for $i = 1, \ldots, n$.

Remark 3. *We use $\sigma_i(\cdot)$ to denote both singular values and embeddings of field elements. If the argument is a matrix, it should be assumed that we are referring to singular values. Otherwise, $\sigma_i(\cdot)$ denotes a field embedding.*

Fields and Lattices We view any finitely generated $\mathbb{Z}$-submodule of K as a lattice in $\mathbb{R}^n$ using the canonical embedding along with the space H. The ring of integers $\mathcal{O}_K$ is an example of a finitely generated $\mathbb{Z}$-submodule of K and so can interpreted as a lattice. The same can be said for any ideal of $\mathcal{O}_K$.

2.6 Number Theoretic Transform

Let R be a power-of-two cyclotomic ring. Here, we define a transform that permits efficient multiplication. We defer useful properties and further details to the relevant sections of this thesis. Let q be a prime such that a $(2n)^{th}$ primitive root of unity $\gamma \in \mathbb{Z}_q$ exists, and set $\omega = \gamma^2 \bmod q$. The negacyclic number theoretic transform (NTT) in dimension n will be defined as the linear function $\mathsf{NTT} : \mathbb{Z}_q^n \to \mathbb{Z}_q^n$ given by

$$\mathsf{NTT}_n(\boldsymbol{a})_i := \sum_{j=0}^{n-1} \gamma^j \omega^{ij} a_j \bmod q.$$

We often refer to $\mathsf{NTT}_n(\boldsymbol{a})$ as being a vector in the *NTT domain* or *frequency domain*.

This transform has been shown to allow for fast polynomial multiplication in rings of the form $\mathbb{Z}_q[X]/\langle X^n + 1\rangle$ where n is a power of two [126, 140]. In particular, polynomial multiplication corresponds to component-wise multiplication in the NTT domain. The notation $\hat{\boldsymbol{a}}$ will be used as shorthand for the NTT of $\boldsymbol{a}$ and we often

drop the subscript n when its value is clear from the context. The inverse negacyclic NTT is given by

$$\mathsf{NTT}_n^{-1}(\hat{\boldsymbol{a}})_i := n^{-1}\gamma^{-i}\sum_{j=0}^{n-1}\omega^{-ij}\hat{a}_j.$$

Remark 4. *The NTT concretises the isomorphism*

$$\frac{\mathbb{Z}_q[X]}{\langle X^n+1\rangle} \cong \frac{\mathbb{Z}_q[X]}{\langle X-\gamma\rangle} \times \frac{\mathbb{Z}_q[X]}{\langle X-\gamma^3\rangle} \times \cdots \times \frac{\mathbb{Z}_q[X]}{\langle X-\gamma^{2n-1}\rangle} \tag{2.6}$$

that appears when applying the chinese remainder theorem to the ring $\mathbb{Z}_q[X]$ and ideal $\langle X^n+1\rangle$.

2.7 Learning With Errors

One of the most common assumptions made in lattice-based cryptography is that the Learning With Errors (LWE) problem is hard to solve — even for a quantum adversary. More precisely, suppose we set λ to be the security parameter. Then the assumption is that there are choices of LWE parameters leading to polynomial sized instances that no polynomial time quantum adversary can solve with non-negligible success probability in the average case. In order to define the decisional and search versions of LWE, we first define the LWE distribution. Let $\mathbb{T} = \{x : 0 \leq x < 1\}$.

Definition 13 (LWE distribution)**.** *For $\boldsymbol{s} \in \mathbb{Z}^n$ and error distribution ψ over $\mathbb{T}$, we sample the ring learning with errors (*LWE*) distribution $A_{n,q,\boldsymbol{s},\psi}$ over $\mathbb{Z}_q \times \mathbb{T}$ by outputting $(\boldsymbol{a}, \frac{1}{q}(\boldsymbol{a}\cdot\boldsymbol{s}) + e \bmod 1)$, where $\boldsymbol{a} \leftarrow U(\mathbb{Z}_q^n)$ and $e \leftarrow \psi$.*

Definition 14 (Decision/search LWE problem [118])**.** *The* decision *learning with errors problem $\mathsf{LWE}_{m,n,q,\Psi}(D)$ entails distinguishing m samples of $U(\mathbb{Z}_q \times \mathbb{T})$ from $A_{n,q,\boldsymbol{s},\psi}$ where $\boldsymbol{s} \leftarrow D$ and ψ is an arbitrary distribution in Ψ. The* search *variant S-$\mathsf{LWE}_{m,n,q,\Psi}(D)$ entails obtaining the secret $\boldsymbol{s} \leftarrow D$ given m samples of $A_{n,q,\boldsymbol{s},\psi}$. If m is omitted, it is assumed that the problem grants a $\mathsf{poly}(\lambda)$ number of samples to the solver. If D is omitted, it is assumed that $\boldsymbol{s} \leftarrow \mathbb{Z}_q^n$.*

One of the initial reasons for believing LWE is a hard problem is the following result due to Regev [118] saying that there is a reduction from SIVP to LWE. In order for the reduction to go through, we must use the error distribution ψ_α which is the continuous normal distribution with standard deviation $\beta/(\sqrt{2\pi})$.

Theorem 1 (Reduction from SIVP to LWE [118])**.** *Let $\alpha = \alpha(n) \in (0,1)$ be some real and $q = q(n)$ be some integer such that $\alpha q > 2\sqrt{n}$. If there exists a quantum algorithm solving $\mathsf{LWE}_{n,q,\psi_\alpha}$ or S-$\mathsf{LWE}_{n,q,\psi_\alpha}$ in polynomial time, then there is a quantum algorithm solving SIVP_γ for approximation factors $\gamma = \tilde{\mathcal{O}}(n/\alpha)$.*

Informally, the above states that solving LWE on average is at least as hard as solving SIVP in the worst case. Unfortunately, the reduction implicit in the theorem statement is not tight. Chatterjee et al. [37] show that if there is an algorithm W_1 solving $\mathsf{LWE}_{m=n^c,n,q,\psi_\alpha}$ with advantage greater than $1/n_1^d$ for a proportion $1/n^{d_2}$ of $s \in \mathbb{Z}_q^n$, then the algorithm for SIVP implicit in the theorem runs W_1 a total of $O(n^{11+c+2d_1+d_2})$ times. Therefore, it has been questioned whether the above theorem is a good indicator of the hardness of LWE for concrete parameters, since the non-tightness in the reduction to SIVP only implies an algorithm with much longer running time which we cannot confidently rule out. However, from a theoretical standpoint, the above theorem suggests that LWE is hard asymptotically.

To remedy the problems inherent in the reduction from LWE to SIVP, whenever concrete parameters are required, the best known attacks on LWE are considered. In order to account for future improvements, the runtimes of known attacks are under-estimated with respect to the state of the art [10].

2.8 Ring Learning With Errors

A common variant of the LWE problem is the so-called Ring Learning With Errors (RLWE) problem. Let R be the ring of integers of an algebraic number field $K/\mathbb{Q}$ and define the ring dual to R as $R^\vee := \{x \in K : \mathrm{Tr}(xR) \subseteq \mathbb{Z}\}$. Also let $K_\mathbb{R} = K \otimes_\mathbb{Q} \mathbb{R}$ and define $\mathbb{T}_{R^\vee} := K_\mathbb{R}/R^\vee$. Note that distributions over $K_\mathbb{R}$ are sampled by choosing an element of the space H (as defined in Section 2.5) according to the distribution and mapping back to $K_\mathbb{R}$ via the vector-space isomorphism $H \simeq K_\mathbb{R}$. For example, sampling the Gaussian distribution D_α over $K_\mathbb{R}$ is done by sampling D_α over $H \simeq \mathbb{R}^n$ and then mapping back to $K_\mathbb{R}$. In all definitions below, let Ψ be a *family* of distributions over $K_\mathbb{R}$, let Γ be a distribution over a family of distributions over $K_\mathbb{R}$, and let D be a distribution over $R_q^\vee$ where $R_q^\vee := R^\vee/(qR^\vee)$ and $R_q := R/(qR)$.

Definition 15 (RLWE distribution)**.** *For $s \in R_q^\vee$ and error distribution ψ over $K_\mathbb{R}$, we sample the ring learning with errors (RLWE) distribution $A_{q,s,\psi}^{(R)}$ over $R_q \times \mathbb{T}_{R^\vee}$ by outputting $(a, \frac{1}{q}(a \cdot s) + e \bmod R^\vee)$, where $a \leftarrow U(R_q)$ and $e \leftarrow \psi$.*

Definition 16 (Decision/search RLWE problem [97])**.** *The* decision *ring learning with errors problem $\mathsf{RLWE}_{m,q,\Gamma}^{(R)}(D)$ entails distinguishing m samples of $U(R_q \times \mathbb{T}_{R^\vee})$ from $A_{q,s,\psi}^{(R)}$ where $s \leftarrow D$ and $\psi \leftarrow \Gamma$. The* search *variant S-$\mathsf{RLWE}_{m,q,\Psi}^{(R)}(D)$ entails obtaining the secret s from m samples of $A_{q,s,\psi}^{(R)}$ where $s \in R_q^\vee$ and $\psi \in \Psi$ are arbitrary. If m is omitted, it is assumed that a $\mathsf{poly}(\lambda)$ number of samples is provided. If D is omitted, it is assumed that $s \leftarrow R_q^\vee$.*

RLWE can be seen as a structured variant of LWE. In fact, there is an analogous result to the reduction from SIVP to LWE in the context of RLWE. For a field K, we can view integral ideals of $\mathcal{O}_K$ as lattices using the canonical embedding. Such lattices are called *ideal lattices over K* and we refer to the SIVP_γ problem over ideal lattices as Ideal SIVP. It has been shown that for both cyclotomic rings [97] and general rings [110] that there is a reduction from Ideal SIVP_γ to RLWE. In the theorem below Γ_α is a family of distributions. For a number field K, let r_1 be the number of real embeddings and r_2 be the number of *pairs* of complex embeddings. Fixing an arbitrary $f(n) = \omega(\sqrt{\log n})$, a sample from Γ_α is an ellipsoidal Gaussian $D_{\boldsymbol{c}}$, where:

- for $i = 1, \ldots, r_1$, sample $x_i \leftarrow D_1$ and set $c_i^2 = \alpha^2(x_i^2 + f(n)^2)/2$,
- for $i = r_1 + 1, \ldots, r_1 + r_2$, sample $x_i, y_i \leftarrow D_{1/\sqrt{2}}$ and set $c_i^2 = c_{i+s_2}^2 = \alpha^2(x_i^2 + y_i^2 + f(n)^2)/2$.

In addition, we define the family $\Psi_{\leq\alpha} := \{D_{\boldsymbol{c}} : 0 < c_i \leq \alpha\}$.

Theorem 2 (Reduction from Ideal SIVP to RLWE [97, 110])**.** *Let K be an arbitrary number field of degree n and $R = \mathcal{O}_K$. Let $\alpha = \alpha(n) \in (0, 1)$ and $q = q(n) > 2$ be an integer such that $\alpha q > \omega(1)$. There is a polynomial time quantum reduction from SIVP_γ over ideal lattices in K to S-$\mathsf{RLWE}_{q,\Psi_{\leq\alpha}}^{(R)}$ and also $\mathsf{RLWE}_{q,\Gamma_\alpha}^{(R)}$ for approximation factor*

$$\gamma \leq \max\left\{\omega\left(\sqrt{n \log n}/\alpha\right), \sqrt{2}n\right\}.$$

Once again, the reduction above suffers from a lack of tightness. Similar to the case of plain LWE, under-estimated running times of the best-known attacks are used for concrete parameter selection. Interestingly, the best known attacks for RLWE and LWE are the same after interpreting the ring multiplication $a \cdot s$ as a matrix-vector multiplication. In other words, it is not known how to take practical advantage of the additional ring structure of the RLWE problem. Beyond the parameter sets used in cryptography, it has been shown that there *is* a polynomial time *quantum* algorithm solving Ideal SIVP_γ [43] for $\gamma = \exp(\tilde{\mathcal{O}}(\sqrt{n}))$. On the other hand, it is not known whether there exists a quantum polynomial time algorithm solving *general* SIVP with the same approximation factor. It is still believed that SIVP_γ is not solvable in quantum polynomial time for $\gamma = \mathcal{O}(n^c)$ in the case that $c < 1/2$.

2.9 Module Learning With Errors

Yet another variant of the LWE problem is Module Learning With Errors (MLWE). To get from the RLWE problem to the MLWE problem, we change ring elements $a \in R_q, s \in R_q^\vee$ to module elements $\boldsymbol{a} = (a_1, \ldots, a_d) \in (R_q)^d, \boldsymbol{s} = (s_1, \ldots, s_d) \in (R_q^\vee)^d$. In addition to this change, we replace the ring multiplication $a \cdot s$ with an inner product

$$\boldsymbol{a} \cdot \boldsymbol{s} := \sum_{i=1}^{d} a_i \cdot s_i.$$

Similarly to RLWE, we let Ψ be a *family* of distributions over $K_\mathbb{R}$, let Γ be a distribution over a family of distributions over $K_\mathbb{R}$, D be a distribution over $R_q^\vee$ where $R_q^\vee := R^\vee/(qR^\vee)$ and $R_q := R/(qR)$. Before presenting the formal definition, we remind the reader that R denotes the ring of integers of an algebraic number field K.

Definition 17 (MLWE distribution)**.** *Let $M := R^d$. For $\boldsymbol{s} \in (R_q^\vee)^d$ and an error distribution ψ over $K_\mathbb{R}$, we sample the module learning with errors distribution $A^{(M)}_{d,q,\boldsymbol{s},\psi}$ over $(R_q)^d \times \mathbb{T}_{R^\vee}$ by outputting $(\boldsymbol{a}, \frac{1}{q}\boldsymbol{a} \cdot \boldsymbol{s} + e \bmod R^\vee)$ where $\boldsymbol{a} \leftarrow U((R_q)^d)$ and $e \leftarrow \psi$.*

Definition 18 (Decision/search MLWE problem [81])**.** *Let $M = R^d$. The* decision *module learning with errors problem $\mathsf{MLWE}^{(M)}_{m,q,\Gamma}(D)$ entails distinguishing m samples of $U((R_q)^d \times \mathbb{T}_{R^\vee})$ from $A^{(M)}_{q,\boldsymbol{s},\psi}$ where $\boldsymbol{s} \leftarrow D^d$ and $\psi \leftarrow \Gamma$. The* search *variant S-*

$\mathsf{MLWE}^{(M)}_{m,q,\Psi}(D)$ *entails obtaining the secret element* $\boldsymbol{s} \leftarrow D^d$ *from* m *samples of* $A^{(M)}_{q,\boldsymbol{s},\psi}$ *where* $\boldsymbol{s} \in (R_q^\vee)^d$ *and* $\psi \in \Psi$ *are arbitrary. If* m *is omitted, it is assumed that a* $\mathsf{poly}(\lambda)$ *number of samples is provided. If* D *is omitted, it is assumed that* $\boldsymbol{s} \leftarrow (R_q^\vee)^d$.

Once again, a reduction from approximate SIVP (on the set of so-called module lattices) to MLWE has been shown to exist. In order to present the result, we first discuss module lattices. Taking a number field K of degree n, and any $\mathcal{O}_K$-module[5] $M \subseteq K^d$, we can embed M into $(\mathbb{R}^n)^d$ by applying σ_H to each component of M individually to obtain a lattice. Such lattices are called *rank-d module lattices over* K and we refer to the SIVP_γ problem over module lattices as Module SIVP. This result uses the same distributions $\Psi_{\leq\alpha}$ and Γ_α as the analogous RLWE result presented above.

Theorem 3 (Reduction from Module SIVP to MLWE [81])**.** *Let* K *be a degree* n *number field,* $R = \mathcal{O}_K$ *and* $M = R^d$ *for some integer* $d \geq 1$*. Further, let* $\alpha = \alpha(n) \in (0,1)$ *and integer* $q \geq 2$ *be such that* $\alpha q > 2\sqrt{d} \cdot \omega(\log n)$*. There is a polynomial time quantum reduction from* SIVP_γ *over rank* d *module lattices in* K *to* $S\text{-}\mathsf{MLWE}^{(M)}_{q,\Psi\leq\alpha}$ *and also* $\mathsf{MLWE}^{(M)}_{q,\Gamma_\alpha}$ *with approximation factor* $\gamma = \tilde{\mathcal{O}}(d\sqrt{n}/\alpha)$*.*

2.9.1 *Practical* R/MLWE Definitions

We will also be considering the definition of RLWE discussed in [98] since it best represents practical use. The reason for this is that the original RLWE definition given in Section 2.8 (and the definition of MLWE in Section 2.9) uses a continuous error distribution which is inconvenient in practice. As we will see, the definition below uses discrete error distributions rather than continuous ones. These definitions will be used in Chapters 3 and 5, whereas the more formal definitions above are used in Chapter 4.

Definition 19 (Practical RLWE distribution)**.** *For a "secret"* $s \in R_q$ *and an error distribution* χ *over* R*, a sample from the ring-*LWE *distribution* $A_{s,\chi}$ *over* $R_q \times R_q$ *is generated by choosing* $a \leftarrow R_q$ *uniformly,* $e \leftarrow \chi$ *and outputting* $(a, a \cdot s + e \bmod qR)$*.*

[5] where $\mathcal{O}_K$ represents the ring of integers of K

Definition 20 (Practical RLWE problems)**.** *The practical S-RLWE problem with secret distribution D over R entails recovering s from arbitrarily many samples of $A_{s,\chi}$ where $s \leftarrow D$. The practical decisional RLWE problem with secret distribution D entails distinguishing $A_{s,\chi}$ from uniform given arbitrarily many samples where $s \leftarrow D$.*

We note that in practice, we usually have a restriction on the number of samples available to an attacker.

Definition 21 (Practical MLWE distribution)**.** *For a "secret" $\boldsymbol{s} \in (R_q)^d$ and error distribution χ over R, a sample from the practical MLWE distribution $A_{d,\boldsymbol{s},\chi}$ over $(R_q)^d \times R_q$ is generated by choosing $\boldsymbol{a} \leftarrow (R_q)^d$ uniformly, $e \leftarrow (\chi)^d$ and outputting $(\boldsymbol{a}, \boldsymbol{a} \cdot \boldsymbol{s} + e \bmod qR)$.*

Definition 22 (Search module-LWE problem)**.** *The practical S-MLWE problem with secret distribution D over R entails recovering $\boldsymbol{s}$ from arbitrarily many samples of $A_{d,\boldsymbol{s},\chi}$ where $\boldsymbol{s} \leftarrow D^d$. The practical decisional MLWE problem with secret distribution D entails distinguishing $A_{d,\boldsymbol{s},\chi}$ from uniform given arbitrarily many samples where $\boldsymbol{s} \leftarrow D^d$.*

2.10 The 1D-SIS Problem

The next computational problem/assumption is slightly less common. It is the short integer solution problem in *dimension 1* (1D-SIS). The following formulation of the problem was used in [33] in conjunction with a lemma attesting to its hardness.

Definition 23. (1D-SIS, [33, Definition 3.4])*‘ The one-dimensional SIS problem, denoted $\text{1D-SIS}_{q,m,t}$, is the following: Given a uniform $\boldsymbol{v} \leftarrow \mathbb{Z}_q^m$, find $\boldsymbol{z} \in \mathbb{Z}^m$ such that $\|\boldsymbol{z}\|_\infty \leq t$ and $\langle \boldsymbol{v}, \boldsymbol{z} \rangle \in [-t, t] + q\mathbb{Z}$.*

In addition there is a useful variant of the 1D-SIS problem that is easier to use when writing certain proofs.

Definition 24. ([33, Definition 3.6]) *Let $q = p \cdot \prod_{i \in [n]} p_i$ where $p_1 < \cdots < p_n$ and p are all co-prime. Further, let $m \in \mathbb{N}$. The $\text{1D-SIS-R}_{q,p,m,t}$ problem is the following: Given $\boldsymbol{v} \leftarrow \mathbb{Z}_q^m$, find $\boldsymbol{z} \in \mathbb{Z}^m$ with $\|\boldsymbol{z}\|_\infty \leq t$ such that $\langle \boldsymbol{v}, \boldsymbol{z} \rangle \in [-t, t] + (q/p)\mathbb{Z}$.*

Evidence that both of these problems are hard is given via reductions from the SIVP problem as stated in the following lemmas.

Lemma 8. ([33, Corollary 3.5]) *Let $n \in \mathbb{N}$ and $q = \prod_{i\in[n]} p_i$ where all $p_1 < \ldots, < p_n$ are co-prime. Let $m \geq cn \log q$ (for some universal constant c). Assuming that $p_1 \geq t \cdot \omega(\sqrt{mn \log n})$, $\mathsf{1D\text{-}SIS}_{q,m,t}$ is at least as hard as $\mathit{SIVP}_{t\cdot\tilde{\mathcal{O}}(\sqrt{mn})}$.*

Lemma 9. ([33], Corollary 3.7) *Let q, p, t, m be as in Definition 24. Then the $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t}$ problem is at least as hard as $\mathsf{1D\text{-}SIS}_{q/p,m,t}$. Further, if $p_1 \geq t \cdot \omega(\sqrt{mn \log n})$, then $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,m,t}$ is at least as hard as $\mathit{SIVP}_{t\cdot\tilde{\mathcal{O}}(\sqrt{mn})}$.*

2.11 Zero Knowledge Proofs of Knowledge

Let L be a language defined by a relation $\mathsf{R_L}$. That is,

$$\mathsf{L} := \{x : \exists w \in \{0,1\}^*, \mathsf{R_L}(x, w) = 1\}.$$

At a high level, a zero knowledge proof of knowledge (ZKPoK) for relation $\mathsf{R_L}$ is an interactive protocol where a prover P wishes to convince a verifier V that it knows a $w^* \in \{0,1\}^*$ such that $\mathsf{R_L}(x, w^*) = 1$ for some common instance x. It is assumed that V is polynomial time. For the sake of V, the protocol should only result in V accepting if P can indeed exhibit knowledge of such a w^* (knowledge soundness). For the sake of P, if it indeed does know of a w^*, it should be able to make V accept (completeness). In addition, even when interacting with a malicious polynomial time verifier V^*, it is desired that P does not leak any information beyond what V^* could have computed on its own (zero-knowledge).

In the definition below, $\mathsf{P}(\cdot,\cdot)$ and $\mathsf{V}(\cdot)$ are interactive probabilistic algorithms, i.e. two algorithms that participate in some prescribed protocol together. We let $\langle \mathsf{P}(\cdot,\cdot), \mathsf{V}^*(\cdot)\rangle$ denote a random variable representing the output of V^* when interacting with P over the random coins of both P and V^*. In addition, we denote that an algorithm $\mathcal{A}$ has oracle access to $\mathcal{B}$ using the notation $\mathcal{A}^{\mathcal{B}}$ where the cost of an oracle call to $\mathcal{B}$ is counted as one. As a technical note, we keep any auxiliary inputs implicit throughout the definition for simplicity. For more details, see [60].

Definition 25 (ZKPoK)**.** *A ZKPoK (with knowledge soundness error κ) for a*

language L *with relation* R_L *is a pair of algorithms* $\mathsf{P}(\cdot,\cdot)$ *and* $\mathsf{V}(\cdot)$ *satisfying the following properties:*

- *(Completeness) For any* $x \in \{0,1\}^{\mathsf{poly}(\lambda)}$ *and* $w \in \{0,1\}^*$ *such that* $\mathsf{R}_\mathsf{L}(x,w) = 1$, $\Pr[\langle \mathsf{P}(x,w), \mathsf{V}(x)\rangle = 1] \geq 1 - \mathsf{negl}(\lambda)$.
- *(Knowledge soundness with error* κ*) There is a probabilistic algorithm* K *whose expected running time is* $\mathsf{poly}(\lambda)$ *such that for every* P^* *satisfying* $\Pr[\langle \mathsf{P}^*(x), \mathsf{V}(x)\rangle = 1] = \epsilon(x) > \kappa(|x|)$, $K^{\mathsf{P}^*(\cdot)}(x)$ *outputs a* w *such that* $\mathsf{R}_\mathsf{L}(x,w) = 1$ *with probability at least* $\epsilon(x) - \kappa(|x|)$.
- *(Zero knowledge) For every PPT* V^*, $\exists$ *a PPT algorithm* M^* *such that* $\forall x \in \{0,1\}^{\mathsf{poly}(\lambda)}$ *and* w *such that* $\mathsf{R}_\mathsf{L}(x,w) = 1$, *the distributions of* $M^*(x)$ *and* $\langle \mathsf{P}(x,w), \mathsf{V}^*(x)\rangle$ *are computationally indistinguishable.*

Note that a zero knowledge argument of knowledge (ZKAoK) is defined similarly to a ZKPoK apart from the fact that the knowledge soundness property only considers PPT adversaries P^*.

The Fiat-Shamir Transform Briefly, the random oracle model introduces a random function that all parties are given oracle access to in a cryptographic scheme. Assuming existence of such an oracle is a powerful tool when writing cryptographic proofs because oracle answers can be simulated/programmed advantageously. We call a protocol public coin if the verifier's messages consist of uniform random values in some range. In addition, a protocol is honest-verifier zero knowledge (HVZK) if the zero knowledge property holds against the prescribed verifier from the protocol description. Finally, a proof of knowledge is a protocol where the completeness and knowledge soundness properties from Definition 25 hold for some negligible κ. Fiat and Shamir proposed a way to transform a public coin, HVZK, constant-round proof of knowledge protocol into a non-interactive ZKPoK (NIZKPoK) in the random oracle model [54]. At an intuitive level, the Fiat-Shamir transform simply replaces the verifier's messages with calls to a random oracle. The formal proof of validity of the Fiat-Shamir transform was subsequently proven formally by Pointcheval and Stern [114]. More recently, the quantum random oracle model has been considered where parties have access to a quantum instantiation of the random function. The

motivation for this is that post-quantum security considers a world where quantum computation is feasible, so it makes sense that implementations of the random oracle function should be quantum as well. This introduces a significant change in the way that the random function is queried, so we cannot take the security of the Fiat-Shamir transform for granted. Nonetheless, recent work shows that the Fiat-Shamir transform is indeed valid in the quantum random oracle model [50, 91].

Chapter 3

Cold Boot/Leakage Attacks

Contents

3.1 Chapter Synopsis

In this chapter, we present attacks that allow for the computation of a RLWE/MLWE secret key given a "noisy" version of the secret key. The nature of the "noise" leads to different attack techniques. Our main attack (Section 3.5) fits into the cold boot setting [64] where an attacker is given the secret key (as stored in memory) apart from the fact that some of the bits are flipped. Our attack considers the case where an NTT is used to store the key which is fairly common amongst practical RLWE/MLWE based schemes [127, 96, 115]. Importantly, the positions of the bit flips are not known to the attacker.

Our main attack follows a relatively simple structure. We first use the divide and conquer strategy for *computing* an NTT to derive an incomplete binary tree of sub-instances of our original cold boot problem. The root of the tree represents the original instance in dimension n, say, and a node at level i represents a sub-instance of dimension $n/2^i$. Although the dimensions of sub-instances decrease further away from the root node, the instances do not necessarily become easier. In particular, sub-instances of *very small* dimension turn out to be ill-defined in the sense that there are many candidate solutions, but no efficient way of verifying which candidate solution is correct. Therefore, we only produce a tree of sub-instances to a carefully chosen level. The remainder of the attack involves solving the relatively low dimensional sub-instances at the bottom of our tree using practical lattice techniques and working a solution to a sub-instance back up the tree to the root node.

We show that our attack is feasible on practical RLWE/MLWE parameter sets and for practical cold boot bit flip rates. Throughout, we use the parameters from the Kyber KEM [127] as a concrete example, eventually estimating attack with a complexity of roughly 2^{43} using the bit flip rates of $\rho_0 = 1\%$, $\rho_1 = 0.1\%$ (see Section 3.2.3 for an explanation and justification of these values). If the NTT is not used for key storage, we estimate that a cold boot attack takes roughly 2^{70} operations at the same bit flip rates using standard lattice-based security estimates. Therefore, it appears that storing a key in the NTT domain makes practical cold boot attacks on MLWE schemes significantly easier. In cases where cold boot attacks are a concern, Kyber may be implemented by storing secret keys in terms of coefficients rather than the NTT. On the other hand, our experiments show that storing a key in the

NTT does not affect the difficulty of running a cold boot attack for the RLWE-based scheme New Hope [115]. For the very low but still very feasible bit flip rates $\rho = 0.17\%, \rho_1 = 0.1\%$, the NTT-based attack costs roughly 2^{49} operations whereas an attack without an NTT costs around 2^{54} operations. Tables 3.10 and 3.15 summarise our estimates for the cost of our attack. These findings are the result of running an implementation of the most expensive lattice-based aspect of our attack. The code used to generate these results is publicly available [1].

Following the main attack is an attack using the Berlekamp-Massey algorithm (Section 3.6) showing that it is very easy to recover RLWE secrets of Hamming weight w when given access to $2w$ consecutive entries of a secret vector in the NTT domain. For w considerably smaller than $n/2$ this attack solves a seemingly non-trivial problem. However, obtaining the leakage of $2w$ consecutive coefficients does not immediately fit nicely into the cold boot setting. Nonetheless, we do consider combining a naive guessing strategy with our Berlekamp-Massey attack to obtain a cold boot attack on RLWE secrets with low Hamming weight. The resulting attack on New Hope parameters with Hamming weight w secrets, for bit flip rates of $\rho_0 = 0.17\%, \rho_1 = 0.1\%$, performs as follows. For $w = 64$, the attack has a complexity of roughly 2^{28} whereas for $w = 128$, the attack costs roughly 2^{50} operations. Unfortunately, low Hamming weight schemes that use an NTT for key storage are not common, so this attack is theoretical rather than practical in nature.

Finally, in Section 3.7 we overview another recent work [45] that aims to solve a similar problem to our work. Informally, their result states that there is an attack recovering a RLWE secret key s given periodic entries of a secret vector in the NTT domain. Although this does not model a realistic attack setting, this attack is based much more explicitly on algebraic structure and does not require any heavy lattice reduction.

Road map We begin with chapter preliminaries in Section 3.2 that include a fairly detailed account of the general cold boot attack setting. We next discuss how to estimate the complexity of a cold boot attack on a M/RLWE-based scheme where an NTT is not used in Section 3.3. Next, we define our NTT-based cold boot attack

[1] at `https://bitbucket.org/Amit_Deo/coldboot-ntt`

problem and describe our main attack in Sections 3.4 and 3.5 respectively. After the main attack, we describe our Berlekamp-Massey based attack in Section 3.6 followed by an account of related literature in Section 3.7.

3.2 Chapter Preliminaries

We switch between polynomials and coefficient vectors throughout for convenience. For example, for a polynomial $s \in R_q$, we assume that $\boldsymbol{s} \in \mathbb{Z}_q^n$ is the coefficient vector of a single ring element s and represent an NTT using the most convenient out of $\mathsf{NTT}(s)$ or $\mathsf{NTT}(\boldsymbol{s})$. In addition, we denote MLWE keys using $\vec{s} = (s_0, \ldots, s_{d-1}) \in (R_q)^d$ and write $\vec{\boldsymbol{s}} = (\boldsymbol{s}_0, \ldots, \boldsymbol{s}_{d-1}) \in \mathbb{Z}_q^{nd}$ to denote the concatenation of coefficient vectors of the polynomials in $\vec{s}$.

We will use the following result, whose proof is an easy exercise:

Proposition 1. *Let* $X \leftarrow \{\pm 2^0, \pm 2^1, \pm 2^2, \ldots, \pm 2^{\ell-1}\}$. *We have*

$$\mathbb{E}[X] = 0 \quad \text{and} \quad \mathbb{E}[X^2] = \frac{4^\ell - 1}{3\,\ell}.$$

Furthermore, let $Y = (X_0, \ldots, X_{n-1})$ *where each* $X_i \leftarrow \{\pm 2^0, \pm 2^1, \pm 2^2, \ldots, \pm 2^{\ell-1}\}$. *Then the expected squared norm of* Y *is*

$$\mathbb{E}[\|Y\|^2] = n\left(\frac{4^\ell - 1}{3\,\ell}\right).$$

3.2.1 Minimal Binary Signed Digit Representation

We will often consider integers in *binary signed digit representation* (BSDR). This representation is reminiscent of a binary representation for positive integers, apart from the fact that each individual bit in a BSDR has its own sign. For example, $(1, 0, -1)$ is a BSDR of -3 because $-3 = 1 \cdot 2^0 + 0 \cdot 2^1 - 1 \cdot 2^2$. We also have that -3 can be written as $(-1, -1)$ in BSDR. It is clear that integers can have many BSDRs. In order to reduce the number of possibilities, we often consider the *minimal* BSDRs corresponding to the BSDRs with the minimum possible Hamming weight. For example, the minimal BSDR of 31 is $(-1, 0, 0, 0, 0, 1)$. Note that this

has a lower Hamming weight than the binary representation of 31 i.e. $(1, 1, 1, 1, 1)$. Even when considering minimal BSDRs, the issue of non-uniqueness can arise. The integer -3 is a simple example of this. One can also consider integers in q-ary signed digit representation (q-SDR). For example, if $q = 3$, a possible q-SDR of the integer 8 would be $(-1, 0, 1)$. Once again these representations are not unique. We extend these definitions to vectors in the obvious way, i.e. by considering vectors component-wise. For example, using 3 digits per symbol (or vector entry), the vector $(2, -4)$ has a minimal BSDR of $(0, 1, 0, 0, 0, -1)$.

3.2.2 Standard Methods for Solving BDD

One possible strategy for solving BDD is to first obtain a "high quality" basis for the lattice and then to run Babai's nearest plane algorithm to obtain a solution. Informally, the most desirable bases for running Babai are short and orthogonal. Obviously, for certain lattice geometries, short orthogonal bases do not necessarily exist. Due to this fact, definitions of "reduced" bases aim to *mimic* the notion of a short and orthogonal basis. LLL reduced bases are reduced in a relatively weak sense. Such bases can be computed in polynomial time in the lattice dimension using the LLL algorithm and lead to solutions of lattice problems with exponential approximation factors. Alternatively, the well-known BKZ algorithm [124, 40] outputs a so-called BKZ-reduced basis. This algorithm is parametrised by a block size β which is at most the lattice dimension d. The BKZ algorithm makes many calls to an oracle solving SVP in dimension β. High level pseudocode for the BKZ algorithm is given in Algorithm 2. Intuitively, both the running time and output basis quality

of BKZ increase with β.

Algorithm 2: High level pseudocode for BKZ

Input: Basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$, SVP oracle
Output: Basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ that is BKZ reduced

$z \leftarrow 0$, $j \leftarrow 0$, $LLL(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1})$ // LLL reduce the whole basis
while $z < d-1$ **do**
  /* π_j = projection onto space orthogonal to $span(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1})$ */
  $k \leftarrow \min\{j+\beta-1, d-1\}$ // define block
  $\mathcal{L}_{[j,k]} \leftarrow \mathcal{L}(\pi_j(\boldsymbol{b}_j), \ldots, \pi_j(\boldsymbol{b}_{d-1}))$
  /* Find shortest vector in local block's lattice */
  $\boldsymbol{b}^* = \sum_{i=j}^{k} v_i \pi_j(\boldsymbol{b}_i) \leftarrow \mathsf{SVP}(\mathcal{L}_{[j,k]})$
  if $\boldsymbol{v} \neq (1, 0, \ldots, 0)$ **then**
    $LLL(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{j-1}, \sum_{i=j}^{k} v_i \boldsymbol{b}_i, \boldsymbol{b}_j, \ldots, \boldsymbol{b}_{\min(k+1,n)})$
    $z \leftarrow 0$
  else
    $LLL(\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{\min(k+1,n)})$
    $z \leftarrow z+1$
  $j \leftarrow (j+1) \bmod (d-1)$

We now give the formal definition of a BKZ-reduced basis.

Definition 26 (BKZ-reduced basis)**.** *A lattice basis $\{\boldsymbol{b}_0, \ldots, \boldsymbol{b}_{d-1}\}$ is a δ-BKZ-β-reduced basis if*

- *it is δ-LLL reduced, and*
- *for $j = 0, \ldots, d-1$, $\|\boldsymbol{b}_j^*\| = \lambda_1(L_{[j,k]})$ where $L_{[j,k]}$ is the lattice with basis given by the components of $\boldsymbol{b}_j, \boldsymbol{b}_{j+1}, \ldots, \boldsymbol{b}_k$ orthogonal to the span of $\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{j-1}$ and $k = \min\{j+\beta-1, d-1\}$.*

After performing BKZ-β reduction, the first vector in the transformed lattice basis is assumed to have norm $\delta_0^n \cdot \det(\Lambda)^{1/n}$ where $\det(\Lambda)$ is the determinant of the lattice under consideration and the root-Hermite factor δ_0 is a constant based on β. More generally, the quality of a reduced basis $\mathbf{B}$ is related to the *flatness* of the slope of the logs of the lengths of the vectors $\boldsymbol{b}_i^\star$ in the Gram-Schmidt orthogonalisation of $\mathbf{B}$. For random lattices, the Geometric Series Assumption (GSA) is commonly assumed to hold:

Definition 27 (Geometric Series Assumption [125])**.** *For a random lattice, the norms of the Gram-Schmidt vectors after lattice reduction satisfy*

$$\|\boldsymbol{b}_i^*\| = \alpha^{i-1} \cdot \|\boldsymbol{b}_1\| \text{ for some } 0 < \alpha < 1.$$

Combining the GSA with the root-Hermite factor and the fact that $\det(\Lambda) = \prod_{i=1}^{n} \|\boldsymbol{b}_i^\star\|$, we get $\alpha = \delta_0^{-2n/(n-1)} \approx \delta_0^{-2}$. Increasing the block-size parameter β of BKZ-β leads to a smaller δ_0 but also leads to an increase in run-time. In this thesis, we consider the "enumeration regime" where lattice point enumeration is used to realise the exact SVP oracle in dimension β. In this case the running time grows as $\beta^{\Theta(\beta)}$ [76, 103]. The precise details of lattice enumeration are not essential. However, as intuition, enumeration in the context of BDD is an exhaustive search over lattice vectors of some maximal distance to the target point. A lattice vector candidate $\boldsymbol{v} = v_1\boldsymbol{b}_1 + \cdots + v_d\boldsymbol{b}_d$ is found one coefficient at a time, starting from v_d and ending at v_1. To obtain v_d, project the target point and lattice onto the space orthogonal to all but the last basis vector. A close vector to the projected target in this projected lattice is by definition an integer multiple of $\|\boldsymbol{b}_d^*\|$ and we set v_d to be such an integer multiple. We then project the lattice, target point and partial candidate $v_d\boldsymbol{b}_d$ onto the space orthogonal to all but the final two basis vectors. The coefficient v_{d-1} is set to be an integer multiple of $\boldsymbol{b}_{d-1}^*$ that is added to shorten the offset between the target point and partial candidate $v_d\boldsymbol{b}_d$ in this projected space. The rest of the coefficients are obtained similarly. Note that as soon as the offset length reaches some threshold, the partial candidate is thrown away and the process restarts using different choices of coefficients. From this description, one can view enumeration as a depth first search of a "pruned" enumeration tree. As a note, Babai's nearest plane algorithm has been generalised to consider multiple planes [88]. This, can also be considered as a form of pruned BDD enumeration [90].

We will follow the enumeration approach to solving BDD, i.e. we first compute a high quality basis and then run pruned enumeration to recover the (hopefully) closest vector to our target vector. As is standard, we run enumeration in some sub-dimension and then extend the solution in the projected sub-lattice to a full solution by running the standard Babai's nearest plane algorithm. This is equivalent to picking very small pruning coefficients for the smallest indices. We will make use of BKZ and enumeration as implemented in [55, 56] for our experiments. This

implementation also features a `Pruning` module, which computes parameters for pruned enumeration.

3.2.3 Cold Boot Attack Scenario

Cold boot attacks were introduced and studied in the seminal work of Halderman et al. [65]. Briefly, cold boot attacks rely on the fact that bits in RAM retain their value for some time after power is cut. Therefore, a cold boot attacker would typically either

1. Remove the RAM chip from the victim's device and plug into an external device to read the information on the RAM chip *or*
2. Power down the machine and then immediately turn it back on while loading a malicious operating system allowing for the bits in RAM to be read.

In order to preserve the information stored in RAM for longer, memory can be cooled to extreme temperatures. However, an attacker always ends up with a noisy version of memory where some of the bits have been flipped. It turns out that RAM chips have regions that eventually decay to a "ground state" of either a 0 value or a 1 value. With this in mind, the experiments of Halderman et al. [65] showed that there were two different classes of bit flips:

1. those flipping *towards* the ground state,
2. those flipping *away* from the ground state (i.e.retrograde flips).

In addition, there are two different bit flip *rates* associated to the two classes of bit flips: ρ_0 being the rate towards the ground state and ρ_1 being the rate in the retrograde (i.e. opposite) direction. Realistic values for ρ_0 and ρ_1 are as follows. Cooling RAM chips to ($-50°$C) before cutting the power allows for a $\rho_0 = 1\%$ bit-flip rate even after a time period of ten minutes. Halderman et al. also noted that bit-flip rates as low as $\rho_0 = 0.17\%$ are possible when liquid nitrogen is used for

cooling. As for the retrograde flips, it was estimated that these occur at a rate of $\rho_1 \in [0.05 - 0.1\%]$.

In a cold boot attack, then, the attacker gets access to a noisy version of memory. As is typically done, we will ignore the problem of identifying the location of secret key material in memory and assume the attacker gets access noisy version of a scheme's secret key, where a small number of bits have been flipped. The attacker then recovers the key by applying bespoke error correction algorithms. Our main attack (Section 3.5) is an example of such a bespoke error correction algorithm in the context of RLWE/MLWE. Since it is typical for secret keys to be stored in memory using an NTT, the main attack considers the case where the attacker is given access to a noisy NTT of the secret key.

3.3 Cold Boot Resilience for Kyber's Parameters (non-NTT)

For comparison, we now include security estimates for the case where we try to solve RLWE/MLWE given a noisy cold boot reading of the *coefficients* of the secret (rather than leakage in the NTT domain). As a concrete example, we use the default parameter set of the Kyber KEM [127], henceforth referred to simply as "Kyber", as the running example. However, we stress that our analysis applies generally to RLWE/MLWE keys as we will see later when the New Hope KEM [115] is considered. Kyber relies on the MLWE problem using module rank 3 over the ring $R_q = \mathbb{Z}_{7681}[x]/(x^{256}+1)$. Let $\mathrm{Bin}(n', p')$ denote the standard binomial distribution using n' trials with parameter $p' \in [0, 1]$. Kyber uses an error distribution denoted by $\mathcal{B}_\eta$ that corresponds to the shifted standard binomial distribution $\mathrm{Bin}(2\eta, 1/2) - \eta$. More specifically, the coefficients of the error polynomials follow the distribution $\mathcal{B}_\eta$ using the value $\eta = 4$.. This distribution has standard deviation $\sqrt{\eta/2} = \sqrt{2}$. In Kyber, the coefficients of the secret also follow $\mathcal{B}_\eta$.

Now, consider the Kyber public key $(\vec{a}, b \coloneqq \vec{a} \cdot \vec{s} + e)$ where $\vec{a}, \vec{s} \in (R_{7681})^3, e \in R$ with $s_i, e \leftarrow \mathcal{B}_\eta$ and assume that, due to some leakage, we are given a noisy version of $\boldsymbol{s} \in (R_{7681})^3$ denoted by $\vec{\tilde{s}} \coloneqq \vec{s} + \Delta$. Here, the addition is over R_{7681} and Δ is an element of $(R_{7681})^3$ representing bit-flips. This means that each coefficient of Δ should have low Hamming weight when written in minimal BSDR. For illustrative

purposes, we will focus on cold boot bit-flip rates of $\rho_0 = 1.0\%$ towards the ground state and a retrograde bit-flip rate of $\rho_1 = 0.1\%$, cf. [65]. We consider

$$\vec{a} \cdot \vec{\tilde{s}} - b = \vec{a} \cdot \vec{\tilde{s}} - \vec{a} \cdot \vec{s} - e = \vec{a} \cdot (\vec{\tilde{s}} - \vec{s}) + e = \vec{a} \cdot \Delta + e \tag{3.1}$$

which is an MLWE instance for the secret Δ. We note that the conversion works both ways, i.e. an attacker who can find Δ can then solve the above MLWE instance, and thus the two problems are equivalent.

By definition of $\mathcal{B}_\eta$ we have that coefficients of $\vec{s}$ have absolute value bounded by $\eta = 4$. Thus, the secret coefficients fit into four bits (including one sign bit) and we may assume that Δ is both relatively sparse (at least when considered in minimal BSDR) and has coefficients that are bounded by $\eta = 4$ in absolute value. This means that we only need to consider $768 \cdot 4$ bits altogether. We assume that half of these bits are in the ground state of memory and the other half are not. That is, for $\rho_0 = 1.0\%, \rho_1 = 0.1\%$, we obtain a Δ with an expected number of $17 = \lceil (1.0 + 0.1)/100 \cdot 768 \cdot 4/2 \rceil$ non-zero coefficients, each bounded by four in absolute value. According to the LWE estimator from [9] the MLWE instance (3.1) for these parameter sets take $\approx 2^{70.3}$ operations to solve assuming enumeration is used to realise the SVP oracle [40].[2] This attack might be improved somewhat by taking into account the *a priori* distribution of $\vec{s}$. More values are given in Table 3.10 and estimates for the New Hope KEM are given in Table 3.15.

3.4 Cold Boot NTT Decoding Problem

The discussion in the previous section assumes that the MLWE secret $\vec{s}$ is stored in RAM as a coefficient vector with small entries, allowing a cold boot attacker to obtain a noisy image of $\boldsymbol{s}$. Yet, to maximise efficiency, Kyber stores $\hat{\vec{s}} = \mathsf{NTT}_n(\vec{s})$ where by abuse of notation, we write $\mathsf{NTT}_n(\vec{s}) := (\mathsf{NTT}_n(\boldsymbol{s}_0), \mathsf{NTT}_n(\boldsymbol{s}_1), \mathsf{NTT}_n(\boldsymbol{s}_2))$. Thus, a cold boot attacker *does not* encounter a noisy version of $\vec{s}$ but a noisy version of $\mathsf{NTT}_n(\vec{s})$. In other words, the costs derived in Section 3.3 are immaterial

[2] The following call to the code available at `https://bitbucket.org/malb/lwe-estimator` was used to establish this cost:

```
sage: f = partial(drop_and_solve, primal_usvp, n=3*256, q=7681, al-
pha=sqrt(2)*sqrt(2*pi)/7681,
reduction_cost_model=BKZ.CheNgu12, decision=False, postprocess=False)
sage: f(secret_distribution=((-4, 4), ceil((1.0 + 0.1)/100 * 4 * 768/2)))
```

for a real-world attack on Kyber. In particular, the decoding problem encountered during a cold boot attack on M/RLWE-based schemes utilising an NTT, is as follows:

Definition 28 (Cold boot NTT decoding problem)**.** *Let* NTT *be a (negacyclic) NTT of dimension n modulo q, let ξ be some known constant mod q, let $\boldsymbol{s} \in \mathbb{Z}_q^n$ be the coefficient vector of a single polynomial with some known distribution χ and let $\Delta \in \mathbb{Z}_q^n$ be a coefficient vector with known distribution ψ. Then the* Cold Boot NTT Decoding Problem *is to recover $\boldsymbol{s}$ given*

$$\tilde{\tilde{\boldsymbol{s}}} := \xi\,\mathsf{NTT}(\boldsymbol{s}) + \Delta.$$

In the definition above, we slightly generalise the cold boot problem encountered by permitting a scaling factor ξ, cf. Section 3.5.1. Note that all quantities in the problem statement are interpreted as coefficient vectors rather than polynomials. In addition, the coefficient vectors are associated to single polynomials rather than non-trivial rank module elements. In other words, recovering an MLWE key involves solving multiple instances of the described cold boot NTT decoding problem.

As before, in our setting Δ corresponds to bit-flips which means that each component of Δ should have low Hamming weight when written in minimal BSDR. However, contrary to the discussion in Section 3.3, the norm of the "noise term" Δ is not necessarily small. By analogy with LWE, it will be convenient to consider the problem with the roles of $\boldsymbol{s}$ and Δ reversed, i.e. to consider the inverse NTT of the above instance. In particular, we will be considering the problem of recovering $\boldsymbol{s}$ or Δ given

$$\tilde{\boldsymbol{s}} := \mathbf{W} \cdot \Delta + \boldsymbol{s} \tag{3.2}$$

where $\mathbf{W}$ is the inverse (of a possibly scaled by some constant) negacyclic NTT matrix for dimension n, $\tilde{\boldsymbol{s}}$ is known, $\boldsymbol{s}$ is small and Δ is sparse in minimal BSDR. We sometimes write $\mathbf{W}_n$ to explicitly indicate the dimension of the NTT.

In a standard LWE setting, an adversary is essentially given a noisy product of a matrix $\mathbf{A}$ with a secret vector where $\mathbf{A}$ has entries uniformly chosen modulo q. Indeed, to prevent precomputation attacks, Kyber specifies that a fresh $\mathbf{A}$ is computed for each new secret. In contrast, each instance of our decoding problem has the same $\mathbf{W}$ which is the matrix representation of a scaled inverse negacyclic NTT. Thus, precomputation attacks become feasible. More importantly, though, this

matrix is highly structured and, indeed, the q-ary lattices derived from this matrix do not behave like random lattices. We consider this in Sections 3.5.1 and 3.5.4.1.

We note that while we are only given n LWE-like samples in our decoding problem, the problem is still well defined, despite Δ not being small. This is because Δ is sparse when its components are written in BSDR form. On the other hand, the distribution of Δ implies that standard techniques for solving LWE-like problems need to be adapted. We consider this in Section 3.5.3.

We parametrise the cold boot NTT decoding problem by a parameter κ representing the number of expected bit-flips; explicitly:

$$\kappa \coloneqq \lceil (\rho_0 + \rho_1) \cdot n \cdot \lceil \log_2 q \rceil / 2 \rceil.$$

Finally, we note that, for Kyber, the dimension of the problem is immediately reduced from $n \cdot d = 768$ to $n = 256$ since a single Kyber key gives rise to d independent cold boot problems. It should be noted that this reduction in dimension does not occur when considering RLWE keys since RLWE is effectively MLWE with $d = 1$. For bit-flip rates of 0.17% and 1% in the ground state direction (and 0.1% in the retrograde direction), we expect a total of less than $\lceil (0.17 + 0.1) \cdot 256 \cdot 13/200 \rceil = 5$ and $\lceil (1 + 0.1) \cdot 256 \cdot 13/200 \rceil = 19$ bits to be flipped respectively. Therefore, under these cold boot assumptions, we expect either 5 or 19 unknown bit-flips. Note that in both cases, the number of retrograde bit-flips is approximately 2. The case $\rho_0 = 0.17\%$ can therefore be solved by exhaustive search in $\binom{13 \cdot 256/2}{3} \cdot \binom{13 \cdot 256/2}{2} \approx 2^{50}$ operations. For the case, $\rho_0 = 1.0\%$, the naive strategy of simply guessing the positions of bit-flips implies an attack of complexity roughly $\binom{13 \cdot 256/2}{17} \cdot \binom{13 \cdot 256/2}{2} \approx 2^{154}$. This latter value of ρ_0 will be used as our running example.

3.5 The Main Cold Boot Attack

In this section, we describe our main cold boot attack on RLWE/MLWE where secret keys are stored in the NTT domain. The attack can be separated into three distinct parts:

1. (Section 3.5.1) Deriving low-dimensional cold boot NTT decoding sub-instances.
2. (Section 3.5.2) Extending the solution of a sub-instance to a solution of the original cold boot attack instance.
3. (Section 3.5.3) Solving the low dimensional sub-instances.

3.5.1 Divide and Conquer

It is well known that a 2^n-dimensional Fourier transform can be written in terms of two 2^{n-1}-dimensional Fourier transforms. The same holds for a negacyclic NTT. To simplify the presentation of the appropriate formulae, define $\boldsymbol{g}^{(e)} := (g_0, g_2, \ldots, g_{n-2})$ and $\boldsymbol{g}^{(o)} := (g_1, g_3, \ldots, g_{n-1})$ for any $\boldsymbol{g} \in \mathbb{Z}_q^n$. The negacyclic NTT can be shown to satisfy the following relations:

$$2\mathsf{NTT}_{n/2}(\boldsymbol{g}^{(e)})_i = \mathsf{NTT}_n(\boldsymbol{g})_i + \mathsf{NTT}_n(\boldsymbol{g})_{i+n/2} \tag{3.3}$$

$$2\gamma\omega^i\mathsf{NTT}_{n/2}(\boldsymbol{g}^{(o)})_i = \mathsf{NTT}_n(\boldsymbol{g})_i - \mathsf{NTT}_n(\boldsymbol{g})_{i+n/2} \tag{3.4}$$

for $i \in \{0, 1, \ldots, n/2 - 1\}$.

Example 2. *Consider $n = 8$, given a $2n$-th root of unity γ, we can write the forward negacyclic NTT in matrix form as*

$$\boldsymbol{V}_n = \begin{pmatrix} 1 & \gamma & \gamma^2 & \gamma^3 & \gamma^4 & \gamma^5 & \gamma^6 & \gamma^7 \\ 1 & \gamma^3 & \gamma^6 & -\gamma & -\gamma^4 & -\gamma^7 & \gamma^2 & \gamma^5 \\ 1 & \gamma^5 & -\gamma^2 & -\gamma^7 & \gamma^4 & -\gamma & -\gamma^6 & \gamma^3 \\ 1 & \gamma^7 & -\gamma^6 & \gamma^5 & -\gamma^4 & \gamma^3 & -\gamma^2 & \gamma \\ 1 & -\gamma & \gamma^2 & -\gamma^3 & \gamma^4 & -\gamma^5 & \gamma^6 & -\gamma^7 \\ 1 & -\gamma^3 & \gamma^6 & \gamma & -\gamma^4 & \gamma^7 & \gamma^2 & -\gamma^5 \\ 1 & -\gamma^5 & -\gamma^2 & \gamma^7 & \gamma^4 & \gamma & -\gamma^6 & -\gamma^3 \\ 1 & -\gamma^7 & -\gamma^6 & -\gamma^5 & -\gamma^4 & -\gamma^3 & -\gamma^2 & -\gamma \end{pmatrix}$$

Adding the rows i and $i+4$ for $i \in \{0, 1, 2, 3\}$, we obtain $W_n^{(+)}$ as shown below which corresponds to the NTT matrix for $n = 4$ scaled by $\xi = 2$:

$$\boldsymbol{V}_n^{(+)} = \begin{pmatrix} 2 & 0 & 2\gamma^2 & 0 & 2\gamma^4 & 0 & 2\gamma^6 & 0 \\ 2 & 0 & 2\gamma^6 & 0 & -2\gamma^4 & 0 & 2\gamma^2 & 0 \\ 2 & 0 & -2\gamma^2 & 0 & 2\gamma^4 & 0 & -2\gamma^6 & 0 \\ 2 & 0 & -2\gamma^6 & 0 & -2\gamma^4 & 0 & -2\gamma^2 & 0 \end{pmatrix}, \quad 2\,\boldsymbol{V}_{n/2} = \begin{pmatrix} 2 & 2\gamma^2 & 2\gamma^4 & 2\gamma^6 \\ 2 & 2\gamma^6 & -2\gamma^4 & 2\gamma^2 \\ 2 & -2\gamma^2 & 2\gamma^4 & -2\gamma^6 \\ 2 & -2\gamma^6 & -2\gamma^4 & -2\gamma^2 \end{pmatrix}.$$

Using this halving property, we can split our cold boot NTT decoding problem into two smaller cold boot NTT decoding problems. Recall that our cold boot instance

is described by the equation $\tilde{\boldsymbol{s}} = \mathsf{NTT}_n^{-1}(\Delta) + \boldsymbol{s}$ (see Equation (3.2)). To show how we utilise Equations (3.3) and (3.4), we perform the following steps:

1. Take a forward NTT to obtain the instance $\mathsf{NTT}_n(\tilde{\boldsymbol{s}}) = \mathsf{NTT}_n(\boldsymbol{s}) + \Delta$.
2. Perform the two folding steps:
   (a) (Positive Fold) Compute the vector described by
   $$\mathsf{NTT}_n(\tilde{\boldsymbol{s}})_i + \mathsf{NTT}_n(\tilde{\boldsymbol{s}})_{i+n/2} = 2\mathsf{NTT}_{n/2}(\boldsymbol{s}^{(e)})_i + (\Delta_i + \Delta_{i+n/2}).$$
   (b) (Negative Fold) Compute the vector described by
   $$\frac{1}{2\gamma\omega^i}\left(\mathsf{NTT}_n(\tilde{\boldsymbol{s}})_i - \mathsf{NTT}_n(\tilde{\boldsymbol{s}})_{i+n/2}\right) = \mathsf{NTT}_{n/2}(\boldsymbol{s}^{(o)})_i + \frac{1}{2\gamma\omega^i}(\Delta_i - \Delta_{i+n/2}).$$
3. Define $\Delta_{(l)} := (\Delta_0, \ldots, \Delta_{n/2-1})$, $\Delta_{(r)} := (\Delta_{n/2}, \ldots, \Delta_{n-1})$ and do the following:
   (a) (Positive Fold): Multiply by $2^{-1} \bmod q$ and take an inverse NTT. The resulting instance is
   $$\tilde{\boldsymbol{s}}^{(e)} = 2^{-1}\,\mathsf{NTT}_{n/2}^{-1}(\Delta_{(l)} + \Delta_{(r)}) + \boldsymbol{s}^{(e)}.$$
   (b) (Negative Fold) Define the matrix $\boldsymbol{\Omega}$ such that $\Omega_{i,j} = (\gamma\omega^i)^{-1}\delta_{i,j}$ where $\delta_{i,j}$ is the Kronecker delta function. Take an inverse NTT to obtain the instance
   $$\tilde{\boldsymbol{s}}^{(o)} = 2^{-1}\,\mathsf{NTT}_{n/2}^{-1}\left(\boldsymbol{\Omega}\cdot(\Delta_{(l)} - \Delta_{(r)})\right) + \boldsymbol{s}^{(o)}.$$

To summarise, in matrix notation, we can halve the dimension of the instance $\tilde{\boldsymbol{s}} = \mathbf{W}_n \cdot \Delta + \boldsymbol{s}$ by performing the folding step and deriving the following two instances of half the dimension:

$$\tilde{\boldsymbol{s}}^{(e)} = 2^{-1}\mathbf{W}_{n/2}\cdot(\Delta_{(l)} + \Delta_{(r)}) + \boldsymbol{s}^{(e)}, \tag{3.5}$$
$$\tilde{\boldsymbol{s}}^{(o)} = 2^{-1}(\mathbf{W}_{n/2}\,\boldsymbol{\Omega})\cdot(\Delta_{(l)} - \Delta_{(r)}) + \boldsymbol{s}^{(o)}. \tag{3.6}$$

Looking at the form of the sub-instance given by the "positive fold" (Equation (3.5)), it is clear that we can run a further divide and conquer step to reduce the dimension

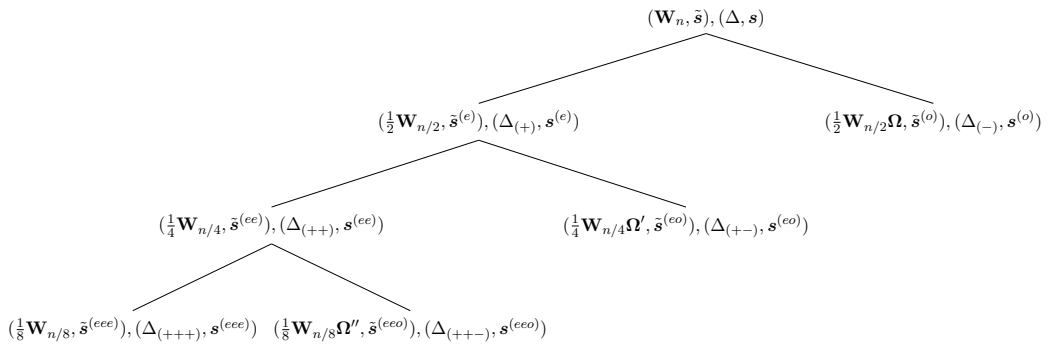


Figure 3.1: Recursive folding/dimension reduction.

further. In fact, we can repeatedly divide and conquer the positive fold to reach any dimension we wish as illustrated in Figure 3.1.

Considering, the "negative fold", the additional scaling matrix $\boldsymbol{\Omega}$ prevents us from folding down further. However, we note that on the lowest level, the attacker may still solve the negative branch.

Remark 5. *Note that we can also attempt to divide and conquer on the inverse NTT directly in the hope of obtaining sub-instances with error terms of the form* $\boldsymbol{s}_{(l)} \pm \boldsymbol{s}_{(r)}$ *and secrets* $\Delta^{(e)}$ *or* $\Delta^{(o)}$*. Yet, when attempting to do this for the* negacyclic *NTT, we actually obtain sub-instances with errors of the form* $\boldsymbol{s}_{(l)} + \omega^{\pm n/4}\boldsymbol{s}_{(r)}$ *which are not guaranteed to be small. However, these instances are still susceptible to lattice attacks for limited folding levels.*

A drawback of reducing to an extremely small dimension is that the "secret term" (i.e. the analogue of Δ) becomes less sparse at each level, eventually to the point that its distribution approaches the uniform distribution. Nonetheless, performing only a limited number of folding steps can preserve sparsity. This is because if $\Delta := (\Delta_{(l)}, \Delta_{(r)})$ is *very* sparse, then $\Delta_{(l)} \pm \Delta_{(r)}$ is still expected to be sparse (albeit not as sparse as Δ) and of the same Hamming weight as Δ when written in minimal BSDR. We will see later that a sparse minimal BSDR is the key to our lattice-based attack for solving sub-instances, so reducing to trivial dimension would be detrimental to our cold boot attack.

3.5.2 Extending Solutions to Sub-Instances

We now show how to derive a solution to an n-dimensional instance given an oracle that solves just one of its child instances in dimension $n/2$. We instantiate such an oracle in Sections 3.5.3 and 3.5.4.1.

We first note that given the solution to one of the sub-instances, we can derive a solution to its neighbour instance (c.f. Figure 3.1). First assume that the minimal BSDR (or possibly elements of a minimal BSDR list) of $\Delta_{(l)} + \Delta_{(r)}$ has Hamming weight equal to that of Δ. In other words, assume that there was no decrease in minimal BSDR Hamming weight when performing the positive fold. Then each bit set in the minimal BSDR of $\Delta_{(l)} + \Delta_{(r)}$ (or the single correct element of the BSDR list) originate from either $\Delta_{(l)}$ or $\Delta_{(r)}$. Therefore, in order to guess $\Delta_{(l)} - \Delta_{(r)}$, we simply flip some bits in the minimal BSDR of $\Delta_{(l)} + \Delta_{(r)}$. We then check the correctness of the guess by substituting the value of $\Delta_{(l)} - \Delta_{(r)}$ back into the neighbour instance. Note that the list of minimal BSDRs is expected to be relatively short. For example, of the integers $\{1, \ldots, 7680\}$, less than 4.92% have a BSDR list length of 4 or more when considering 13-bit representations. The maximum BSDR list length observed for these integers is 21 and occurs just 4 times. Since we will typically be encountering integers with low Hamming weight minimal BSDR, the length of the minimal BSDR lists ought to be shorter than suggested by these figures over $\{1, \ldots, 7680\}$.

If the Hamming weight of the minimal BSDR of $\Delta_{(l)} + \Delta_{(r)}$ is different to that of Δ, it has decreased with very high probability.[3] For example, assume that each $\mathbb{Z}_q$-component of $\Delta_{(l)}$ is the result of at most a single bit-flip. Assume the same for $\Delta_{(r)}$. Performing a fold in such a case would mean that each $\mathbb{Z}_q$-component of $\Delta_{(l)} + \Delta_{(r)}$ is the result of at most two bit-flips. Therefore the minimal BSDR of *each component* should have Hamming weight *at most* 2. In what follows, a sum "$a + b$" for $a, b \in \mathbb{Z}_q$ is intended to represent a single entry of a folding $\Delta_{(l)} + \Delta_{(r)}$ where a is from $\Delta_{(l)}$ and b is from $\Delta_{(r)}$. Under this assumption, there are two cases where the Hamming weight decreases by 1:

[3] Modular reduction by q may increase the Hamming weight, but this case occurs so infrequently that we ignore it here.

(a) Two bits with the same sign and position collide after folding e.g., $(1+1) = 2, (-1-1) = -2$

(b) Two bits with opposite signs appear in consecutive positions after folding e.g., $(-1+2) = 1, (2-1) = 1, (1-2) = -1, (-2+1) = -1$.

The Hamming weight can also decrease by 2 if two bits with opposite signs collide e.g., $(1-1) = 0, (-1+1) = 0$.

In light of these observations, we can still use a combinatorial approach to derive $\Delta_{(l)} - \Delta_{(r)}$ from $\Delta_{(l)} + \Delta_{(r)}$ even when folding caused the Hamming weight to decrease. For now, assume that the Hamming weight κ of Δ is known,[4] let κ' denote the Hamming weight of $\Delta_{(r)} + \Delta_{(l)}$ and ignore the small factors arising from the non-uniqueness of the minimal BSDR. We perform one of the following three guessing strategies depending on $\kappa - \kappa'$:

0: Flip signs of $\Delta_{(l)} + \Delta_{(r)}$ to guess $\Delta_{(l)} - \Delta_{(r)}$; $2^{\kappa'}$ guesses required.

1: Assume the Hamming weight decreased by 1 when folding due to either case (a) or (b) above; $3\kappa' \cdot 2^{\kappa'-1}$ guesses required.

2: Assume the Hamming weight decreased by 2 due to a single collision in bits with opposing signs; at most $(n/2 \cdot \lceil \log(q) \rceil - \kappa) \cdot 2 \cdot 2^{\kappa}$ guesses required

Note that the $3\kappa'$ factor arises because we must choose one out of the κ' bits that directly resulted from the Hamming weight decrease, and there are at most three ways that this spurious bit occurred. For example, suppose the spurious bit represented the integer 2. Then it could be that this value arose from the $(1+1), (4-2)$ or $(-2+4)$. The $(n/2 \cdot \lceil \log(q) \rceil - \kappa)$ factor arises in the third case because we must choose a 0 bit that arose from a collision and there are at most $(n/2 \cdot \lceil \log(q) \rceil - \kappa)$ zeros that are set to 0. There is a chance that this guessing approach fails. In order to increase the probability of success, we would have to perform additional guessing phases where we try to correct multiple spurious bits assuming various configurations. However, our experimental results below show that performing the three guessing phases above already yields a good probability of success. We also

[4] While this does not hold in a real cold boot attack, we will discuss below how to handle this.

note that in a cold boot attack the exact value of κ is not known. In this case, the attacker starts by assuming $\kappa = \kappa'$, followed by $\kappa = \kappa' + 1$ and $\kappa = \kappa' + 2$. This is sufficient to achieve a high rate of success.

An attacker may also directly solve the problem of the neighbour branch $\Delta_{(l)} - \Delta_{(r)}$. Indeed, given $\Delta_{(l)} + \Delta_{(r)}$, we can eliminate either $\Delta_{(l)}$ or $\Delta_{(r)}$ from the neighbour instance to obtain a problem in either $\Delta_{(l)}$ or $\Delta_{(r)}$. This new problem will have associated Hamming weight roughly $\kappa/2$. Furthermore, since $\kappa < n$ there is a very high probability that a known value $(\Delta_{(l)})_i + (\Delta_{(r)})_i = 0$ is indeed the result of adding $(\Delta_{(l)})_i = 0$ and $(\Delta_{(r)})_i = 0$. Thus, the dimension of the neighbour instance can be further reduced by eliminating those components, producing a relatively easy instance.

Combining the solutions from the two neighbour instances yields a solution for the parent instance. Thus, a solution in dimension n, implies a solution in dimension $2n$ which can then be extended to solutions in $4n, 8n, \ldots$ using the simple guessing approach above. The overall divide and conquer strategy can be summarised as follows:

1. Repeatedly divide and conquer the positive fold until a desired target dimension n' has been reached.
2. Solve the bottom (positive fold) instance, cf. Section 3.5.4.1.
3. (a) Given a solution to the positive fold, guess the solution to the negative fold and work the solution upwards. This costs in the order of[5]
$$\max\left(2^{\kappa}, \quad 3(\kappa-1)2^{\kappa-2}, \quad (n'/2 \cdot \lceil \log(q) \rceil - (\kappa-2)) \cdot 2 \cdot 2^{\kappa-2}\right)$$
operations multiplied by the number of folds.
   (b) If guessing fails, solve the negative instance directly, using partial information about $\Delta_{(l)}$ or $\Delta_{(r)}$.
4. Repeat the previous step until the full solution is recovered.

Table 3.1 uses Kyber parameters with $\kappa = 19$ bits flipped to give an overview of how the Hamming weight of Δ evolves as we fold multiple times. Assuming two

[5] Once again, we ignore the small factor arising from the non-uniqueness of the minimal BSDR.

folds, this shows a rough success rate of 74% when only considering the trivial ($\kappa - \kappa' = 0$) phase of guessing to work a 64-dimensional solution upwards. However, when all three phases of guessing are used, we empirically estimate that the success probability is around 97% when working a solution up from dimension 64. The corresponding success probability with $\kappa = 25$ is 94%. These values were obtained by sampling 1,000 random vectors Δ with minimal BSDR of Hamming weight $\kappa = 19$ and 25 and then analysing the cause of a decrease in Hamming weight whenever this occurred. A breakdown of the statistics of 1000 trials at the 128 to 64-dimensional fold are shown in Tables 3.2 and 3.3. In particular, we include how many times the Hamming weight decreases by 0,1 and 2 as well as how many of these are solvable in the three simple guessing phases described above. We also report success rates of 98% and 96% for solving this particular fold for $\kappa = 19$ and $\kappa = 25$ respectively. We reiterate that even when the simple guess-and-verify algorithm presented here fails, we expect to be able to solve the neighbour branch by making use of partial information about Δ_l or Δ_r. Thus, from now on, we will assume that the aspect of our attack introduced in this section always succeeds.

Table 3.1: The preservation rate of the Hamming weight of Δ on folding multiple times for $\kappa = 19$ cold boot flips on Kyber parameters.

Folds	Bottom level dimension	Hamming weight preserved
1	128	90.4%
2	64	73.5%
3	32	48.3%
4	16	19.1%

Table 3.2: A breakdown of the statistics on the 128 to 64 dimensional fold on 1000 Kyber cold boot instances ($\kappa = 19$) when carrying out the three guessing phases. The "Solvable" row indicates how many of the instances in each category are solvable by the three guessing phases.

	No decrease	Decrease by 1	Decrease by 2
Frequency	824	119	45
Solvable	824	119	39
Success rate	100%	100%	87%

Overall success rate for fold: 98.2%

What remains to be established is how to solve one or both of the bottom level instances; this is the subject of the following sections.

Table 3.3: The analogous statistics to those in Table 3.2 for $\kappa = 25$. For details on the table entries, see the caption for Table 3.2.

	No decrease	Decrease by 1	Decrease by 2
Frequency	714	174	94
Solvable	714	173	70
Success rate	100%	99%	74%

Overall success rate for fold: 95.7%

3.5.3 Lattice Formulation

Our algorithm for solving the bottom level instance after applying repeated folding is inspired by the normal form of the primal attack on LWE . At a high level, the aim of this attack is to construct a lattice Λ which contains a vector $\boldsymbol{v}$ closest to $(\mathbf{0}, \tilde{\boldsymbol{s}})$, such that the offset between $\boldsymbol{v}$ and $(\mathbf{0}, \tilde{\boldsymbol{s}})$ is $(\Delta, \boldsymbol{s})$. Then, finding this unique closest vector $\boldsymbol{v}$ to $(0, \tilde{\boldsymbol{s}})$ allows to recover $(\Delta, \boldsymbol{s})$. The success of this attack depends on $\boldsymbol{v}$ being the *unique* closest vector. Heuristically, we can expect the attack to work if $(\Delta, \boldsymbol{s})$ is shorter than the shortest vector in Λ.[6] Looking at our instance in Equation (3.2), our "secret term" (interpreting the instance as LWE) is the vector Δ, which is not guaranteed to have small norm, but *is* guaranteed to be sparse. Note that we abuse notation slightly here and let Equation (3.2) refer to the bottom level instance after folding, i.e. $\xi > 1$ and Δ is a vector obtained by repeated folding. Note that this setting is somewhat similar to that considered in [25, 47]. Now, since we know that the component-wise minimal BSDR of Δ will be small in norm, the idea is to construct a lattice resembling the primal attack lattice with an offset vector containing the minimal BSDR of Δ in its components.

In fact, we will generalise this idea to construct a lattice with the 2^ℓ-ary signed digit representation of Δ as an offset. Let $b = \lceil \log_{2^\ell} q \rceil$ and $\Delta^{(\ell)} \in \mathbb{Z}^{nb}$ be the vector where all components of Δ are expanded in the 2^ℓ-ary signed digit representation of minimal *norm*, i.e. we consider 2^ℓ-SDR. Concretely, for Kyber the reader may assume $\ell = 7$ and thus $b = 2$. Now, let $\mathbf{W}^{(\ell)} = \mathbf{W} \otimes (1, 2^\ell, \ldots, 2^{(b-1)\ell}) \in \mathbb{Z}^{n \times nb}$ and $\theta \in \mathbb{Q}$ be some rational scaling factor. We take as our lattice

$$\Lambda := \{\boldsymbol{x} \in \mathbb{Z}^{nb} \times \mathbb{Q}^n : \left[\mathbf{W}^{(\ell)} | (1/\theta)\mathbf{I}_n\right] \cdot \boldsymbol{x} \equiv \mathbf{0} \bmod q\}. \tag{3.7}$$

[6] The attack might still succeed even if Λ contains shorter vectors if these vectors are fairly orthogonal to the offset vector $(\Delta, \boldsymbol{s})$.

Concretely, a basis for this $(nb+n)$-dimensional lattice can be constructed from the rows of

$$\mathbf{B} = \begin{pmatrix} \mathbf{I}_{bn\times bn} & \theta\,(\mathbf{W}^{(\ell)^T}) \\ 0 & q\,\theta\,\mathbf{I}_{n\times n} \end{pmatrix}$$

where $(\cdot)^T$ denotes a transpose. Our aim is that $\boldsymbol{v} := (\mathbf{0}, \theta\tilde{\boldsymbol{s}}) - (\Delta^{(\ell)}, \theta\boldsymbol{s}) \in \Lambda$ is the closest lattice vector to $(\mathbf{0}, \theta\tilde{\boldsymbol{s}})$. To estimate whether this is the case, we need to estimate the norm of the offset vector $\|(\Delta^{(\ell)}, \theta\boldsymbol{s})\|$ and the length of the shortest vector in Λ denoted by $\lambda_1(\Lambda)$.

For primal attacks analysis on LWE, $\lambda_1(\Lambda)$ is estimated using the Gaussian heuristic. This is well justified for the LWE case where $\mathbf{A}$ is a uniformly random matrix mod q. However, the tensor product in $\mathbf{W}^{(\ell)}$ means that there are two classes of unusually short vectors in Λ. The first class contains vectors of the form

$$(0, \ldots, 0, 2^\ell, -1, 0, \ldots, 0)$$

where the last n components are 0 and the 2^ℓ and 1 belong to the same chunk of b entries. This vector essentially "undoes" the tensor product, producing zero in the part corresponding to $\mathbf{W}^{(\ell)}$. This vector has norm $\approx 2^\ell$, e.g. 128 in our Kyber-based running example.

The second class of fairly short vectors is given in terms of the 2^ℓ-ary signed digit representation of q that has minimum norm, which we denote as $\boldsymbol{q}^{(\ell)} \in \mathbb{Z}^b$. Explicitly, the second class of vectors are of the form

$$(0, \ldots, 0, \boldsymbol{q}^{(\ell)}, 0, \ldots, 0)$$

where b divides the number of leading zeros and the last n components are 0. For example, for $\ell = 7$, we can write $q = 7681$ as $60\cdot 2^{128}+1$ implying our lattice contains vectors of the form $(0, \ldots, 0, 60, 1, 0, \ldots, 0)$ of norm ≈ 60.

In addition to these short vectors, we must consider the expected length of the shortest vector in Λ ignoring such unusually short vectors. We will denote this length as $\lambda_1'(\Lambda)$. As mentioned above, if $\mathbf{W}$ were uniformly random, we could follow the usual strategy and consider the Gaussian heuristic to estimate this norm as:

$$\lambda_1'(\Lambda) \coloneqq \sqrt{\frac{n+nb}{2\pi e}}\,(\theta q)^{n/(n+nb)}.$$

However, as we will discuss in Section 3.5.4.1 the Gaussian Heuristic does not hold in our case, even when ignoring the vectors discussed above. Thus, we will establish $\lambda_1'(\Lambda)$ empirically using strong lattice reduction.

Now, we expect that the unique vector $\boldsymbol{v} \in \Lambda$ closest to $(\mathbf{0}, \theta\,\tilde{\boldsymbol{s}})$ satisfies $\boldsymbol{v} + (\Delta^{(\ell)}, \theta\boldsymbol{s}) = (\mathbf{0}, \theta\,\tilde{\boldsymbol{s}})$ when the following three conditions are all met:

$$\|(\Delta^{(\ell)}, \theta\boldsymbol{s})\| < \begin{cases} \sqrt{2^{2\ell}+1} \approx 2^{\ell} \\ \|\boldsymbol{q}^{(\ell)}\| \\ \lambda_1'(\Lambda). \end{cases}$$

We note that the above conditions by themselves do not imply that it is *efficient* to recover the appropriate closest vector.

In order to use the above inequalities, we need to estimate the expected length of the vector $(\Delta^{(\ell)}, \theta s)$. Assuming $\kappa \ll n$ bit-flips and $(\rho_0 + \rho_1) \cdot \log_2 q \ll 1$ (so that each non-zero component of Δ is with high probability the result of a single bit-flip), we have that $\|\Delta^{(\ell)}\|^2 \approx \kappa\,\frac{4^\ell - 1}{3\,\ell}$, cf. Proposition 1. We then expect that

$$\mathbb{E}\left[\|(\Delta^{(\ell)}, \theta s)\|^2\right] = \kappa\,\frac{4^\ell - 1}{3\,\ell} + n\,\theta^2\,\sigma^2 \tag{3.8}$$

where σ is the standard deviation of the secret distribution.

Example 3. *To carry out the analysis for Kyber, we pick $\ell = 7$ which means $\|\boldsymbol{q}^{(\ell)}\|^2 = 3601$ and $\lceil \log_{2^\ell}(q) \rceil = 2$. Thus, we heuristically require our offset vector to have squared norm $< \min(16385, 3601)$. Even picking a very small θ and ignoring the third condition above, this implies that we can only satisfy our constraints for $\kappa \leq 15$.*

3.5.4 A Guessing Strategy

To shorten the distance between the lattice and our target vector (i.e. $\|(\Delta^{(\ell)}, \theta\boldsymbol{s})\|$) we can simply guess the bits of Δ that contribute most significantly to the norm of $\Delta^{(\ell)}$.[7] To formalise this approach, we define a "band size" β that describes which bits we consider as contributing significantly to $\Delta^{(\ell)}$. For example, suppose we choose

[7] Of course, other guessing strategies are possible. For example, for sufficiently small κ we may have Δ sparse even mod q. An attacker might thus attempt to guess which columns of Δ can be ignored in an attack.

some $\ell \geq 2$ and a band size of $\beta < \ell$. Then we consider the top β bits of each entry in $\Delta^{(\ell)}$ (written in minimum Hamming weight BSDR) as being significant.

We can decompose $\Delta^{(\ell)} \in \mathbb{Z}_{nb}$ into two parts: $\Delta^{(\ell,\uparrow)}$ (the vector arising from the bits in the significant band) and $\Delta^{(\ell,\downarrow)}$ (the vector arising from the non-significant band). In doing so, we can write $\Delta^{(\ell)} = \Delta^{(\ell,\uparrow)} + \Delta^{(\ell,\downarrow)}$.

Our "guessing approach" is simply to guess $\Delta^{(\ell,\uparrow)}$ and use the basic primal attack to find the short vector $\Delta^{(\ell,\downarrow)}$. Note that assuming sparsity, the norm of $\Delta^{(\ell,\downarrow)}$ is smaller than that of $\Delta^{(\ell)}$ so it is more likely that the primal attack will succeed. More concretely, once we have guessed $\Delta^{(\ell,\uparrow)}$, we define $\tilde{\boldsymbol{s}}^{(\downarrow)} := \tilde{\boldsymbol{s}} - W^{(\ell)}\Delta^{(\ell,\uparrow)}$ and target offset vector $(\Delta^{(\ell,\downarrow)}, \theta\boldsymbol{s})$.

Now to investigate when $(\Delta^{(\ell,\downarrow)}, \theta\boldsymbol{s})$ is likely to be the offset to the unique closest vector to $(\mathbf{0}, \tilde{\boldsymbol{s}}^{(\downarrow)})$ in Λ, we begin by assuming some fixed ℓ and $\beta < \ell$ and calculating the expected length of $\Delta^{(\ell,\downarrow)}$. For every individual entry of $\Delta^{(\ell)}$, there are $\ell - \beta$ bits in the non-significant band and β bits in the significant band. Therefore, assuming κ bit-flips in total, we would expect roughly $\frac{\ell-\beta}{\ell}\kappa$ bit-flips[8] in $\Delta^{(\ell,\downarrow)}$. Assuming $\kappa \ll n$ (i.e. sparsity of bit-flips), we expect that

$$\mathbb{E}\left[\|(\Delta^{(\ell,\downarrow)}, \theta\boldsymbol{s})\|^2\right] = \frac{\ell-\beta}{\ell}\,\kappa\,\frac{4^{\ell-\beta-1}-1}{3\,(\ell-\beta-1)} + n\,\theta^2\,\sigma^2. \tag{3.9}$$

At this point, we can reuse the three success conditions detailed above, as the characteristic properties of Λ remain unchanged. We refer to the process of removing the top-most bits of a vector as "shaving". This process is parametrised by a band size β and a maximum number of bits to correct, α. Setting α to be less than the expected number of bits set in the top band has the advantage of yielding a smaller number of potential guesses available, but there is also the disadvantage that there may still be a few non-zero bits left in the top band. If there are some bits still set in the top band, then the candidate vector $\Delta^{(\ell,\downarrow)}$ may still be too long for a successful primal attack. The number of possible guesses for the top band with at most α bits flipped is

$$\sum_{i=0}^{\alpha} 2^i \cdot \binom{\#\text{bits in significant band}}{i} \tag{3.10}$$

[8]The number of expected bit-flips is actually less than this for some parameters (see Example 4 below).

Table 3.4: The maximum possible κ handled by each guessing band size β for Kyber parameters and the cost of guessing the significant band.

		guessing cost for $\kappa = 19$	
β	max κ	$n = 16$	$n = 32$
0	15		
1	52	$2^{9.0}$	$2^{11.0}$
2	169	$2^{25.8}$	$2^{30.9}$

where the factor 2^i takes care of the fact that each set bit-flip takes values in $\{-1, 1\}$ when multiple folding steps have been performed. If we have not folded, the factor of 2^i may be omitted since the sign of the bit-flips are known.

Example 4. *Returning to our example of Kyber, we analyse the case $\ell = 7$ again. Firstly, there are $256 \cdot 2\,\beta$ bits in the significant band. Note the factor of 2 due to the fact that each element of $\mathbb{Z}_{7681}$ requires two integers when written in base 2^7. However, since $7681 < 2^{13}$, the top most bit of each element of $\mathbb{Z}_{7681}$ must be 0. This leaves $256 \cdot (2\beta - 1)$ unknown bit positions where we must correct bit-flips. There is an average of $\frac{2\beta-1}{13} \cdot \kappa$ bit-flips in the unknown part of the significant band. The maximum κ such that (3.9) $< 3601 = \|q^{(\ell)}\|$ with θ arbitrarily small, i.e. we are ignoring the second summand in (3.9), is given in Table 3.4. We use Equation (3.10) with α set to the expected number of bit-flips to estimate the number of guesses required for $\kappa = 19$ bit-flips in total.*

Even strategy. As illustrated in Example 4, the existence of vectors $\boldsymbol{q}^{(\ell)}$ is a main limiting factor for ensuring that our offset vector is sufficiently short. To remove this class of vectors from our lattice, we focus on resolving bit-flips in the least significant bits of the components of Δ. Assume for the moment that this has been achieved, and $\Delta_i \bmod 2 \equiv 0$ for all $0 \leq i < n$. Then, instead of considering $\mathbf{W} \otimes (1, 2^\ell, \ldots, 2^{(b-1)\ell}) \in \mathbb{Z}^{n \times nb}$ we may consider

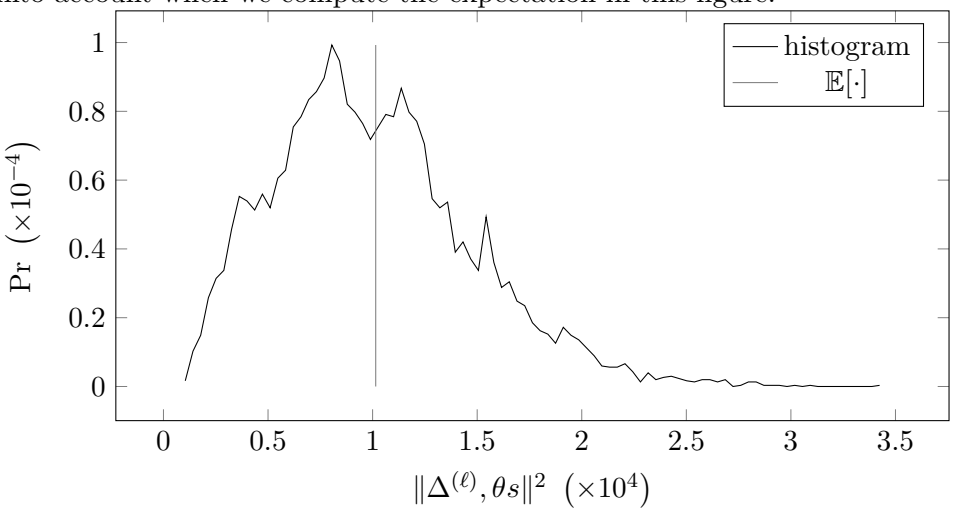
$$\mathbf{W} \otimes (2, 2^\ell, \ldots, 2^{(b-1)\ell}) \in \mathbb{Z}^{n \times nb},$$

i.e. scale rows $0, b, 2b, \ldots, nb$ of $\mathbf{B}$ by a factor of two. Since $q \bmod 2 \equiv 1$ we cannot write q as a linear combination of $2, 2^\ell, \ldots, 2^{(b-1)\ell}$. This removes the annoying vectors $\boldsymbol{q}^{(\ell)}$ from our lattice. To ensure $\Delta_i \bmod 2 \equiv 0$, as assumed here, we may apply a similar guessing strategy as discussed above. However, we note that this

comes with some additional cost for guessing and correcting the least significant bits of the components of Δ.

Finally, we stress that our analysis so far uses expected values throughout. In Figure 3.2, we plot an example histogram of the $\|(\Delta^{(\ell)}, \theta s)\|^2$ against our expectation for $\kappa = 19$ and $\theta = 3$. As illustrated in Figure 3.2, the actual observed distribution has a large variance. Thus, to estimate the cost of our attack, we will derive parameters from empirical evidence.

Figure 3.2: Histogram of observed squared norms of vectors of length $n = 256$ mod $q = 7681$, folded three times to dimension 32, written in base 2^7, for $\theta = 3$ and $\kappa = 19$. Note that in this example $\mathbb{E}[\|(\Delta^{(\ell)}, \theta s)\|^2] < \kappa \frac{4^\ell - 1}{3\ell} + n\,(\theta\sigma)^2$ because $\log_2 q < 14$. Thus, half of our entries are bounded by 2^6 instead of 2^7 This is taken into account when we compute the expectation in this figure.



3.5.4.1 BDD on NTT lattices

So far, we have only analysed the existence of a unique closest vector to our target. The last ingredient of our attack is to *find* this vector, i.e. a vector in

$$\Lambda := \{\boldsymbol{x} \in \mathbb{Z}^{n\lceil \log_{2^\ell} q \rceil} \times \mathbb{Q}^n : \left[\mathbf{W}^{(\ell)}(1/\theta)\mathbf{I}_n\right] \cdot \boldsymbol{x} \equiv \mathbf{0} \bmod q\}$$

that is close to $(\mathbf{0}, \theta\tilde{\boldsymbol{s}})$. Concretely, for Kyber we set $\ell = 7$ and $n = 32$, where $n > 16$ is chosen to preserve sparsity of BSDRs for $\rho_0 = 1.0\%, \rho_1 = 0.1\%$, where we expect $\kappa = 19$ bit-flips. To consider the geometry of the lattice spanned by our instances, consider the smaller case $n = 4$, $\theta = 1$ (since it fits on this page). We obtain the

q-ary lattice basis

$$\mathbf{B} = \begin{pmatrix} 1 & & & & & & & & 1 & -\omega^3 & -\omega^2 & -\omega \\ & 1 & & & & & & & 2^7 & -2^7\omega^3 & -2^7\omega^2 & -2^7\omega \\ & & 1 & & & & & & 1 & -\omega & \omega^2 & -\omega^3 \\ & & & 1 & & & & & 2^7 & -2^7\omega & 2^7\omega^2 & -2^7\omega^3 \\ & & & & 1 & & & & 1 & \omega^3 & -\omega^2 & \omega \\ & & & & & 1 & & & 2^7 & 2^7\omega^3 & -2^7\omega^2 & 2^7\omega \\ & & & & & & 1 & & 1 & \omega & \omega^2 & \omega^3 \\ & & & & & & & 1 & 2^7 & 2^7\omega & 2^7\omega^2 & 2^7\omega^3 \\ & & & & & & & & 7681 & & & \\ & & & & & & & & & 7681 & & \\ & & & & & & & & & & 7681 & \\ & & & & & & & & & & & 7681 \end{pmatrix}.$$

where all of the omitted entries are zero. Note that the lattice spanned by $\mathbf{B}$ contains the unusually short vector

$$(1, 0, 1, 0, 1, 0, 1, 0, 4, 0, 0, 0). \tag{3.11}$$

This vector is not an artefact of the tensor product but an artefact of $\mathbf{B}$ being derived from an NTT matrix: it corresponds to folding all the way down to dimension $n = 1$.

More generally, the geometry of the q-ary lattices Λ considered in this work is far from what we would expect from a random q-ary lattice. In Figure 3.3, we plot the lengths of the Gram-Schmidt vectors of a BKZ-90 reduced basis for a lattice Λ corresponding to folding our 256-dimensional instance down to dimension $n = 32$. This lattice has dimension $96 = \lceil \log_{2^7} q \rceil \cdot n + n$. For comparison, we also plot the expected lengths of the Gram-Schmidt vectors according to the Geometric Series Assumption which approximates the behaviour of random q-ary lattices reasonably well.

Due to this unusual geometry, we cannot readily apply standard estimates for lattice reduction. As a case in point, computing a BKZ-90 reduced basis of the 96-dimensional lattice in Figure 3.3 took less than an hour with FPLLL [55], i.e. reducing this basis is considerably faster than expected for random q-ary lattices.

Thus, to find the vector $\boldsymbol{v} \in \Lambda$ closest to $(\mathbf{0}, \theta\tilde{\boldsymbol{s}})$, we proceed as follows. First, we remove the unusually short vector given in Equation (3.11). This is accomplished by guessing the value of Δ_0 and considering the sublattice spanned by the rows of Λ except for the first $\lceil \log_{2^\ell} q \rceil$ rows. Pessimistically, we expect that this increases our guessing cost by a factor of $\lceil \log_2 q \rceil$. We refer to this smaller basis as $\mathbf{B}'$ and

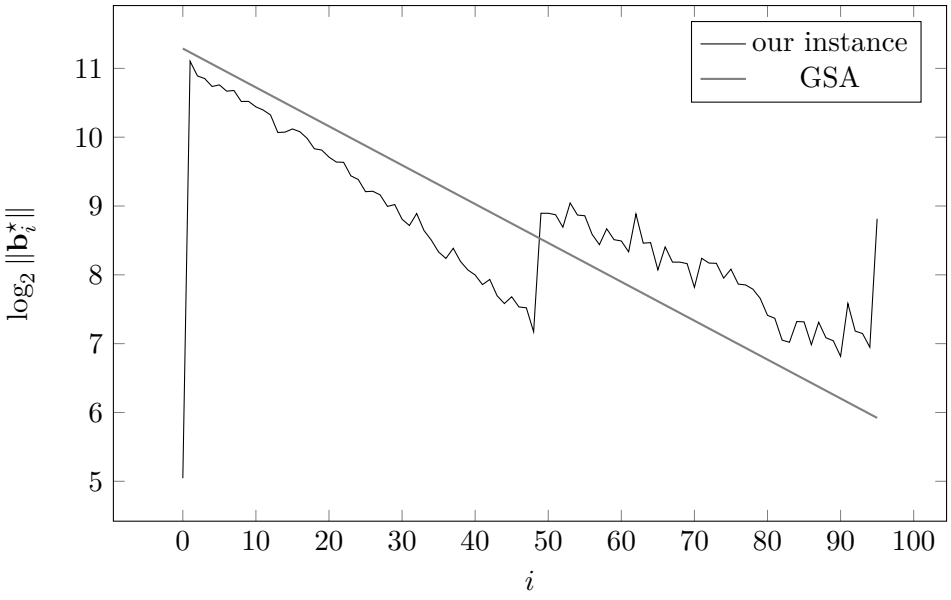


Figure 3.3: Length of Gram-Schmidt vectors in q-ary lattice derived from negacyclic inverse NTT at dimension 32 using parameters $\ell = 7, \theta = 1$.

call d the dimension of the lattice spanned by $\mathbf{B}'$. Then, we compute a high-quality basis for the lattice spanned by $\mathbf{B}'$. In particular, for $n = 32$ we compute a BKZ-90 reduced basis. Then, for each guess as in Section 3.5.4, we perform one pruned BDD enumeration in dimension $\mathsf{bs} = \min(60, d)$, i.e. the bs-dimensional sub-lattice orthogonal to the first $d - \mathsf{bs}$ vectors in $\mathbf{B}'$. We heuristically expect that BDD enumeration in block size bs will find the closest vector iff the projection of the offset vector orthogonal to the first $d - \mathsf{bs}$ vectors in $\mathbf{B}'$ is shorter than $\boldsymbol{b}^\star_{d-\mathsf{bs}}$, the Gram-Schmidt vector at index $d - \mathsf{bs}$ in $\mathbf{B}'$ [10, 8]. As in [10, 8], we assume that the length of this projection is

$$\sqrt{\frac{\mathsf{bs}}{d}}\,\mathbb{E}\left[\|\Delta^{(\ell)}, \theta\boldsymbol{s}\|\right]$$

where $\mathbb{E}[\|(\Delta^{(\ell)}, \theta\boldsymbol{s})\|]$ is experimentally established by sampling 1024 vectors.[9] As enumeration radius we pick

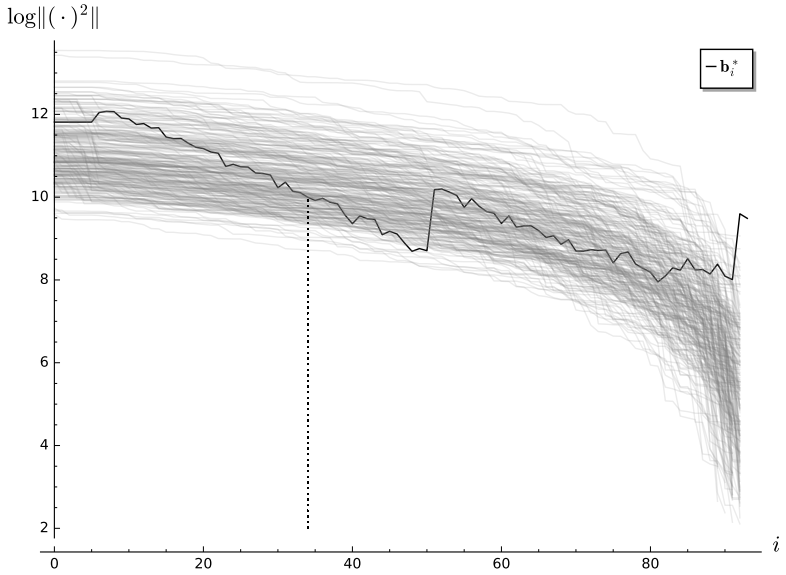
$$\min\left(\sqrt{\frac{\mathsf{bs}}{d}} \cdot \mathbb{E}[\|(\Delta^{(\ell)}, \theta\boldsymbol{s})\|],\ \|\boldsymbol{b}^\star_{d-\mathsf{bs}}\|\right). \tag{3.12}$$

The right-hand argument in (3.12) takes care of the fact that there is little point in enumerating beyond the length of the shortest vector in the projected sub-lattice if

[9] In our experiments, the approximation $\sqrt{\frac{\mathsf{bs}}{d}}\,\mathbb{E}\left[\|\Delta^{(\ell)}, \theta\boldsymbol{s}\|\right]$ indeed appears reasonably accurate.

we are targeting a unique closest vector. We illustrate the expected behaviour in Figure 3.4, where we plot the projected norms for 256 samples of $(\Delta^{(\ell)}, \theta \boldsymbol{s})$ against the norms of the Gram-Schmidt vectors for our reduced basis $\mathbf{B}'$ for $\theta = 3$. Note that in contrast to Figure 3.3, the basis in Figure 3.4 is $\mathbf{B}'$ and not $\mathbf{B}$. We expect enumeration to succeed for every grey line that stays below the Gram-Schmidt vectors for all indices $< d - \mathsf{bs}$. Figure 3.4 illustrates that we can improve our probability of success by increasing the enumeration dimension at the cost of increasing the running time. Note that the algorithm may still succeed when the heuristic success condition discussed above is not satisfied due to the orientation of the vectors involved. Therefore, we use the empirical evidence (cf. Tables 3.5-3.9) to establish the success rate.

Figure 3.4: Projected lengths for 256 samples of $(\Delta^{(\ell)}, \theta s)$ and the norms of the Gram-Schmidt vectors for our reduced basis B' for $\theta = 3$ with $\kappa = 19$, folded down three times to $n = 32$ and shaved with parameters $\alpha, \beta = 4, 2$. The dotted line indicates $d - \mathsf{bs}$, i.e. where we start enumerating.



The experiments we performed are as follows.[10] We sample random sparse binary vectors Δ in dimension n for various κ and construct a corresponding cold boot NTT decoding problem using secrets sampled from the Kyber secret distribution. We then folded this instance down to dimension $n = 32$ and simulated the guessing part of the algorithm for some parameters α, β. Since the cost of the guessing part of the attack is easy to predict, we simulated it by always picking the best "shaving" under

[10] Code available at `https://bitbucket.org/Amit_Deo/coldboot-ntt/`

the constraints imposed by α, β. This is implemented as the `shave` function. We then ran lattice point enumeration to recover the offset vector, this is implemented in the function `offset_vector`. We report success when the returned vector matches the norm of our target exactly and failure otherwise. In summary, we implemented the full attack on the $n = 32$ sub-problem except for the guessing part. We note that we also implemented and verified extending the solution upwards as described in Section 3.5.2.

We summarise the observed behaviour of our algorithm for solving the bottom-level $n = 32$ instance in Tables 3.5-3.9. These tables illustrate the trade off between the two pruned exhaustive search steps in our algorithm, the first searching for set higher-order bits, the second searching for lattice points. Increasing one reduces the other. Furthermore, according to our empirical evidence, the "even" strategy may provide a small gain in some cases, but it is not clearly more efficient than the "not even", i.e. "odd" strategy. All numbers in these tables were obtained using the proof of concept implementation in Sage [135]. To establish the cost of the enumeration, we use the number of nodes in the pruned enumeration tree as reported by the `Pruner` class from FPLLL/FPYLLL [55, 56]. Processing each node is generally assumed to take about 100 CPU cycles [55].

3.5.5 Putting It All Together

3.5.5.1 Kyber KEM

We now draw together Sections 3.5.1-3.5.4.1 to give a concise account of our attack and its performance on the Kyber KEM. Recall that we have 3 instances of the form $\tilde{\boldsymbol{s}} = \mathbf{W}_n \cdot \Delta + \boldsymbol{s}$ for a single private key in Kyber, with $n = 256$. We first establish some notation. Below, "label (n, m)" indicates an instance from Figure 3.1 having n coefficients/entries in $\boldsymbol{s}^{(\dots)}$ where the error term $\Delta_{(\dots)}$ is the sum of m original error terms Δ_j. Note that in Figure 3.1, the label of a node is given by the subscript in Δ.

root (256,1) This instance is in the secrets s_i for $i \in \{0, 1, 2, , \dots, n-1\}$ and has error Δ_j for the j-th equation. It corresponds to the root node in Figure 3.1.

Table 3.5: Experimental results for Kyber parameters and number of bit-flips $\kappa = 5$ ($\rho_0 = 0.2\%$, $\rho_1 = 0.1\%$); θ is the scaling factor of our lattice, α the number of bits we guess in a band of size β. In the "even" case we target the least significant bits of the components of Δ first. The column "guess" holds the number of guesses before lattice enumeration which includes the cost of guessing Δ_0, the column "enum" holds the number of nodes in the pruned lattice-point enumeration tree. The column "total" is the product of the two. All costs are give as $\log_2(\cdot)$. The column "rate" is the success rate over 200 experiments. Only parameters with success rate $\geq 60\%$ are shown. The minimal total cost is highlighted in bold and used in Table 3.10.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 5$, odd							$\kappa = 5$, even						
2	1	1	9.7	16.2	25.9	77.5%	2	1	1	10.7	11.1	21.8	74.5%
2	1	2	11.3	14.8	26.1	85.0%	2	1	2	11.7	9.4	**21.1**	78.5%
2	2	1	14.7	16.2	30.9	83.5%	2	2	1	16.7	10.7	27.4	87.5%
2	2	2	17.9	13.7	31.5	96.5%	2	2	2	18.7	6.0	24.7	94.0%
2	3	1	19.0	16.2	35.3	84.0%	2	3	1	22.1	10.5	32.6	90.5%
2	3	2	23.8	13.5	37.3	99.5%	2	3	2	25.1	5.7	30.8	98.5%
3	1	1	9.7	14.0	23.8	81.0%	3	1	1	10.7	11.6	22.4	79.5%
3	1	2	11.3	13.6	24.9	87.0%	3	1	2	11.7	10.3	22.0	81.0%
3	2	1	14.7	14.0	28.7	87.5%	3	2	1	16.7	11.3	28.0	92.0%
3	2	2	17.9	12.8	30.7	98.5%	3	2	2	18.7	7.8	26.5	96.0%
3	3	1	19.0	14.0	33.1	88.0%	3	3	1	22.1	11.2	33.3	94.5%
3	3	2	23.8	12.7	36.5	100.0%	3	3	2	25.1	7.6	32.7	99.5%

`+` (128, 2) This instance is the result of folding once on the plus branch. It is in the secrets s_i for $i \in \{0, 2, 4, \ldots, n-2\}$. The j-th equation has error term $\Delta_j + \Delta_{j+128}$.

`++` (64, 4) This instance is the result of folding twice on the plus branch. It is in the secrets s_i for $i \in \{0, 4, 8 \ldots, n-4\}$. The j-th equation has error term $\Delta_j + \Delta_{j+64} + \Delta_{j+128} + \Delta_{j+192}$.

`+++` (32, 8) This instance is the result of folding three times on the plus branch. It is in the secrets s_i for $i \in \{0, 8, 16, \ldots, n-8\}$. The j-th equation has error term $\Delta_j + \Delta_{j+32} + \Delta_{j+64} + \Delta_{j+96} + \Delta_{j+128} + \Delta_{j+160} + \Delta_{j+192} + \Delta_{j+224}$.

`++-` (32, 8) This instance the result of folding twice on the plus branch and once on the negative. It is in the secrets s_i for $i \in \{4, 12, 20, \ldots, n-4\}$. The j-th equation has error term $\Delta_j - \Delta_{j+32} + \Delta_{j+64} - \Delta_{j+96} + \Delta_{j+128} - \Delta_{j+160} + \Delta_{j+192} - \Delta_{j+224}$.

For each of our three independent instances comprising a full Kyber key, we perform

Table 3.6: Experimental results for Kyber parameters and $\kappa = 10$ ($\rho_0 = 0.5\%$, $\rho_1 = 0.1\%$). For details see Table 3.5.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 10$, odd							$\kappa = 10$, even						
							2	3	1	22.1	11.1	33.2	66.5%
2	3	2	23.8	16.2	40.0	93.0%	2	3	2	25.1	9.8	34.9	89.5%
							2	4	1	27.0	11.1	38.1	69.0%
2	4	2	29.4	16.2	45.6	97.5%	2	4	2	31.1	8.2	39.3	95.0%
							2	5	1	31.6	11.1	42.7	70.0%
2	5	2	34.6	15.9	50.5	99.5%	2	5	2	36.7	7.7	44.4	99.5%
3	3	1	19.0	14.0	**33.1**	63.5%	3	3	1	22.1	14.2	36.3	76.5%
3	3	2	23.8	14.0	37.9	95.5%	3	3	2	25.1	10.6	35.7	90.5%
3	4	1	22.9	14.0	37.0	64.0%	3	4	1	27.0	14.2	41.3	79.0%
3	4	2	29.4	14.0	43.4	98.0%	3	4	2	31.1	9.4	40.5	95.0%
3	5	1	26.5	14.0	40.5	64.0%	3	5	1	31.6	14.2	45.9	79.0%
3	5	2	34.6	14.0	48.6	98.5%	3	5	2	36.7	9.0	45.8	99.5%
4	3	1	19.0	20.9	40.0	64.0%	4	3	1	22.1	12.2	34.3	86.0%
4	3	2	23.8	19.5	43.4	87.5%	4	3	2	25.1	12.1	37.2	95.5%
4	4	1	22.9	20.9	43.9	64.5%	4	4	1	27.0	12.2	39.3	87.0%
4	4	2	29.4	19.1	48.5	91.5%	4	4	2	31.1	11.2	42.2	98.0%
4	5	1	26.5	20.9	47.4	64.5%	4	5	1	31.6	12.2	43.9	87.5%
4	5	2	34.6	18.8	53.4	92.0%	4	5	2	36.7	10.9	47.6	99.5%

Table 3.7: Experimental results for Kyber parameters and $\kappa = 19$ ($\rho_0 = 1.0\%$, $\rho_1 = 0.1\%$). For details see Table 3.5.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 19$, odd							$\kappa = 19$, even						
2	4	2	29.4	16.2	45.6	62.5%							
2	5	2	34.6	16.2	50.8	80.0%	2	5	2	36.7	11.1	47.8	70.5%
2	6	2	39.5	16.2	55.7	86.5%	2	6	2	42.1	11.1	53.1	84.0%
2	7	2	44.2	16.2	60.4	91.0%	2	7	2	47.2	11.1	58.3	90.5%
2	8	2	48.7	16.2	64.9	94.5%	2	8	2	52.1	10.6	62.8	96.0%
3	4	2	29.4	14.0	43.4	71.0%							
3	5	2	34.6	14.0	48.6	82.0%	3	5	2	36.7	14.2	50.9	74.0%
3	6	2	39.5	14.0	53.6	87.0%	3	6	2	42.1	13.3	55.4	86.0%
3	7	2	44.2	14.0	58.3	91.5%	3	7	2	47.2	12.2	59.4	93.0%
3	8	2	48.7	14.0	62.8	92.5%	3	8	2	52.1	11.8	63.9	98.0%
4	4	2	29.4	20.9	50.3	66.5%	4	4	2	31.1	12.2	**43.3**	70.5%
4	5	2	34.6	20.9	55.5	70.5%	4	5	2	36.7	12.2	48.9	83.0%
4	6	2	39.5	20.9	60.5	79.0%	4	6	2	42.1	12.2	54.3	89.0%
							4	7	1	40.0	12.2	52.2	60.0%
4	7	2	44.2	20.9	65.2	83.5%	4	7	2	47.2	12.2	59.4	96.5%
							4	8	1	43.8	12.2	56.1	60.5%
4	8	2	48.7	20.9	69.7	84.0%	4	8	2	52.1	12.2	64.4	97.5%

Table 3.8: Experimental results for Kyber parameters and $\kappa = 25$ ($\rho_0 = 1.4\%, \rho_1 = 0.1\%$). For details see Table 3.5.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 25$, odd							$\kappa = 25$, even						
2	6	2	39.5	16.2	55.7	60.5%							
2	6	3	44.0	16.2	60.2	68.0%							
2	7	2	44.2	16.2	60.4	72.5%	2	7	2	47.2	11.1	58.3	65.0%
2	7	3	49.5	16.2	65.7	82.5%	2	7	3	51.4	11.1	62.4	67.0%
2	8	2	48.7	16.2	64.9	79.0%	2	8	2	52.1	11.1	63.2	77.5%
2	8	3	54.8	16.2	71.0	90.5%	2	8	3	56.9	11.1	68.0	80.5%
3	5	3	38.3	14.0	**52.4**	60.0%							
3	6	2	39.5	14.0	53.6	70.0%							
3	6	3	44.0	14.0	58.1	74.0%							
3	7	2	44.2	14.0	58.3	76.0%	3	7	2	47.2	14.2	61.4	71.0%
3	7	3	49.5	14.0	63.5	81.5%	3	7	3	51.4	14.2	65.6	71.5%
3	8	2	48.7	14.0	62.8	80.0%	3	8	2	52.1	14.2	66.4	82.5%
3	8	3	54.8	14.0	68.8	87.0%	3	8	3	56.9	13.3	70.2	84.5%
4	6	2	39.5	20.9	60.5	60.5%	4	6	2	42.1	12.2	54.3	66.5%
4	6	3	44.0	20.9	65.0	68.0%	4	6	3	45.6	12.2	57.9	66.0%
4	7	2	44.2	20.9	65.2	69.0%	4	7	2	47.2	12.2	59.4	80.0%
4	7	3	49.5	20.9	70.4	80.5%	4	7	3	51.4	12.2	63.6	79.0%
4	8	2	48.7	20.9	69.7	71.0%	4	8	2	52.1	12.2	64.4	89.5%
4	8	3	54.8	19.5	74.3	84.0%	4	8	3	56.9	12.2	69.1	89.0%

Table 3.9: Experimental results for Kyber parameters and $\kappa = 30$ ($\rho_0 = 1.7\%, \rho_1 = 0.1\%$). For details see Table 3.5.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 30$, odd							$\kappa = 30$, even						
2	8	3	54.8	16.2	71.0	66.5%							
2	9	2	53.0	16.2	69.2	63.0%	2	9	2	56.9	11.1	67.9	65.5%
2	9	3	59.8	16.2	76.0	82.5%	2	9	3	62.3	11.1	73.3	66.5%
3	7	2	44.2	14.0	58.3	60.0%							
3	7	3	49.5	14.0	63.5	66.0%							
3	8	2	48.7	14.0	**62.8**	67.5%	3	8	2	52.1	14.2	66.4	65.0%
3	8	3	54.8	14.0	68.8	73.0%	3	8	3	56.9	14.2	71.1	65.0%
3	9	2	53.0	14.0	67.1	74.5%	3	9	2	56.9	14.2	71.1	73.5%
3	9	3	59.8	14.0	73.9	84.0%	3	9	3	62.3	14.2	76.5	75.0%
							4	7	2	47.2	12.2	59.4	64.5%
4	7	3	49.5	20.9	70.4	62.5%	4	7	3	51.4	12.2	63.6	64.5%
4	8	2	48.7	20.9	69.7	63.5%	4	8	2	52.1	12.2	64.4	72.0%
4	8	3	54.8	20.9	75.7	73.0%	4	8	3	56.9	12.2	69.1	72.0%
4	9	2	53.0	20.9	74.0	66.0%	4	9	2	56.9	12.2	69.1	79.5%
4	9	3	59.8	20.7	80.5	80.0%	4	9	3	62.3	12.2	74.5	79.0%

the following steps:

1. Divide and conquer three times to obtain two bottom level instances +++ and ++- as in Section 3.5.1.
2. Solve at least one bottom level instance using combinatorial and lattice-reduction techniques as in Sections 3.5.3 and 3.5.4.1. The cost and expected success rate for solving one such instance are given in Section 3.5.4.1. If solving one instance succeeds with probability p_0, we assume that this step succeeds with probability $1 - (1 - p_0)^2$, i.e. we assume the two bottom level instances are sufficiently different.
3. Substitute the solution obtained into the instance ++. This reduces it from $(64, 4)$ to $(32, 4)$, Solve this instance as in Sections 3.5.3 and 3.5.4.1. Note that solving this instance is much easier than in the previous step since the Hamming weight of the noise is reduced to $\approx \kappa/2$. We assume this step always succeeds.
4. Work the solution of ++ upward to + by solving +- using the information from ++ as in Section 3.5.2. This step succeeds with probability p_1 and we assume that it is cheaper than the previous steps.
5. Work the solution of + upward to "root" by solving - using the information from + as in Section 3.5.2. We assume this step always succeeds and we assume that it is cheaper than the previous steps.

Thus, the overall complexity of recovering 256 components of the Kyber secret is to run the lattice attack from Section 3.5.4.1 three times (steps 2 and 3) and succeeds with probability $\approx p_1 \cdot (1 - (1 - p_0)^2)$. In particular, for our choice of parameters we have[11] $p_1 \approx 1$ and $p_0 > 0.6$ and thus expect success with probability > 0.84. For example, with $\kappa = 19$, Table 3.7 shows that we can solve the hardest BDD problem with a cost of $2^{43.3}$ and success probability $p_0 = 0.705$. Since this is by far the most expensive stage of the attack, we report an attack cost of enumerating $\approx 2^{43.3}$ nodes in an enumeration tree where each node requires about 100 CPU cycles to process and a $p_1 \cdot (1 - (1 - p_0)^2) \approx 0.91$ success probability. We can attack each

[11] Note that it is easy to amplify p_1 by performing additional guessing phases in Section 3.5.2.

Table 3.10: Cold boot attacks on Kyber KEM keys stored in the NTT domain with ρ_0, ρ_1 the cold boot bit-flip rates. The column "cost" gives the cost of recovering 256 components of the secret in terms of the number of lattice points visited during enumeration ($\approx$ 100 CPU cycles each). The attack can be repeated to recover all 768 components. The column "rate" shows the overall success rate $1-(1-p_0)^2$ for recovering 256 components of the secret, cf. Section 3.5.4.1. We also give the costs of a cold boot attack when the secret key is stored in the time domain in the column "non-NTT", cf. Section 3.4. In that case, the success rate is always expected to be close to 100%.

bit-flip rates		NTT		non-NTT
ρ_0	ρ_1	cost	rate	cost
0.2%	0.1%	$3 \cdot 2^{21.1}$	95%	$2^{38.7}$
0.5%	0.1%	$3 \cdot 2^{33.1}$	87%	$2^{51.6}$
1.0%	0.1%	$3 \cdot 2^{43.3}$	91%	$2^{70.3}$
1.4%	0.1%	$3 \cdot 2^{53.6}$	91%	$2^{89.2}$
1.7%	0.1%	$3 \cdot 2^{62.8}$	89%	$2^{100.1}$

of the $d = 3$ module elements separately and combine the final solution. We note that the attacker can detect with high probability when a sub-solution is incorrect and thus invest more computational resources to increase the chance of success. We summarise our results in Table 3.10.

The attack needs to be run $d = 3$ times to recover a full Kyber secret. If a solution cannot be obtained for one of the three secret ring elements, then the solutions of the other two sub-problems can be substituted back into the original MLWE problem for Kyber's public key. This reduces the effective dimension of the public key to $n = 256$. An attacker could then target this smaller RLWE instance. Solving such an instance costs roughly 2^{77} according to the LWE estimator from [9], again assuming that enumeration is used to realise the SVP oracle inside BKZ. As suggested above, an attacker could alternatively attempt to re-run our cold boot attack on the remaining unknown secret element with different parameter choices from Tables 3.5-3.9. This would boost the probability of success at the expense of a greater computational cost.

3.5.5.2 New Hope KEM

We now move away from our MLWE-based example of Kyber KEM and give a concise account of the performance of our attack on the RLWE-based New Hope KEM [115]. The parameters used are $n = 1024, q = 12289$ and the secret polynomials have coefficients lying in the set $\{0, \pm 1, \ldots, \pm 8\}$. Similarly to Kyber KEM, New Hope uses an NTT to store its secret keys, meaning that we can launch the same cold boot attack. An important distinction between the Kyber and New Hope cases is that, for Kyber, we obtain multiple independent cold boot instances, each one corresponding to an individual polynomial in the secret key; this leads to multiple instances of relatively low dimension for Kyber. However, in the case of New Hope, we have just one cold boot instance in a large dimension. This distinction between MLWE- and RLWE-based schemes holds true in general for our cold boot attack in the NTT domain.

We focus our attention on the lattice aspect of the attack, assuming that we have folded the New Hope 1024-dimensional cold boot instance repeatedly to reach a 32-dimensional instance using the methods in Section 3.5.1. We can then experimentally estimate the success rate of solving this bottom level instance for various choices of θ, α, β using the methods in Section 3.5.4.1 with $b = 2$ and $\ell = 7$. The results for $\kappa = 10, 19, 25, 30$ are given in Tables 3.11– 3.14. Note that the value $\kappa = 19$ roughly corresponds to the limiting cold boot case of $\rho_0 = 0.17\%, \rho_1 = 0.1\%$ where liquid nitrogen is used to cool the RAM chip.

We now reuse the analysis and notation from Section 3.5.5.1 to estimate the running time and success probability of the full attack on New Hope. The success probability of the attack is $\approx p_1 \cdot (1 - (1 - p_0)^2)$ where p_1 is the success probability of working a bottom level solution up and p_0 is the probability of successfully solving a bottom level instance. Once again, we assume that this aspect of the attack can be performed successfully with probability $p_1 \approx 1$ without dominating the complexity of the overall attack. To determine p_0, we use the results form Tables 3.11-3.14. A summary of our results for $\kappa = 19, 25, 30$ are given in Table 3.15.

Table 3.11: Experimental results for New Hope parameters and number of bit-flips $\kappa = 10$; θ is the scaling factor of our lattice, α the number of bits we guess in a band of size β. In the "even" case we target the least significant bits of the components of Δ first. The column "guess" holds the number of guesses before lattice enumeration which includes the cost of guessing Δ_0, the column "enum" holds the number of nodes in the pruned lattice-point enumeration tree. The column "total" is the product of the two. All costs are give as $\log_2(\cdot)$. The column "rate" is the success rate over 100 experiments. Only parameters with success rate $\geq 50\%$ are shown. The minimal total cost is highlighted in bold and used in Table 3.15.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 10$, odd							$\kappa = 10$, even						
							2	3	1	23.8	10.3	**34.2**	60.9%
2	3	2	25.1	15.6	40.7	82.0%	2	3	2	26.1	10.3	36.4	73.4%
							2	4	1	29.4	10.3	39.7	73.4%
2	4	2	31.1	15.6	46.6	96.1%	2	4	2	32.4	10.3	42.7	93.0%
							2	5	1	34.6	10.3	44.9	77.3%
2	5	2	36.7	15.6	52.3	100.0%	2	5	2	38.3	10.3	48.7	97.7%
							3	3	1	23.8	11.1	35.0	63.3%
3	3	2	25.1	12.8	37.9	68.0%	3	3	2	26.1	11.1	37.2	71.9%
							3	4	1	29.4	11.1	40.5	76.6%
3	4	2	31.1	12.8	43.9	76.6%	3	4	2	32.4	11.1	43.5	91.4%
							3	5	1	34.6	11.1	45.7	78.1%
3	5	2	36.7	12.8	49.5	81.2%	3	5	2	38.3	11.1	49.4	93.0%

Table 3.12: Experimental results for New Hope parameters and number of bit-flips $\kappa = 19$; for details see Table 3.11.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 19$, odd							$\kappa = 19$, even						
2	5	2	36.7	15.6	52.3	57.8%	2	5	2	38.3	10.3	**48.7**	59.4%
2	6	2	42.1	15.6	57.6	76.6%	2	6	2	44.0	10.3	54.4	69.5%
2	7	2	47.2	15.6	62.8	82.8%	2	7	2	49.5	10.3	59.8	84.4%
2	8	2	52.1	15.6	67.7	90.6%	2	8	2	54.8	10.3	65.1	89.8%
3	6	2	42.1	12.8	54.9	51.6%	3	5	2	38.3	11.1	49.4	58.6%
3	7	2	47.2	12.8	60.0	56.2%	3	6	2	44.0	11.1	55.1	75.0%
3	8	2	52.1	12.8	64.9	62.5%	3	7	2	49.5	11.1	60.6	81.2%
							3	8	2	54.8	11.1	65.9	85.9%
							4	6	2	44.0	11.7	55.7	60.9%
							4	7	2	49.5	11.7	61.2	69.5%
							4	8	2	54.8	11.7	66.4	75.0%
							5	6	2	44.0	12.7	56.7	50.8%
							5	7	2	49.5	12.7	62.2	59.4%
							5	8	2	54.8	12.7	67.5	60.2%

Table 3.13: Experimental results for New Hope parameters and $\kappa = 25$. For details see Table 3.11.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 25$, odd							$\kappa = 25$, even						
2	7	2	47.2	15.6	62.8	52.3%	2	7	2	49.5	10.3	59.8	53.9%
2	7	3	51.4	15.6	66.9	59.4%	2	7	3	52.9	10.3	63.3	54.7%
2	8	2	52.1	15.6	67.7	63.3%	2	8	2	54.8	10.3	65.1	64.1%
2	8	3	56.9	15.6	72.5	75.0%	2	8	3	58.7	10.3	69.0	64.8%
							3	7	2	49.5	11.1	**60.6**	56.2%
							3	7	3	52.9	11.1	64.1	56.2%
3	8	2	52.1	12.8	64.9	50.0%	3	8	2	54.8	11.1	65.9	66.4%
3	8	3	56.9	12.8	69.7	57.8%	3	8	3	58.7	11.1	69.8	67.2%

Table 3.14: Experimental results for New Hope parameters and $\kappa = 30$. For details see Table 3.11.

			cost							cost			
θ	α	β	guess	enum	total	rate	θ	α	β	guess	enum	total	rate
$\kappa = 30$, odd							$\kappa = 30$, even						
2	8	3	56.9	15.6	72.5	52.3%							
2	9	2	56.9	15.6	72.4	56.2%	2	9	2	59.8	10.3	**70.2**	56.2%
2	9	3	62.3	15.6	77.8	66.4%	2	9	3	64.3	10.3	74.6	58.6%
2	10	2	61.5	15.6	77.0	64.1%	2	10	2	64.8	10.3	75.1	68.0%
2	10	3	67.5	15.6	83.0	76.6%	2	10	3	69.7	10.3	80.1	70.3%
							3	9	2	59.8	11.1	71.0	55.5%
3	9	3	62.3	12.8	75.1	50.0%	3	9	3	64.3	11.1	75.4	56.2%
							3	10	2	64.8	11.1	75.9	67.2%
3	10	3	67.5	12.8	80.3	54.7%	3	10	3	69.7	11.1	80.8	68.0%

Table 3.15: Cold boot attacks on New Hope KEM. The column "cost" gives the cost of recovering all 1024 components of the secret in terms of the number of lattice points visited during enumeration ($\approx$ 100 CPU cycles each). The column "rate" shows the overall success rate $1 - (1 - p_0)^2$ for recovering 1024 components of the secret, cf. Section 3.5.4.1. For the columns labelled "non-NTT", see caption of Table 3.10.

bit-flip rates		NTT		non-NTT
ρ_0	ρ_1	cost	rate	cost
0.17%	0.1%	$2^{48.7}$	84%	$2^{53.7}$
0.25%	0.1%	$2^{60.6}$	81%	$2^{60.0}$
0.32%	0.1%	$2^{70.2}$	81%	$2^{66.1}$

3.5.5.3 Conclusions of the Main Attack

We have shown that for practical cold boot bit flip rates, both Kyber and New Hope are vulnerable to relatively efficient cold boot attacks whether the NTT is used or not. In the case of New Hope, key storage using an NTT does not make much of a difference from the perspective of a cold boot attacker for the attacks considered above. However, for the Kyber case, use of the NTT for key storage does seem to reduce resistance to cold boot attacks. One reason for this is because the cold boot problem when considering MLWE keys splits into d independent problems in relatively low dimension. When the NTT is not used for key storage, this independency does not exist. Therefore, a suggested counter-measure to make cold boot attacks harder for MLWE schemes is to avoid storing secret keys in the NTT domain. Note that this does not rule out the possibility of using an NTT altogether, as the NTT may be computed as and when polynomial multiplication is required.

3.6 Low Hamming Weight Secret Block Leakage Attack

In this section, we discuss an extremely efficient attack that allows for the recovery of a low Hamming weight RLWE secret key $s \in R_q$ given knowledge of a block of consecutive entries in $\mathsf{NTT}(s)$. In particular, if the Hamming weight of the secret is w, $2w$ consecutive entries are required for this key recovery attack. Note that this leakage characteristic is difficult to motivate in practice unlike the cold boot leakage considered in the previous section.

3.6.1 Linear Complexity

Linear feedback shift registers (LFSR) for binary sequences are well known as a concept. We will be considering LFSRs over a field $\mathbb{Z}_q$ for prime q i.e. shift registers where the input (or feedback function) is a linear combination (over $\mathbb{Z}_q$) of the current register values. We call the number of registers in an LFSR the length of the LFSR. Although the definition of an LFSR is well-known, we include it below for the sake of completeness. For a simple pictoral representation of an LFSR, see

Figure 3.5. Note that LFSRs have a wide range of applications and have been used in the design of efficient stream-ciphers in the past that are now known to be vulnerable to efficient attacks [21, 20].

Definition 29 (Linear Feedback Shift Register)**.** *Let q be a prime, n be a positive natural number and $F : \mathbb{Z}_q^n \to \mathbb{Z}_q$ be a linear function. Denote the entries of an n-entry register as $(x_{n-1}, \ldots, x_0)$. A linear feedback shift register with respect to feedback function F over $\mathbb{Z}_q$ with length n is an n-entry register that evolves according to the mapping*

$$(x_{n-1}, \ldots, x_0) \longmapsto (F(x_{n-1}, \ldots, x_0), x_{n-1}, \ldots, x_1)) \,.$$

Definition 30 (Linear Complexity)**.** *The linear complexity of a sequence is the length of the shortest LFSR generating the sequence.*

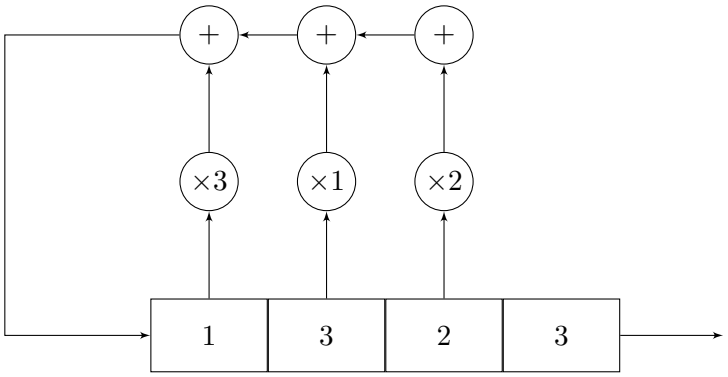
Definition 31 (Connection Polynomial)**.** *Suppose an LFSR produces a sequence $(a_i)_{i=0}^{L'}$ via the relation $a_n + c_1 \cdot a_{n-1} + \ldots c_L \cdot a_{n-L} = 0$ for $L' \geq n > L$ and constants c_i. Then the connection polynomial of this LFSR is defined to be $C(D) := 1 + c_1 D + \ldots c_L D^L$.*

Remark 6. *The linear complexity need not be equal to the degree of the minimal connection polynomial for* finite *sequences (see the example below). However, these two quantities are equal when considering infinite periodic sequences with a finite period.*

Example 5. *Suppose that we are working in the field $\mathbb{Z}_7$ and consider the sequence $(3, 2, 3, 1, 3, 2, 4)$. It can be shown that a LFSR with 4 registers with a connection polynomial of $C(D) = 1 + 4D + 6D^2 + 5D^3$ can be used to generate this sequence. To check this, pick the initial loading of the 4 registers to be $(3, 2, 3, 1)$ and then observe that $3 + 4 + 6 \cdot 3 + 5 \cdot 2 = 0$, $2 + 4 \cdot 3 + 6 + 5 \cdot 3 = 0$ etc. A pictorial representation of this LFSR is given in Figure 3.5.*

The linear complexity and minimal connection polynomial of any finite (or finite period infinite) sequence can be calculated in polynomial time using the Berlekamp-Massey algorithm [99]. This algorithm is generic as it accounts for sequences over *any* field. Clearly, the Berlekamp-Massey can be used as a cryptanalytic tool against any LFSR-based stream-cipher. However, we will be using the Berlekamp-Massey algorithm in a slightly different context here.

Figure 3.5: A minimal length LFSR generating the finite sequence $(3, 2, 3, 1, 3, 2, 4)$ over $\mathbb{Z}_7$. Note that the coefficients of the connection polynomial are the negation of the multiplicands in this diagram. This LFSR has length 4, yet the minimal degree connection polynomial has degree 3.



We now briefly overview the structure of the Berlekamp-Massey algorithm. Suppose we wish to find the linear complexity and connection polynomial of the finite sequence $(a_0, \ldots, a_{n-1})$. Then the Berlekamp-Massey algorithm iteratively calculates the linear complexity and connection polynomial of each subsequence $a_0, \ldots, a_i$ for $i = 0, \ldots, n-1$. Suppose we have just completed the $(k-1)^{th}$ loop and have arrived at a linear complexity of l_{k-1} and connection polynomial $C^{(k-1)}(D) := 1 + c_1^{(k-1)}D + \cdots + c_{l_{k-1}}^{(k-1)}D^{l_{k-1}}$ (recall that some of these coefficients may be 0) for the subsequence $(a_0, \ldots, a_{k-1})$. To start the k^{th} iteration, we calculate the *discrepancy* defined to be $d := a_k + \sum_{i=1}^{l_{k-1}} c_1^{k-i} a_{k-i}$. This tells us how far $C^{k-1}(D)$ is from being the connection polynomial of the subsequence $(a_0, \ldots a_k)$. There are three cases to consider when updating the linear complexity and connection polynomial:

1. If $d = 0$, then the linear complexity and connection polynomial remain the same.

2. If $d \neq 0$, there are two sub-cases:

   (a) If $2 \cdot l_{k-1} > k + 1$, then the connection polynomial must change but the linear complexity stays the same.

   (b) If $2 \cdot l_{k-1} \leq k + 1$ then the connection polynomial changes and the linear complexity increases.

The Berlekamp-Massey algorithm gives explicit formulae for updating linear complexities and connection polynomials depending on which of the three cases is relevant. For a rigorous proof of correctness, see [99]. The pseudocode for the Berlekamp-Massey algorithm is given as Algorithm 3.

Algorithm 3: The Berlekamp-Massey algorithm

Input: $s = (s_0, \ldots, s_{n-1})$
Output: Linear complexity of s and connection polynomial $(L, C(D))$
/* Initialisation */
$C(D) \leftarrow 1$; $B(D) \leftarrow 1$; $x \leftarrow 1$;; $L \leftarrow 0$; $b \leftarrow 1$; $N \leftarrow 0$;
/* Main Loop */
while $N < n$ **do**
  $(c_0, c_1, \ldots, c_{n-1}) \leftarrow C(D)$.Coefficients(); $d \leftarrow s_N + \sum_{i=1}^{L} c_i s_{N-i}$; // the discrepancy
  /* Case 1: no updates required */
  if $d = 0$ **then**
    $x \leftarrow x + 1$; **continue**;
  /* Case 2: update only the connection polynomial */
  else if $d \neq 0$ *and* $2L > N$ **then**
    $C(D) \leftarrow C(D) - db^{-1}D^x B(D)$; $x \leftarrow x + 1$; **continue**;
  /* Case 3: update both linear complexity and connection polynomial */
  else
    $T(D) \leftarrow C(D)$; $C(D) \leftarrow C(D) - db^{-1}D^x B(D)$; $L \leftarrow N + 1 - L$;
    $B(D) \leftarrow T(D)$; $b \leftarrow d$; $x \leftarrow 1$;
  $N \leftarrow N + 1$;
return $(L, C(D))$

An important ingredient of our attack is the following theorem. In the below, $\mathsf{HW}(\boldsymbol{s})$ denotes the Hamming weight (i.e. number of non-zero entries) of $\boldsymbol{s}$ and $\mathsf{LC}((s))$ denotes the linear complexity of the sequence (s).

Theorem 4 (Blahut [26])**.** *Let q be a prime such that there exists an $(2n)^{th}$ primitive root of unity in $\mathbb{Z}_q$ and let $\mathsf{NTT}(\cdot)$ denote a traditional NTT[12] of dimension n over $\mathbb{Z}_q$. For any $\boldsymbol{s} \in \mathbb{Z}_q^n$, define $(\hat{\boldsymbol{s}}) := (\mathsf{NTT}(\boldsymbol{s}), \mathsf{NTT}(\boldsymbol{s}), \ldots)$ to be the sequence comprising of infinitely many copies of $\mathsf{NTT}(\boldsymbol{s})$. Then $\mathsf{LC}((\hat{\boldsymbol{s}})) = \mathsf{HW}(\boldsymbol{s})$.*

Blahut's Theorem has been proven for the traditional NTT. However, in this work

[12] Note that the matrix associated to the traditional NTT has $(i, j)^{th}$ component given by ω^{ij} for n^{th} primitive root of unity ω

we are considering the *negacyclic* NTT. It turns out that the correctness of Blahut's Theorem for the negacyclic NTT follows straight-forwardly from the traditional case:

Lemma 10 (Negacyclic Blahut)**.** *Let q be a prime such that there exists an $(2n)^{th}$ primitive root of unity in $\mathbb{Z}_q$ and let $\mathsf{NTT}(\cdot)$ denote a negacyclic NTT of dimension n over $\mathbb{Z}_q$. For $\boldsymbol{s} \in \mathbb{Z}_q^n$, define $(\hat{\boldsymbol{s}}) := (\mathsf{NTT}(\boldsymbol{s}), \mathsf{NTT}(\boldsymbol{s}), \dots)$ to be the sequence comprising of infinitely many copies of $\mathsf{NTT}(\boldsymbol{s})$. Then, from Blahut's theorem for the traditional NTT, $\mathsf{LC}((\hat{\boldsymbol{s}})) = \mathsf{HW}(\boldsymbol{s})$.*

Proof. In this proof, we denote whether an NTT is negacyclic or traditional using *neg* or *trad* in the subscript. Let $\omega \in \mathbb{Z}_q$ be a primitive n^{th} root of unity and $\gamma \in \mathbb{Z}_q$ be a square root of ω. Also let $\boldsymbol{g} = (1, \gamma, \gamma^2, \dots, \gamma^{n-1})$ and $\odot$ denote the component-wise multiplication of vectors. We then have

$$\mathsf{NTT}_{neg}(\boldsymbol{s})_i = \sum_{j=0}^{n-1} \omega^{ij}(\gamma^j s_j) = \mathsf{NTT}_{trad}(\boldsymbol{g} \odot \boldsymbol{s})_i \tag{3.13}$$

Defining the infinite sequence $(\widehat{\boldsymbol{g} \odot \boldsymbol{s}_{trad}}) := (\mathsf{NTT}_{trad}(\boldsymbol{g} \odot \boldsymbol{s}), \mathsf{NTT}_{trad}(\boldsymbol{g} \odot \boldsymbol{s}), \dots)$, we have that

$$\mathsf{LC}((\hat{s})) = \mathsf{LC}((\widehat{\boldsymbol{g} \odot \boldsymbol{s}_{trad}})) = \mathsf{HW}(\boldsymbol{g} \odot \boldsymbol{s}) = \mathsf{HW}(\boldsymbol{s}) \tag{3.14}$$

where the first equality is due to Equation (3.13), the second is due to Blahut's theorem for the traditional NTT, and the last is due to the fact that $\gamma^j s_j = 0$ if and only if $s_j = 0$. □

Blahut's theorem tells us that a secret with Hamming weight w corresponds precisely to an infinite sequence in the NTT domain with linear complexity w. In order to exploit this relation, we use the Berlekamp-Massey algorithm [99] which provides a method for finding the linear complexity and connection polynomial of any sequence.

To investigate this further, we can produce a linear complexity *profile* for a sequence by plotting the maximal index present in the subsequence in each iteration against the linear complexity calculated for that subsequence. This is a trivial task when considering the previously mentioned structure of the Berlekamp-Massey algorithm. For the linear complexity profile of a random sequence, we typically end up observing that the points on the profile exhibit a step behaviour roughly lying on the line

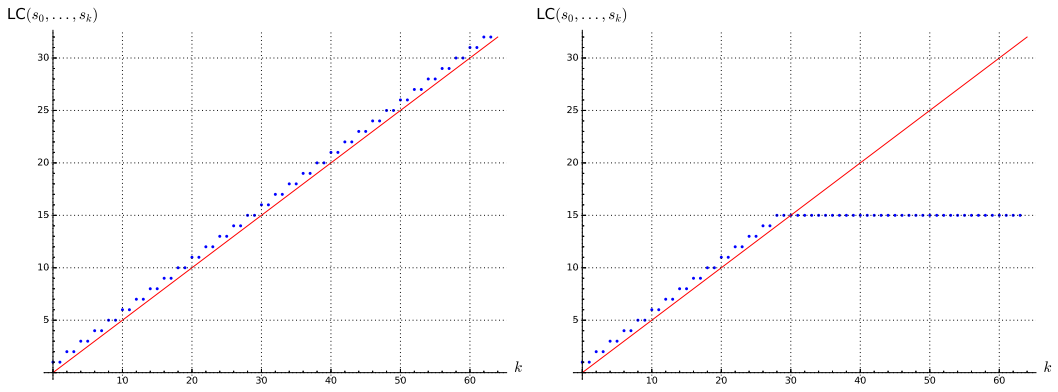


Figure 3.6: Linear complexity profiles for a random sequence (left) and for the NTT of a low Hamming weight vector (right).

$y = x/2$. However, if our sequence is the result of a NTT transform of a low Hamming weight vector, Blahut's theorem tells us that we should get low linear complexity. In this case, the linear complexity profile shows the same step behaviour, but levels off when the low linear complexity of the sequence is reached. Examples of linear complexity profiles in both these cases in given in Figure 3.6.

3.6.2 Full Attack Description

Suppose we are given a noisy version of a secret key with low Hamming weight w in the NTT domain. We will show that the Berlekamp-Massey algorithm implicitly yields a strategy for finding such a key given $2w$ consecutive error-less symbols. The logic behind the attack is that the connection polynomial is recovered fully by the Berlekamp-Massey algorithm once $2w$ symbols have been considered. A consequence of this is that the attack in Algorithm 4 works if there are $2w$ clean symbols in the noisy key. Note that if we were to disregard the NTT, leaking $2w$ symbols of the secret key does not lead to an immediate key recovery attack.

Lemma 11. *For a prime q, integer n and vector $\boldsymbol{s} \in \mathbb{Z}_q^n$ with Hamming weight w, the minimal connection polynomial of $(\hat{\boldsymbol{s}}) := (\mathsf{NTT}(\boldsymbol{s}), \mathsf{NTT}(\boldsymbol{s}), \ldots)$ can be recovered given $2w$ consecutive symbols of $\hat{\boldsymbol{s}}$.*

Proof. Without loss of generality, assume throughout this proof that the given $2w$ consecutive symbols are at the beginning of the sequence. Suppose that our lin-

ear complexity has reached w (which is its maximum value for the error-less NTT sequence) after the consideration of the first $2w$ symbols. We analyse the loop in the Berlekamp-Massey algorithm that considers $2w + 1$ symbols. Since we know that the linear complexity cannot increase, we must either be in case 1 or 2 from Algorithm 3. However, to be in case 2, we must have $2L > N$ which translates to $2w > 2w + 1$ for the loop in consideration. This is clearly impossible, so we must be in the case where the connection polynomial does not change. The same argument holds for the remaining iterations.

To complete the argument, we need to show that the linear complexity after $2w$ iterations is in fact w. Suppose not, i.e. that we have a linear complexity of $w' < w$. Then at some point in the remaining iterations, we must increase the linear complexity to w. Using the notation from Algorithm 3, suppose the first increase occurs when $N = 2w + k$ for some $k \geq 0$. Then we must be in case 3 from Algorithm 3, so we update the linear complexity to $2w + k + 1 - w' > w$ which is a contradiction. Therefore we must reach the linear complexity of w after $2w$ symbols have been considered. □

Note that we can change the starting point of the sequence $(\hat{\boldsymbol{s}})$ without changing the proof of the result above. Therefore in the attack, we do not require that the $2w$ error-less symbols occur in the first components of $\mathsf{NTT}(\boldsymbol{s})$. If we do not know where the error-less symbols are, we can simply re-run the Berlekamp-Massey algorithm on all of the cyclic shifts of $\mathsf{NTT}(\boldsymbol{s})$ in an attempt to get the error-less symbols at the beginning of the sequence.

A general framework for this attack is given as Algorithm 4. Note that this algorithm outputs a list of candidates given a noisy NTT secret. A simple way to find the solution within this list would be to substitute each candidate back into the RLWE instance that is being attacked.

3.6.3 Cold Boot Scenario

We now consider our Blahut-Berlekamp-Massey attack within an NTT cold boot scenario. We will work with RLWE parameters $n, q, w := \mathsf{HW}(s)$. Recall that we need

Algorithm 4: Generic attack based on Berlekamp-Massey algorithm

Input: $\tilde{\hat{s}} = (\tilde{s}_0, \ldots, \tilde{s}_{n-1})$ % noisy NTT of secret, Hamming weight $w < n/2$

Output: List of candidate secrets $\mathcal{L}$

$\mathcal{L} \leftarrow \emptyset$; **for** $i = 0, \ldots, n-1$ **do**
  $(t_0, \ldots, t_{2w-1}) \leftarrow (s_i, \ldots, s_{2w+i})$;
  $(L, C(D)) \leftarrow$Berlekamp-Massey$(t_0, \ldots t_{2w-1})$;
  $(c_0, \ldots, c_w) \leftarrow C(D)$.coefficients(); **for** $j = 2w, \ldots, n-1$ **do**
    $t_j \leftarrow -\sum_{k=1}^{w} c_i t_{j-i}$; % derive the remaining symbols
  for $j = 0, \ldots, n-1$ **do**
    $r_j = t_{j-i \bmod n}$;
  $\mathcal{L}$.Add$((r_0, \ldots, r_{n-1}))$;
return $\mathcal{L}$

$2w$ consecutive clean symbols/NTT entries for the attack to go through which is equivalent to requiring $2w\lceil \log_2 q \rceil$ consecutive *bits* of the secret key. When considering these bits in a noisy version of the secret key, about half of the bits will be out of the ground state. Therefore, assuming a bit-flip rate of ρ_0 towards the ground state and a bit-flip of ρ_1 away from the ground state, we expect $(\rho_0 + \rho_1)w\lceil \log_2 q \rceil$ bit-flips within the entire block of $2w\lceil \log_2 q \rceil$ bits. The strategy is to exhaustively search for the bits that were flipped and run the Berlekamp-Massey attack algorithm to check each guess. Ignoring the trivial cost of running Berlekamp-Massey, we have a rough average complexity of

$$\binom{w\lceil \log_2 q \rceil}{\lfloor \rho_0 w \lceil \log_2 q \rceil \rceil} \cdot \binom{w\lceil \log_2 q \rceil}{\lfloor \rho_1 w \lceil \log_2 q \rceil \rceil}. \tag{3.15}$$

For example parameters $w = 64, q = 12289, n = 1024$, we have an attack with complexity roughly 2^{80} for bit-flip rate $\rho_0 = 1\%, \rho_1 = 0.1\%$ remembering that ρ_1 is the retrograde flip rate (if $\rho_0 = 0.17\%$, the attack complexity is roughly 2^{28} for $w = 64$ and 2^{50} for $w = 128$). In certain scenarios, this complexity could be much lower. For example, suppose there is a block of $2w\lceil \log_2 q \rceil$ consecutive bits where the majority flips could have only occurred away from the ground state. Then we expect only a small number of bit-flips in this block since $\rho_1 < \rho_0$, which reduces the amount of guesses required before the attack is successful. Therefore, in a cold boot attack, we would be able to identify the optimal consecutive block of $2w$ symbols to launch our attack on very easily.

3.6.4 Future Directions for Linear Complexity Attacks

The k-error linear complexity of a sequence is the minimal linear complexity attainable when changing at most k symbols. This notion corresponds closely to the case where we have a noisy version of the key that contains at most k erroneous symbols. If we had an algorithm that computed k-error linear complexity along with the symbol changes required to minimise the linear complexity, then we would be able to recover secret keys in many non-trivial cases.

However, efficient algorithms for calculating k-error linear complexities only exist for specific classes of sequences [132, 74]. There is currently no efficient algorithm that handles sequences with power-of-two period n over a field $GF(q)$ satisfying $2n|(q-1)$. It is an interesting open problem to discover such an algorithm.

3.7 Periodic Leakage Attack [45]

In this section, we present another NTT-based key recovery attack from the literature [45] enabled by a certain leakage characteristic. Let $n'|n$. Using indexing in $\mathbb{Z}_{2n}^*$ (see Section 3.7.1), the leakage considered in this attack is periodic on all entries of $\mathsf{NTT}(s)$ with index j satisfying $j \bmod 2n' \in \mathcal{S}$ for some $\mathcal{S} \subset \mathbb{Z}_{2n'}^*$. As we will see, the attack performance varies according to the size of n and $\mathcal{S}$. The attack consists of two main parts:

1. (Section 3.7.2) Deriving a large system of noiseless equations.
2. (Section 3.7.3) Using lattice reduction in small dimensions to derive the most likely solutions for the key.

3.7.1 Indexing in $\mathbb{Z}_{2n}^*$

If we fix n to be a power of 2, a very convenient convention when presenting this attack is to index the NTT by integers in $\mathbb{Z}_{2n}^*$ rather than in $\mathbb{Z}_n$. This means that the index of an entry of $\mathsf{NTT}(\boldsymbol{s})$ corresponds exactly to the power of the $(2n)^{th}$ primitive

root of unity γ used to derive that entry. For example, consider $s \in R_q$. Then using this indexing convention, we have

$$\mathsf{NTT}(s) = (\hat{s}_1, \hat{s}_3, \ldots, \hat{s}_{2n-1}) := (s(\gamma^1), s(\gamma^3), \ldots, s(\gamma^{2n-1})).$$

Furthermore, suppose $n'|n$. Then n' is also a power of 2 and $\mathbb{Z}_{2n'}^* = \mathbb{Z}_{2n'} \cap \mathbb{Z}_{2n}^*$.

3.7.2 Deriving the Noiseless Systems of Equations

Fix some $n'|n$ and $\mathcal{S} = \{\alpha \in \mathbb{Z}_{2n'}^*\}$. Assume that the values of $\hat{s}_j$ are known for all j such that $j \bmod 2n' = \alpha$. In the NTT domain, a RLWE sample is of the form $(\hat{\boldsymbol{a}}, \hat{\boldsymbol{a}} \odot \hat{\boldsymbol{s}} + \hat{\boldsymbol{e}})$ where $\odot$ represents component-wise multiplication of vectors. Since q is prime, leakage of $\hat{s}_j$ implies leakage of $\hat{e}_j$. Noting that $\gamma^{2n'}$ is an $(n/n')^{th}$ primitive root of unity, we have the isomorphism

$$\frac{\mathbb{Z}_q[X]}{X^{n/n'} - (\gamma^\alpha)^{n/n'}} \equiv \frac{\mathbb{Z}_q[X]}{X - \gamma^\alpha} \times \frac{\mathbb{Z}_q[X]}{X - \gamma^{\alpha+2n'}} \times \frac{\mathbb{Z}_q[X]}{X - \gamma^{\alpha+2\cdot 2n'}} \times \cdots \times \frac{\mathbb{Z}_q[X]}{X - \gamma^{\alpha+(\frac{n}{n'}-1)2n'}}$$

by the chinese remainder theorem. This means that given the coordinates $\hat{e}_j$ for all $j = \alpha \bmod 2n'$, or equivalently $e(x) \bmod \left(x - \gamma^{\alpha+2n'k}\right)$ for $k = 0, \ldots, \frac{n}{n'} - 1$, we can explicitly calculate $e^\alpha(x) := e(x) \bmod \left(x^{n/n'} - (\gamma^\alpha)^{n/n'}\right) = \sum_{i=0}^{n/n'-1} e_i^\alpha x^i$ via the CRT isomorphism. Since we are modding out the polynomial $\left(x^{n/n'} - (\gamma^\alpha)^{n/n'}\right)$, considering one coefficient of $e^\alpha(x)$ at a time, we end up with the under-determined (but noiseless) systems of equations

$$\sum_{k=0}^{n'-1} \gamma^{\alpha k n/n'} e_{i+kn/n'} = e_i^\alpha. \tag{3.16}$$

In the above system, the e_i^α are known whereas the e_i are unknowns. Note that each value of $i \in \{0, \ldots, \frac{n}{n'} - 1\}$ gives rise to an independent system in the n' unknowns $e_i, \ldots, e_{i+\frac{n}{n'}\cdot(n'-1)}$. At a high level, the attack will proceed by finding candidates to these under-determined systems conditioned on the fact that the e_i are small integers as they are drawn from an error distribution.

For now, assume that the shortest possible solution to the systems in (3.16) is used as the candidate solution. Although this is a reasonable criteria for picking candidate solutions, there is a high chance that we will not be able to recover e (and therefore s) using this simple strategy. In particular, each of the candidate

solutions to the independent systems that we pick must be the correct one if we want to recover the correct value for s. Unfortunately, this is not very likely due to how under-determined the systems are. To get over this issue, multiple RLWE samples are required to amplify the success probability.

Making use of multiple RLWE samples Obtaining a correct solution to a *single* system of the form (3.16) yields the correct values of $e_{i+kn/n'}$ i.e. n' coefficients of an error vector. Viewing a RLWE sample as $(\mathbf{A}, \boldsymbol{b} = \mathbf{A} \cdot \boldsymbol{s} + \boldsymbol{e})$ where $\mathbf{A}$ is a matrix with negacyclic structure, suppose we look at the entries of $\boldsymbol{b}$ with indices of the form $i + kn/n'$. Since we know the value of the error term at these positions, we derive a system of n' equations in the n unknown coefficients of s. Bearing this in mind, once we have solved n/n' distinct systems of the form (3.16) we end up with a total of $n' \cdot n/n' = n$ equations in the coefficients of s. At this point, recovering s is trivial linear algebra. Recall that we cannot always effectively find the correct solution to under-determined systems of the form (3.16). To remedy this problem, the success probability of the attack can be amplified by allowing the attacker access to extra RLWE samples (i.e. extra systems of the form (3.16)) so that the attacker has the freedom to use only the systems that can be solved correctly with high confidence. How the attacker chooses the appropriate system is addressed in the following section.

3.7.3 Solving for the Most Likely Solution

When presented with the under-determined noiseless systems of the form (3.16), the attacker must make an educated guess on what the correct solution is. This process is aided by the fact that the error polynomial has small coefficients from a *known* error distribution χ. Therefore, once the attacker finds a candidate solution, the probability of seeing that solution when sampling from distribution χ can be used as a way of measuring the confidence of the candidate solution. For the usual (spherical) Gaussian error distribution, it is always the case that the shortest solution will be the one the attacker has the most confidence in. The same can be said for binomial error distributions.

Therefore, the attacker strategy will be to solve the individual systems of the form (3.16) for the *shortest solution*. In addition to finding the shortest solution, the attacker records the confidence of the candidate solution by calculating the probability of the shortest solution under χ divided by the sum of the probabilities of all valid solutions to that system under χ. If the confidence level recorded is higher than a threshold, the candidate solution is kept. Otherwise the candidate solution is removed. Once n/n' candidate solutions with an acceptable confidence level have been recorded, the attacker stops and solves a linear system to recover the secret s.

Since the individual systems will be in dimension n' where n' is a power of two dividing n, finding the shortest solution can be done by running BKZ reduction and lattice enumeration as in Section 3.5.4.1. In particular, the CVP problem to be solved would be over the lattices

$$\Lambda = \{\boldsymbol{v} \in \mathbb{Z}^{n'} : \sum_{k=0}^{n'-1} \gamma^{\alpha k n/n'} v_k = 0 \bmod q\}$$

using the target point $\boldsymbol{t} = (e_i^{\alpha}, 0, \ldots, 0)$. The offset of the closest vector and this target point would yield a candidate solution.

Similar to our cold boot attack, the form of the individual systems is fixed regardless of the RLWE sample. This means that we can perform BKZ reduction once and for all, while simply running lattice enumeration when running the attack. However, at small dimensions e.g. $n' = 8$, a meet-in-the-middle exhaustive search is reasonably efficient and this method used by [45].

Remark 7. *It is easy to extend the above attack to the case where $\hat{s}_j$ are known for $j \bmod 2n' = \alpha_1$ and $j \bmod 2n' = \alpha_2$ i.e. $\mathcal{S} = \{\alpha_1, \alpha_2\}$. The difference is that the single equation given as* (3.16) *becomes two equations in the unknowns $e_{i+kn/n'}$.*

3.7.4 Experimental Evaluation and Findings

In this section, we briefly present the experimental findings of [45]. Exact complexities were omitted in [45] since the attack is quite cheap for the leakage settings considered. However, the number of RLWE samples required gives an idea of the feasibility of these attacks when an attacker has a limited number of RLWE samples

to work with. Mainly, leakage patterns revealing 1/4 of the secret coefficients were considered. For the rest of this summary, we fix $n = 1024$ and binomial error distribution $\mathcal{B}_{16}$ used in New Hope. Consider the case $n' = 4, \mathcal{S} = \{1\}$ or equivalently, $n' = 8, \mathcal{S} = \{1, 9\}$ using the notation from the remark above. Using a confidence threshold of 0.98 for candidate solutions to individual systems, it was observed that 22 RLWE samples yields an attack that succeeds with probability 0.5%. Alternatively, the leakage pattern $n' = 8, \mathcal{S} = \{1, 15\}$ with a confidence threshold of 0.95 yields an attack requiring 170 samples with probability of success 0.1408%. The final experimental result we mention is for $n' = 8, \mathcal{S} = \{1, 7\}$, which yields an attack with success probability 7.5% using 1929 RLWE samples and a confidence threshold of 98%. As we can see, the attack requires many RLWE samples, even for low success probabilities.

We now overview what was found for the case $n' = 8$, $\alpha = 1$ i.e. a leakage of 1/8 NTT coordinates. The first thing to note is that the confidence levels observed when solving the systems of the form (3.16) for cases $n' = 8$ and higher appear to be at most $0.12 \approx 0.125 = 2^{-3}$ for the binomial distribution $\mathcal{B}_{16}$. Therefore, for $n' = 8$, 12% is the maximal reasonable threshold. Using 12.5% as a threshold for acceptable guesses, one would then get a $2^{-3n/n'}$ success probability. Therefore, even for $n = 256$, the highest probability of success expected is 2^{-96} and for the New Hope dimension $n = 1024$, the highest probability of success is around 2^{-384}. Furthermore, these success probabilities assume a very large number of RLWE samples so that the 12.5% threshold can be met for n/n' of the noiseless systems.

The experimental findings presented above along with the required number of RLWE samples suggest that this attack is mainly of theoretical interest. It nonetheless uses the structure of the NTT to launch a novel and interesting attack.

Chapter 4

Reductions between MLWE and RLWE

Contents

4.1 Chapter Synopsis

In this chapter, we present reductions between different algebraic LWE variants. The main results from this section are heavily inspired by a well-known reduction from plain LWE with modulus q, dimension n to modulus $q^{n/n'}$, dimension n' for any $n'|n$ [32]. The error rate in this well-known reduction increases by roughly a factor of $\sqrt{n}$. We stress that all reductions mentioned in this section ensure that standard Gaussian error distributions are preserved (but do incur an increase in the Gaussian error width). In addition, the reductions require small secrets to work. In this chapter, we generalise the techniques from [32] to the ring/module

setting. In particular, we show that for module ranks d, d' where $d'|d$ and ring R, there is a reduction from the MLWE problem with modulus q, rank d to the MLWE problem with modulus $q^{d/d'}$, rank d'. Importantly, the error distribution remains statistically indistinguishable from an ellipsoidal Gaussian, but the bound on the error width grows by an amount dependent on properties of the underlying ring R. For power of two cyclotomic R of ring dimension n, the error distribution is shown to grow by a multiplicative factor of roughly $n^{1/2+c}\sqrt{d}$ for any constant $c > 0$ when the normal form secret distribution is used (see Corollary 3 and the following discussion). Note that this is an improvement on the $n^2\sqrt{d}$ factor reported in our original publication [6]. Also, taking $d' = 1$, we get a reduction from MLWE to RLWE.

Still inspired by the work of [32] we derive a reduction from RLWE for power-of-two cyclotomic R with ring dimension n and modulus q to RLWE with power-of-two cyclotomic R with ring dimension $n/2$ and modulus q^2. The error rate in this case increases by a factor of $n^{3/4+c}$ when using the normal form secret distribution (see Corollary 6 and the following discussion). Once again, this improves on our original publication which reported a growth factor of $n^{9/4}$. Unfortunately, the analysis of this reduction uses the Renyi divergence and therefore only works between search variants of RLWE. We end this chapter by briefly overviewing related works in the area of reductions between algebraic variants of LWE. In addition, we highlight an improvement to our RLWE in (n, q) to RLWE in $(n/2, q^2)$ reduction using a recent theorem of Peikert and Pepin [109]. This improvement allows us to both obtain a reduction between *decision* variants, and also reduce the growth rate in error to just a factor of about $n^{1/2+c}$ for any constant $c > 0$. In addition, this improvement allows us to reduce RLWE in (n, q) to RLWE in $(n', q^{n/n'})$ for any power-of-two $n' < n$ at the cost of a factor $\frac{n^{3/2}}{(n')^{1-c}}$ growth in the error rate.

Road map The main technical effort of this chapter is contained in Section 4.3 where a reduction from MLWE to MLWE for general rings is described. The general result is given in Corollary 3. We then describe the consequences of this general reduction in the case that the underlying ring is a power-of-two cyclotomic in Section 4.3.4. Following this, in Section 4.4 we use a similar reduction between different parameter settings of the power-of-two cyclotomic RLWE. Finally, in Section 4.5 we

briefly overview related work and present an improvement/generalisation of the results in Section 4.4.

4.2 Chapter Preliminaries

We will require a result stating that a discrete Gaussian premultiplied by a matrix $\mathbf{S}$ representing multiplication by a field element in the space H, can be drowned by a wide enough continuous Gaussian (Lemma 12). To simplify presentation, we include the proof of this result here. Before proving this lemma, we need the following claim.

Claim 2. *For any $\boldsymbol{\tau} \in \mathbb{R}^n, r \in \mathbb{R}$, define $t_i = \sqrt{\tau_i^2 + r^2}$ for $i = 1, \ldots, n$ and let Λ be an n-dimensional lattice. Let $X \sim D_{\Lambda+\boldsymbol{u},r}$ and $Y \sim D_{\boldsymbol{\tau}}$ and define the random variable $Z := X + Y$. Provided that $\tau_i r/t_i \geq \eta_\epsilon(\Lambda)$ for all i, the distribution of Z is within statistical distance 2ϵ of $D_{\boldsymbol{t}}$.*

Proof. Throughout the proof, we denote the density function of Z at any point $\boldsymbol{z} \in \mathbb{R}^n$ as $p(\boldsymbol{z})$. Let $c_1 = \int_{\mathbb{R}^n} \rho_{\boldsymbol{\tau}}(\boldsymbol{x})d\boldsymbol{x}$, let $\mathbf{T}$ be the diagonal matrix with $T_{i,i} = r\tau_i/t_i$ and let $\boldsymbol{u_z}$ be the vector whose i^{th} entry is $\frac{r^2}{\tau_i^2+r^2}z_i$. We have that

$$\begin{aligned} p(\boldsymbol{z}) &= \sum_{\boldsymbol{x}\in\Lambda+\boldsymbol{u}} \Pr[X = \boldsymbol{x}] \cdot \frac{\rho_{\boldsymbol{\tau}}(\boldsymbol{z}-\boldsymbol{x})}{c_1} \\ &= \frac{1}{c_1 \cdot \rho_r(\Lambda+\boldsymbol{u})} \cdot \left(\sum_{\boldsymbol{x}\in\Lambda+\boldsymbol{u}} \frac{\rho_r(\boldsymbol{x}) \cdot \rho_{\boldsymbol{\tau}}(\boldsymbol{z}-\boldsymbol{x})}{\rho_{\boldsymbol{t}}(\boldsymbol{z})} \right) \cdot \rho_{\boldsymbol{t}}(\boldsymbol{z}) \\ &= \underbrace{\frac{1}{c_1 \cdot \rho_r(\Lambda+\boldsymbol{u})}}_{(\clubsuit_1)} \cdot \underbrace{\left(\sum_{\boldsymbol{x}\in\Lambda+\boldsymbol{u}} \rho_{\mathbf{T}}(\boldsymbol{x}+\boldsymbol{u_z}) \right)}_{(\clubsuit_2)} \cdot \rho_{\boldsymbol{t}}(\boldsymbol{z}). \end{aligned}$$

The term labelled ($\clubsuit_1$) is a constant with respect to $\boldsymbol{z}$. Informally speaking, we will show that ($\clubsuit_2$) is *almost* constant with respect to $\boldsymbol{z}$. This will imply that the mass function of Z is *almost* proportional to $\rho_{\boldsymbol{t}}$ as required. Let $\boldsymbol{v_z} := \boldsymbol{u} + \boldsymbol{u_z}$ and define $f(\boldsymbol{x}) := \rho(\mathbf{T}^{-1}(\boldsymbol{x} + \boldsymbol{v_z}))$. Note that the Fourier transform of f is given by $\hat{f}(\boldsymbol{y}) = e^{2\pi i\langle \boldsymbol{y}, \boldsymbol{v_z}\rangle} \cdot \rho(\mathbf{T}\boldsymbol{y})/\det(\mathbf{T}^{-1})$. Using this fact and the Poisson summation

formula $\sum_{\boldsymbol{x}\in\Lambda} f(\boldsymbol{x}) = \det(\Lambda^*)\cdot\sum_{\boldsymbol{y}\in\Lambda^*}\hat{f}(\boldsymbol{y})$, we get

$$\begin{aligned}
\left|(\clubsuit_2) - \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\right| &= \left|\sum_{\boldsymbol{x}\in\Lambda}\rho_{\mathbf{T}}(\boldsymbol{x}+\boldsymbol{v_z}) - \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\right| \\
&= \left|\sum_{\boldsymbol{x}\in\Lambda}\rho(\mathbf{T}^{-1}(\boldsymbol{x}+\boldsymbol{v_z})) - \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\right| \\
&= \left|\frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\sum_{\boldsymbol{y}\in\Lambda^*} e^{2\pi i\langle\boldsymbol{y},\boldsymbol{v_z}\rangle}\cdot\rho(\mathbf{T}\boldsymbol{y}) - \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\right| \\
&= \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\cdot\left|\sum_{\boldsymbol{y}\in\Lambda^*\setminus\{\mathbf{0}\}} e^{2\pi i\langle\boldsymbol{y},\boldsymbol{v_z}\rangle}\cdot\rho(\mathbf{T}\boldsymbol{y})\right| \\
&\le \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\cdot\left|\sum_{\boldsymbol{y}\in\Lambda^*\setminus\{\mathbf{0}\}}\rho(\mathbf{T}\boldsymbol{y})\right| \\
&\le \frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}\cdot\left|\sum_{\boldsymbol{y}\in\Lambda^*\setminus\{\mathbf{0}\}}\rho\left(\min_i\frac{\tau_i r}{t_i}\boldsymbol{y}\right)\right| \\
&\le \epsilon\cdot\frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}
\end{aligned}$$

Therefore, we can conclude that $(\clubsuit_2)\in[1-\epsilon,1+\epsilon]\cdot\frac{\det(\Lambda^*)}{\det(\mathbf{T}^{-1})}$ and

$$p(\boldsymbol{z})\in[1-\epsilon,1+\epsilon]\cdot C\cdot\rho_{\boldsymbol{t}}(\boldsymbol{z}) \tag{4.1}$$

for some constant C. Integrating over all $\boldsymbol{z}$, we get that $C\in[\frac{1}{1+\epsilon},\frac{1}{1-\epsilon}]\cdot\frac{1}{c_1}$ implying that

$$p(\boldsymbol{z})\in\left[\frac{1-\epsilon}{1+\epsilon},\frac{1+\epsilon}{1-\epsilon}\right]\cdot\frac{\rho_{\boldsymbol{t}}(\boldsymbol{z})}{c_1}. \tag{4.2}$$

Using Claim 1 and the fact that $(1-\epsilon)/(1+\epsilon)\ge 1-2\epsilon$ completes the proof. □

Lemma 12. *Let R be the ring of integers of a field K with degree n and take arbitrary positive $r, B\in\mathbb{R}$. For any non-zero $s\in K$, let $\boldsymbol{S}\in\mathbb{Q}^{n\times n}$ be the matrix corresponding to field multiplication by s in the space H and let $\sigma_i=\sigma_i(s)$ for $i=1,\ldots,n$. Let $\boldsymbol{S'}$ be the diagonal matrix with $(i,i)^{th}$ entry $\sqrt{B^2+|\sigma_i|^2}$. For any n-dimensional lattice Λ and $\boldsymbol{u}\in\mathbb{R}^n$, the random variable $Z=\boldsymbol{S}\cdot X+Y$ where $X\sim D_{\Lambda+\boldsymbol{u},r}$ and $Y\sim D_{rB\mathbf{I}_n}$ is within statistical distance 2ϵ of $D_{r\boldsymbol{S'}}$ where provided that $\frac{rB}{\sqrt{B^2+|\sigma_i|^2}}\ge\eta_\epsilon(\Lambda)$ for all i.*

Proof. Let $W=\mathbf{S}^{-1}Z=X+\mathbf{S}^{-1}Y$. The distribution of $\mathbf{S}^{-1}Y$ is a continuous Gaussian with covariance matrix $(rB)^2(\mathbf{S}^T\cdot\mathbf{S})^{-1}$ which is diagonal with i^{th}

entry $(rB)^2/|\sigma_i|^2$. Therefore, we express the distribution of $\mathbf{S}^{-1}Y$ as $D_{\boldsymbol{\tau}}$ where $\tau_i = rB/|\sigma_i|$. We can now apply Claim 2 to the random variable W whenever $\frac{r^2B}{|\sigma_i|\sqrt{(rB/|\sigma_i|)^2+r^2}} = \frac{rB}{\sqrt{B^2+|\sigma_i|^2}} \geq \eta_\epsilon(\Lambda)$. This allows us to conclude that the distribution of W is within statistical distance 2ϵ of $D_{\boldsymbol{t}}$ where $t_i = \sqrt{\tau_i^2 + r^2}$. In other words, the distribution of W is continuous Gaussian with diagonal covariance matrix $\mathbf{T}$ where $T_{i,i} = t_i^2$.

To complete the proof, we consider the distribution of $Z = \mathbf{S}W$. By using the data processing inequality for statistical distance, Z is at most a statistical distance 2ϵ away from a continuous Gaussian with covariance matrix $\mathbf{STS}^T$. Recall from the discussion in Section 2.5.2 that $\mathbf{S} = \mathbf{U}_H^\dagger \mathbf{D}\mathbf{U}_H$ for some diagonal matrix $\mathbf{D}$ and unitary $\mathbf{U}_H$ given in Equation (2.5). Let r_1 be the number of real field embeddings and r_2 the number of pairs of complex embeddings. This means that for $i = r_1 + 1, \ldots, r_1 + r_2$, $|\sigma_i| = |\sigma_{i+r_2}|$ and therefore that $t_i = t_{i+r_2}$. From these observations, we can see that $\mathbf{T}$ and $\mathbf{S}$ commute as they share a basis of eigenvectors given by the columns of

$$\mathbf{U}_H^\dagger = \begin{bmatrix} \mathbf{I}_{r_1} & 0 & 0 \\ 0 & \frac{1}{\sqrt{2}}\mathbf{I}_{r_2} & \frac{1}{\sqrt{2}}\mathbf{I}_{r_2} \\ 0 & \frac{-i}{\sqrt{2}}\mathbf{I}_{r_2} & \frac{i}{\sqrt{2}}\mathbf{I}_{r_2} \end{bmatrix} \in \mathbb{C}^{n\times n}. \tag{4.3}$$

Therefore, we can write $\mathbf{STS}^T = \mathbf{T} \cdot \mathbf{SS}^T$. Since $\mathbf{T}$ and $\mathbf{SS}^T$ are both diagonal with i^{th} entry $|\sigma_i|^2$ and t_i^2 respectively, the covariance matrix associated with Z is diagonal with i^{th} entry $|\sigma_i|^2 t_i^2 = r^2(B^2 + |\sigma_i|^2)$. □

4.3 Reductions Between MLWE Problems

In this section, we show how to reduce an MLWE instance in module rank d and modulus q to an MLWE instance in rank d' and modulus q'. The particular case where $d' = 1$ yields a reduction from MLWE to RLWE .

4.3.1 Intuition

We start by describing the high-level intuition behind the reduction for the case $d' = 1$ and where the modulus goes from q to q^d. For the intuition, we ignore the dual ring $R^\vee$, replacing it with R. In this case, our strategy is to map $(\boldsymbol{a}, \boldsymbol{s}) \in (R_q)^d \times (R_q)^d$ to $(\tilde{a}, \tilde{s}) \in R_{q'} \times R_{q'}$ aiming to satisfy the approximate equation

$$\frac{1}{q} \langle \boldsymbol{a}, \boldsymbol{s} \rangle \approx \frac{1}{q^d} (\tilde{a} \cdot \tilde{s}) \bmod R. \tag{4.4}$$

We then map from b to $\tilde{b} \approx b \bmod R$. For $q = \Omega(\mathrm{poly}(n))$, if we take $\tilde{s} = (q^{d-1}, \ldots, 1)^T \cdot \boldsymbol{s}$ and $\tilde{a} = (1, \ldots, q^{d-1})^T \cdot \boldsymbol{a}$, we obtain

$$\begin{aligned} \frac{1}{q^d}(\tilde{a} \cdot \tilde{s}) &= \frac{1}{q} \langle \boldsymbol{a}, \boldsymbol{s} \rangle + \frac{1}{q^2} \left(\sum_{i=1}^{d} a_i \cdot s_{(i+1 \bmod d)} \right) + \frac{1}{q^3}(\ldots) + \ldots \bmod R \\ &\approx \frac{1}{q} \langle \boldsymbol{a}, \boldsymbol{s} \rangle \bmod R. \end{aligned} \tag{4.5}$$

This mapping satisfies the requirement $\tilde{b} \approx (\tilde{a} \cdot \tilde{s})/q^d$ but leads to a narrow, yet non-standard error distribution. The reduction in Theorem 5 is a generalisation of the above idea. Specifically, take some $\mathbf{G} \in (R)^{d' \times d}$ and $\tilde{\boldsymbol{s}} = \mathbf{G} \cdot \boldsymbol{s} \bmod (q'R)^{d'}$. Then we simply require that

$$\frac{1}{q'} \sum_{i=1}^{d'} \sum_{j=1}^{d} \tilde{a}_i g_{ij} s_j \approx \frac{1}{q} \sum_{j=1}^{d} a_j s_j \bmod R. \tag{4.6}$$

This requirement can be satisfied if we choose $\tilde{\boldsymbol{a}}$ such that

$$\frac{1}{q'} \sum_{i=1}^{d'} \tilde{a}_i g_{ij} \approx \frac{1}{q} a_j \bmod R \tag{4.7}$$

for $j = 1, \ldots, d$. To carry out this strategy, we will sample $\tilde{\boldsymbol{a}}$ over an appropriate lattice defined by $\mathbf{G}$ in the canonical embedding. The main challenge in applying this strategy is that we want the error in the new MLWE sample to follow a standard error distribution, i.e. a *continuous* Gaussian. In order to carry out the analysis, we will make use of Lemma 12.

4.3.2 The Main Theorem

Theorem 5. *Let R be the ring of integers of some algebraic number field K of degree n, let d, d', q, q' be integers, $\epsilon \in (0, 1/2)$, and $\boldsymbol{G} \in R^{d' \times d}$. Also, fix $\boldsymbol{s} = (s_1, \ldots, s_d) \in$*

$(R^\vee)^d$. *Further, let* $\boldsymbol{B}_\Lambda$ *be some known basis of the lattice* $\Lambda = \frac{1}{q'}\boldsymbol{G}_H^T R^{d'} + R^d$ *(in* H^d*),* $\boldsymbol{B}_R$ *be some known basis of* R *in* H *and*

$$r \geq \max\left(\|\tilde{\boldsymbol{B}}_\Lambda\|, \frac{1}{q}\|\tilde{\boldsymbol{B}}_R\|\right) \cdot \sqrt{2\ln(2nd(1+1/\epsilon))/\pi}.$$

There exists an efficient probabilistic mapping $\mathcal{F} : (R_q)^d \times \mathbb{T}_{R^\vee} \longrightarrow (R_{q'})^{d'} \times \mathbb{T}_{R^\vee}$ *such that:*

1. *The output distribution given uniform input* $\mathcal{F}(U((R_q)^d \times \mathbb{T}_{R^\vee}))$ *is within statistical distance* 4ϵ *of the uniform distribution over* $(R_{q'})^{d'} \times \mathbb{T}_{R^\vee}$.
2. *Let* $M = R^d$, $M' = R^{d'}$ *and define* $B := \max_{i,j} |\sigma_i(s_j)|$. *The distribution of* $\mathcal{F}(A^{(M)}_{q,\boldsymbol{s},D_\alpha})$ *is within statistical distance* $(2d+6)\epsilon$ *of* $A^{(M')}_{q',\boldsymbol{G}\boldsymbol{s},D_{\boldsymbol{\alpha}'}}$ *where* $(\boldsymbol{\alpha}')_i^2 = \alpha^2 + r^2(\beta^2 + \sum_{j=1}^d |\sigma_i(s_j)|^2)$ *for any* β *satisfying* $\beta^2 \geq B^2 d$.

Proof. We use the canonical embedding on each component of R^d individually, e.g. $\boldsymbol{a}_H = (\sigma_H(a_1), \ldots, \sigma_H(a_d)) \in H^d \simeq \mathbb{R}^{nd}$ and similarly for other module elements. We will also refer to the canonical embedding of R as simply R to ease notation. Suppose we are given $(\boldsymbol{a}, b) \in (R_q)^d \times \mathbb{T}_{R^\vee}$. The mapping $\mathcal{F}$ is performed as follows:

1. Sample $\boldsymbol{f} \leftarrow D_{\Lambda - \frac{1}{q}\boldsymbol{a}_H, r}$. Note that the parameter r is large enough so that we can sample the discrete Gaussian efficiently by Lemma 7.
2. Let $\boldsymbol{v} = \frac{1}{q}\boldsymbol{a}_H + \boldsymbol{f} \bmod R^d \in \Lambda / R^d$ and set $\boldsymbol{x} \in (R_{q'})^{d'}$ to be a random solution of $\frac{1}{q'}\mathbf{G}_H^T \boldsymbol{x} = \boldsymbol{v} \bmod R^d$. Then set $\tilde{\boldsymbol{a}} \in M'$ to be the unique element of M' such that $\tilde{\boldsymbol{a}}_H = \boldsymbol{x}$.
3. For some $\beta > B\sqrt{d}$ sample $\tilde{e}$ from the distribution $D_{r\beta}$ over $K_\mathbb{R} \simeq H$ and set $\tilde{b} = b + \tilde{e}$.
4. Finally, output $(\tilde{\boldsymbol{a}}, \tilde{b}) \in (R_{q'})^{d'} \times \mathbb{T}_{R^\vee}$.

Distribution of $\tilde{\boldsymbol{a}}$. Suppose that $\boldsymbol{a} \in (R_q)^d$ was drawn uniformly at random. Step 2 of the reduction can be performed by adding a random element of the basis of solutions to $\frac{1}{q'}\mathbf{G}_H^T \boldsymbol{y} = 0 \bmod R^d$ to a particular solution of $\frac{1}{q'}\mathbf{G}_H^T \boldsymbol{x} = \boldsymbol{v} \bmod R^d$. In order to show that $\tilde{\boldsymbol{a}}$ is *nearly* uniform random, we will show that the vector

$\boldsymbol{x}$ is *nearly* uniform random over the set $(R_{q'})^{d'}$. Note that every $\boldsymbol{x} \in (R_{q'})^{d'}$ is a solution to $\frac{1}{q'}\mathbf{G}_H^T \boldsymbol{x} = \boldsymbol{v} \bmod R^d$ for some $\boldsymbol{v}$ and the number of solutions to this equation in $(R_{q'})^{d'}$ for each $\boldsymbol{v}$ is the same. Thus, proving that $\boldsymbol{v}$ is *almost* uniform suffices. Observe that $r \geq \eta_\epsilon(\Lambda)$. Therefore, Lemma 6 tells us that for any particular $\bar{\boldsymbol{a}} \in (R_q)^d$ and $\bar{\boldsymbol{f}} \in \Lambda - \frac{1}{q}\bar{\boldsymbol{a}}_H$, we have

$$\begin{aligned}
\Pr[\boldsymbol{a} = \bar{\boldsymbol{a}} \wedge \boldsymbol{f} = \bar{\boldsymbol{f}}] &= q^{-nd} \cdot \frac{\rho_r(\bar{\boldsymbol{f}})}{\rho_r\left(\Lambda - \frac{1}{q}\bar{\boldsymbol{a}}_H\right)} \\
&= \frac{q^{-nd}}{\rho_r(\Lambda)} \cdot \frac{\rho_r(\Lambda)}{\rho_r\left(\Lambda - \frac{1}{q}\bar{\boldsymbol{a}}_H\right)} \cdot \rho_r(\bar{\boldsymbol{f}}) \\
&\in C \cdot \left[1, \frac{1+\epsilon}{1-\epsilon}\right] \cdot \rho_r(\bar{\boldsymbol{f}})
\end{aligned} \tag{4.8}$$

where $C := q^{-nd}/\rho_r(\Lambda)$ is a constant. By summing this equation over appropriate values of $\bar{\boldsymbol{a}}$ and $\bar{\boldsymbol{f}}$, Lemma 6 tells us that for any coset $\bar{\boldsymbol{v}} \in \Lambda/R^d$,

$$\begin{aligned}
\Pr[\boldsymbol{v} = \bar{\boldsymbol{v}}] &\in C \cdot \left[1, \frac{1+\epsilon}{1-\epsilon}\right] \cdot \rho_r(q^{-1}R^d + \bar{\boldsymbol{v}}) \\
&\in C \cdot \rho_r(q^{-1}R^d) \cdot \left[1, \frac{1+\epsilon}{1-\epsilon}\right] \cdot \frac{\rho_r(q^{-1}R^d + \bar{\boldsymbol{v}})}{\rho_r(q^{-1}R^d)} \\
&\in C' \cdot \left[\frac{1-\epsilon}{1+\epsilon}, \frac{1+\epsilon}{1-\epsilon}\right]
\end{aligned} \tag{4.9}$$

where $C' := C\rho_r(q^{-1}R^d)$. Note that we may apply Lemma 6 here since we know that $r \geq \eta_\epsilon(q^{-1}R^d)$ by Lemma 4. This allows us to conclude that the distribution of $\boldsymbol{v}$ is within statistical distance $1 - [(1-\epsilon)/(1+\epsilon)]^2 \leq 4\epsilon$ of the uniform distribution. This means that $\boldsymbol{x}$ is uniformly random over $(R_{q'})^{d'}$ to within statistical distance 4ϵ implying that $\tilde{\boldsymbol{a}}$ is uniform random over $(R_{q'})^{d'}$ to within statistical distance 4ϵ by the data processing inequality. It is also clear that if b is uniform random, then so is $\tilde{b}$. This proves the first claim (uniform-to-uniform).

Distribution of $-\boldsymbol{f}$. In our analysis of the resulting error, it will be useful to understand the distribution of the vector $-\boldsymbol{f}$ for fixed $\tilde{\boldsymbol{a}}$ (and thus fixed $\boldsymbol{v} = \bar{\boldsymbol{v}}$). Note that fixing a value $\boldsymbol{f} = \bar{\boldsymbol{f}}$ fixes $\frac{1}{q}\boldsymbol{a} = \bar{\boldsymbol{v}} - \bar{\boldsymbol{f}} \bmod R^d$. By summing over all appropriate values of $\bar{\boldsymbol{f}}$ in Equation (4.8), one can show that the distribution of $-\boldsymbol{f}$ for any fixed $\tilde{\boldsymbol{a}}$ is within statistical distance $1 - (1-\epsilon)(1+\epsilon) \leq 2\epsilon$ of $D_{\frac{1}{q}R^d - \bar{\boldsymbol{v}}, r}$.

Distribution of the error. Suppose we are given the MLWE sample $(\boldsymbol{a}, b = \frac{1}{q}\langle \boldsymbol{a}, \boldsymbol{s}\rangle + e) \in (R_q)^d \times \mathbb{T}_{R^\vee}$ as input where $e \in K_{\mathbb{R}}$ is drawn from D_α. We have already shown that our map outputs $\tilde{\boldsymbol{a}} \in (R_{q'})^{d'}$ that is *almost* uniformly random. Now we condition on a fixed $\tilde{\boldsymbol{a}} = \bar{\tilde{\boldsymbol{a}}}$ and analyse the distribution of

$$(\tilde{b} - \frac{1}{q'}\langle \bar{\tilde{\boldsymbol{a}}} \cdot \tilde{\boldsymbol{s}}\rangle) \bmod R^\vee \tag{4.10}$$

where $\tilde{\boldsymbol{s}} = \mathbf{G}\boldsymbol{s}$. Let $\boldsymbol{f}_i \in \mathbb{R}^n$ be the vector consisting of the i^{th} block of n entries of $\boldsymbol{f} \in \mathbb{R}^{nd}$ for $i = 1, \ldots, d$. Using the fact that $\tilde{\boldsymbol{s}} = \mathbf{G}\boldsymbol{s}$ and that $R^\vee$ is closed under multiplication by elements of R, we can rewrite this as

$$(\tilde{b} - \frac{1}{q'}\langle \bar{\tilde{\boldsymbol{a}}} \cdot \tilde{\boldsymbol{s}}\rangle) = \sum_{i=1}^{d} s_i \cdot \sigma_H^{-1}(-\boldsymbol{f}_i) + \tilde{e} + e \bmod R^\vee. \tag{4.11}$$

We want to show that the RHS of this equation is almost distributed as a Gaussian in canonically embedded space H. To do so, define the invertible matrix $\mathbf{S}_{i,H} \coloneqq \mathbf{U}_H \mathbf{S}_i \mathbf{U}_H^\dagger \in \mathbb{R}^{n\times n}$ where $\mathbf{U}_H$ is given in Equation (2.5) and $\mathbf{S}_i$ is the diagonal matrix with the field embeddings of s_i along the diagonal i.e. $[\mathbf{S}_i]_{jk} = \sigma_j(s_i)\delta_{jk}$. Note that $\mathbf{S}_{i,H}$ is the matrix representing field multiplication by s_i in the basis $(\boldsymbol{h}_i)_{i=1}^n$ of H. Therefore, in canonical space, the error is given by

$$\sum_{i=1}^{d} \mathbf{S}_{i,H} \cdot (-\boldsymbol{f}_i) + \sigma_H(\tilde{e}) + \sigma_H(e) \bmod R^\vee \tag{4.12}$$

where $\sigma_H(\tilde{e})$ and $\sigma_H(e)$ are distributed as $D_{r\beta}$ and D_α respectively. Note that we can conceptualise $\sigma_H(\tilde{e})$ as $\sum_{i=1}^d \tilde{e}^{(i)}$ where each $\tilde{e}^{(i)}$ is distributed as a continuous spherical Gaussian in $\mathbb{R}^n$ with parameter $\gamma_i \geq rB$ provided that $\sum_{i=1}^d \gamma_i^2 = r^2\beta^2$. Also, letting $\bar{\boldsymbol{v}}_i$ denote the i^{th} block of n coordinates of $\bar{\boldsymbol{v}}$, we know that $-\boldsymbol{f}_i$ is *almost* distributed as $D_{\frac{1}{q}R - \bar{\boldsymbol{v}}_i, r}$. Therefore, writing

$$\sum_{i=1}^{d} \mathbf{S}_{i,H} \cdot (-\boldsymbol{f}_i) + \sigma_H(\tilde{e}) = \sum_{i=1}^{d} \mathbf{S}_{i,H} \cdot (-\boldsymbol{f}_i) + \tilde{e}^{(i)}, \tag{4.13}$$

and applying Lemma 12 to the summand on the RHS, we find (using the triangle and data processing inequalities for statistical distance) that the error term in Equation (4.12) is a statistical distance of at most $2\epsilon + 2d\epsilon$ away from $D_{\boldsymbol{\alpha}'}$. □

The following corollary specialises to a map from MLWE in module rank d to d/k and from modulus q to q^k for general rings. Taking $k = d$ constitutes a reduction from

MLWE to RLWE. Note that the new secret distribution is non-standard in general, but we can always use the usual re-randomizing process to obtain a uniform secret i.e. sample $s' \leftarrow U(R_q^\vee)$ and transform (a, b) to $(a, b + \frac{1}{q}(a \cdot s'))$.

Corollary 2. *Let R be a ring with basis $\boldsymbol{B}_R$ in the canonical embedding and χ be a distribution satisfying*

$$\Pr_{s \leftarrow \chi}\left[\max_i |\sigma_i(s)| > B\right] \leq \delta$$

for some (B, δ). Also take any $\alpha > 0$ and any $\epsilon \in (0, 1/2)$. For any $k > 1$ that divides d and

$$r \geq \frac{1}{q} \, \|\tilde{\boldsymbol{B}}_R\| \, \cdot \sqrt{2\ln(2nd(1 + 1/\epsilon))/\pi},$$

there is an efficient reduction from $\mathsf{MLWE}^{(R^d)}_{m,q,\Psi_{\leq\alpha}}(\chi^d)$ to $\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,\Psi_{\leq\alpha'}}(\boldsymbol{G} \cdot \chi^d)$ where $\boldsymbol{G} = \boldsymbol{I}_{d/k} \otimes (1, q, \ldots, q^{k-1}) \in R^{d/k \times d}$ and

$$(\alpha')^2 \geq \alpha^2 + 2r^2B^2d.$$

Moreover, this reduction reduces the advantage by at most $[1-(1-\delta)^d]+(2d+10)\epsilon m$.

Proof. We run the reduction from Theorem 5, taking $q' = q^k$, $\beta^2 \geq B^2d$ and $\mathbf{G} \in R^{d/k \times d}$ as in the corollary statement. The main task is to show that the conditions on r in the theorem are satisfied by the choice of r in this corollary. In particular, for $\Lambda = \frac{1}{q'}\mathbf{G}_H^T R^{d'} + R^d$, we will attempt to express $\|\tilde{\mathbf{B}}_\Lambda\|$ in terms of $\|\tilde{\mathbf{B}}_R\|$. Define $\boldsymbol{g} := (1, q, \ldots, q^{k-1})$ so that $\mathbf{G} = \mathbf{I}_{d/k} \otimes \boldsymbol{g}$. In the canonical embedding, we may write the lattice from Theorem 5 as $\Lambda = \frac{1}{q^k}(\mathbf{I}_{d/k} \otimes \boldsymbol{g}^T \otimes \mathbf{I}_n) \cdot (\mathbf{I}_{d/k} \otimes \mathbf{B}_R) \cdot \mathbb{Z}^{nd/k} + (\mathbf{I}_d \otimes \mathbf{B}_R) \cdot \mathbb{Z}^{nd}$. Pre-multiplying by $\mathbf{I}_d \otimes \mathbf{B}_R^{-1}$, we get the related lattice

$$\begin{aligned}
\Lambda' &:= \left(\mathbf{I}_d \otimes \mathbf{B}_R^{-1}\right) \cdot \Lambda \\
&= \frac{1}{q^k}\left(\mathbf{I}_{d/k} \otimes \boldsymbol{g}^T \otimes \mathbf{B}_R^{-1}\right) \cdot \left(\mathbf{I}_{d/k} \otimes \mathbf{B}_R\right) \cdot \mathbb{Z}_q^{nd/k} + \mathbb{Z}^{nd} \\
&= \frac{1}{q^k}\left(\mathbf{I}_{d/k} \otimes \boldsymbol{g}^T \otimes \mathbf{I}_n\right) \cdot \mathbb{Z}^{nd/k} + \mathbb{Z}^{nd}
\end{aligned}$$

The lattice $\bar{\Lambda}$ can be shown to have basis $\mathbf{B}' = \mathbf{I}_{d/k} \otimes \mathbf{Q} \otimes \mathbf{I}_n$ where

$$\mathbf{Q} = \begin{bmatrix} q^{-1} & q^{-2} & \cdots & q^{-k} \\ & q^{-1} & \cdots & q^{1-k} \\ & & \ddots & \vdots \\ & & & q^{-1} \end{bmatrix}.$$

Pre-multiplying $\mathbf{B}'$ by $\mathbf{I}_d \otimes \mathbf{B}_R$ gives us a basis $\mathbf{B}_\Lambda = \mathbf{I}_{d/k} \otimes \mathbf{Q} \otimes \mathbf{B}_R$ for Λ. Orthogonalising from left to right, we can see that $\|\tilde{\mathbf{B}}_\Lambda\|$ is precisely $\frac{1}{q}\|\tilde{\mathbf{B}}_R\|$.

Finally, the loss in advantage can be derived from the statistical distances in Theorem 5 and the fact that the reduction is only guaranteed to work when each of the d secret polynomials has embeddings of modulus at most B (which occurs with probability at least $(1-\delta)^d$). □

4.3.3 Normal Form Secret Distribution

There are well-known results stating that an LWE problem where the secret is drawn from the error distribution is at least as hard as an LWE problem where the secret is uniform [12]. The connection is straight-forward for plain LWE, but for R/MLWE, we need to remember that the secrets lie in $(R^\vee)^d$ whereas the errors are sampled from $K \otimes \mathbb{R}$. However, this complication can be solved using discretisation techniques that transform continuous errors to discrete ones [98]. Using discretization, Langlois et al. showed that $\mathsf{MLWE}^M_{q,D_\alpha}(U)$ is at least as hard as $\mathsf{MLWE}^M_{q,D_{\frac{1}{q}R^\vee,\sqrt{2}\alpha}}\left(D_{(R^\vee)^d,\sqrt{2}q\alpha}\right)$ [81]. We can also add continuous Gaussian noise to $\mathsf{MLWE}^M_{q,D_{\frac{1}{q}R^\vee,\sqrt{2}\alpha}}\left(D_{(R^\vee)^d,\sqrt{2}q\alpha}\right)$ challenges in an attempt to reintroduce continuous noise distributions. By Claim 3.9 from [118], adding the noise $D_{\sqrt{2}q\alpha}$ results in challenges that are statistically close to those from $\mathsf{MLWE}^M_{q,D_{2\alpha}}\left(D_{(R^\vee)^d,\sqrt{2}q\alpha}\right)$ provided that $\alpha q \geq \|\tilde{\mathbf{B}}_{R^\vee}\| \cdot \tilde{O}(1)$. An informal summary of this is that we may use a discrete Gaussian secret with a continuous Gaussian error distribution roughly q times narrower without compromising hardness.

Now that we have established the significance of secret distributions of the form $D_{(R^\vee)^d,\sqrt{2}q\alpha}$, we next discuss valid choices of (B,δ) with respect to this secret distribution. The below lemma shows that we can choose $B = \sqrt{2}q\alpha n^c$ for any positive constant c along with $\delta = 2n\exp\left(-\pi n^{2c}\right)$ which is negligible assuming that n is polynomial in the security parameter.

Lemma 13. *For any algebraic number field K of degree n with ring of integers R, any $\sigma > 0$ and any constant $c > 0$*

$$\Pr_{s\leftarrow D_{R^\vee,\sigma}}\left[\max_i |\sigma_i(s)| > \sigma n^c\right] \leq 2n\exp\left(-\pi n^{2c}\right).$$

Proof. We first recall that $s \leftarrow D_{(R^\vee),\sqrt{2}q\alpha}$ means that $\sigma_H(s) \leftarrow D_{\sigma_H((R^\vee)),\sqrt{2}q\alpha}$. Assuming r_1 real embeddings and r_2 pairs of complex embeddings so that $n =$

$r_1 + 2r_2$, we have that

$$\sigma_H(s) = \begin{bmatrix} \sigma_1(s) \\ \vdots \\ \sigma_{r_1}(s) \\ \sqrt{2}\ \mathrm{Re}(\sigma_{r_1+1}(s)) \\ \vdots \\ \sqrt{2}\ \mathrm{Re}(\sigma_{r_1+r_2}(s)) \\ \sqrt{2}\ \mathrm{Im}(\sigma_{r_1+1}(s)) \\ \vdots \\ \sqrt{2}\ \mathrm{Im}(\sigma_{r_1+r_2}(s)) \end{bmatrix}.$$

It is clear that $|\sigma_i(s)| \leq |\sigma_H(s)|_\infty$ for $i = 1, \ldots, r_1$. For $i = r_1 + 1, \ldots, r_1 + r_2$, we have that $2|\sigma_i(s)|^2 = \sigma_H(s)_i^2 + \sigma_H(s)_{i+r_2}^2 \leq 2|\sigma_H(s)|_\infty^2$. Therefore, we have that $\max_i |\sigma_i(s)| \leq |\sigma_H(s)|_\infty$. Applying Lemma 1 to $\sigma_H(s)$ completes the proof. □

Corollary 3. *Let R be a ring with basis $\boldsymbol{B}_R$ in the canonical embedding, $c > 0$ be an arbitrary constant and χ denote the distribution $D_{(R^\vee)^d, \alpha q}$. Also take any $\alpha > 0$ and any $\epsilon \in (0, 1/2)$. For any $k > 1$ that divides d and*

$$r \geq \frac{1}{q} \|\tilde{\boldsymbol{B}}_R\| \cdot \sqrt{2\ln(2nd(1 + 1/\epsilon))/\pi},$$

there is an efficient reduction from $\mathsf{MLWE}^{(R^d)}_{m,q,\Psi_{\leq\alpha}}(\chi^d)$ to $\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,\Psi_{\leq\alpha'}}(\boldsymbol{G} \cdot \chi^d)$ where $\boldsymbol{G} = \boldsymbol{I}_{d/k} \otimes (1, q, \ldots, q^{k-1}) \in R^{d/k \times d}$ and

$$(\alpha')^2 \geq \alpha^2 \left(1 + 2(rqn^c)^2 d\right).$$

Moreover, this reduction reduces the advantage by at most $[1-(1-\delta)^d]+(2d+10)\epsilon m$ where $\delta = 2n\exp(-\pi n^2 c)$.

4.3.4 Instantiation: Power-of-Two Cyclotomic Rings

We now consider the case of cyclotomic rings with power-of-two dimension n. It can be shown that the map taking the coefficient embedding to the canonical embedding is a scaled isometry with scaling factor $\sqrt{n}$. In this case, we have $R = \mathbb{Z}(\xi)$ for $(2n)^{th}$ primitive root of unity ξ. Taking the "power basis" of R given by $1, \xi, \ldots, \xi^{n-1}$, gives us an orthonormal lattice basis of R in the coefficient embedding. Applying the aforementioned scaled isometry, we find an *orthogonal* basis in the canonical

embedding where each vector has length $\sqrt{n}$. Therefore, in the canonical embedding $\|\tilde{\mathbf{B}}_R\| = \sqrt{n}$ when using this basis.

In the following, we will informally take normal form MLWE to mean the case where the error distribution is D_α for some α and where the secret distribution is $D_{R^\vee, q\alpha}$ (i.e. we will ignore any small constant factor differences between the Gaussian parameters of the secret and error distributions). We assume that m, n and d are $\mathsf{poly}(\lambda)$. We will also use $\delta = 2n\exp\left(-\pi n^{2c}\right)$, $\epsilon = n^{-\log n}$ as negligible functions while ignoring constant/logarithmic factors. In this setting, the losses in advantage are negligible. Taking the above into account, Corollary 3 shows we can reduce normal form MLWE in modulus q, module rank d to MLWE in modulus q^k, module rank d/k with a uniform secret distribution (after re-randomising the secret). In particular, the bound on the width of the error distribution grows from α to roughly $n^{c+1/2}\sqrt{d}\alpha$ for any positive constant c. Since normal form MLWE is at least as hard as uniform secret MLWE, we also have that uniform secret MLWE in modulus q, rank d reduces to uniform secret MLWE in modulus q^k, rank d/k at the cost of roughly a factor $n^{c+1/2}\sqrt{d}$ blow-up in the bound on the error rate. Finally, taking $k = d$ gives us the result that RLWE in modulus q^d is at least as hard as MLWE with modulus q, rank d with error rate roughly $n^{c+1/2}\sqrt{d}$ smaller than the RLWE error rate bound.

4.3.5 Strictly Spherical Error Distributions

Note that the reduction presented in Theorem 5 results in a skewed, but bounded error distribution. We will now present a lemma that allows us to reduce from MLWE to MLWE with a *spherical* error distribution following the strategy laid out in [97]. The price paid when targeting a strictly spherical error distribution is a larger (but still polynomial) blow-up in the error rate. Corollary 4 shows that the extra blow-up factor incurred when targeting a strictly spherical distribution can be as low as $(mn)^{1/4}$ where m is the number of MLWE samples provided. It is important to note that we will be using the Renyi divergence to carry out this analysis. As a result, the analysis only applies to the *search variants* of the MLWE problem. Note that the reduction and analysis of Theorem 5 implicitly contains a reduction and analysis between *search* variants of MLWE. This is because the reduction takes the distribution $A^{(M)}_{q,\boldsymbol{s},D_\alpha}$ to a distribution statistically close to $A^{(M)}_{q,\boldsymbol{s},D_\alpha}$. It must be noted

that the reduction maps a secret $\boldsymbol{s} \in (R^\vee)^d$ to $\tilde{\boldsymbol{s}} = \mathbf{G} \cdot \boldsymbol{s}$, so we only have a search to search reduction when this mapping is efficiently invertible between the spaces $(R_q^\vee)^d$ and $(R_{q^k}^\vee)^{d/k}$ as is the case for $\mathbf{G} = \mathbf{I}_{d/k} \otimes (1, q, \ldots, q^{k-1})$.

Lemma 14. *For integers m, n, let $\boldsymbol{M} \in \mathbb{R}^{m \times n}$ be a matrix with non-zero singular values σ_i for $i = 1, \ldots, n$ such that $\sigma_1^2 \geq \cdots \geq \sigma_n^2$ and take $\beta^2 \geq \sigma_1^2$. Then*

- $R_2\left(D_{r\beta} \| D_{r(\beta^2 \boldsymbol{I} + \boldsymbol{M}^T \boldsymbol{M})^{1/2}}\right) \leq \left(1 + \frac{\sigma_1^4}{\beta^4}\right)^{n/2}$,
- $R_\infty\left(D_{r\beta} \| D_{r(\beta^2 \boldsymbol{I} + \boldsymbol{M}^T \boldsymbol{M})^{1/2}}\right) \leq \left(1 + \frac{\sigma_1^2}{\beta^2}\right)^{n/2}$.

Proof. To prove this lemma, simply work in the orthogonal basis where the matrix $\mathbf{M}^T\mathbf{M}$ takes a diagonal form. For the first claim,

$$
\begin{aligned}
&R_2\left(D_{r\beta} \| D_{r(\beta^2 \mathbf{I} + \mathbf{M}^T \mathbf{M})^{1/2}}\right) \\
&= \prod_{i=1}^{n} \frac{\sqrt{r^2(\beta^2 + \sigma_i^2)}}{r^2\beta^2} \int_{\mathbb{R}} \exp\left[-\pi x_i^2 \left(\frac{2}{r^2\beta^2} - \frac{1}{r^2(\beta^2 + \sigma_i^2)}\right)\right] dx_i \\
&= \prod_{i=1}^{n} \sqrt{\frac{\beta^2 + \sigma_i^2}{r^2\beta^4}} \int_{\mathbb{R}} \exp\left[-\pi x_i^2 \left(\frac{\beta^2 + 2\sigma_i^2}{r^2\beta^2(\beta^2 + \sigma_i^2)}\right)\right] dx_i \\
&= \prod_{i=1}^{n} \sqrt{\frac{\beta^2 + \sigma_i^2}{r^2\beta^4}} \cdot \sqrt{\frac{r^2\beta^2(\beta^2 + \sigma_i^2)}{\beta^2 + 2\sigma_i^2}} = \prod_{i=1}^{n} \sqrt{\frac{(\beta^2 + \sigma_i^2)^2}{\beta^4 + 2\beta^2\sigma_i^2}} \\
&= \prod_{i=1}^{n} \sqrt{1 + \frac{\sigma_i^4}{\beta^4 + 2\beta^2\sigma_i^2}} \leq \left(1 + \frac{\sigma_1^4}{\beta^4}\right)^{n/2}.
\end{aligned}
$$

For the second claim, we have

$$
\begin{aligned}
&R_\infty\left(D_{r\beta} \| D_{r(\beta^2 \mathbf{I} + \mathbf{M}^T \mathbf{M})^{1/2}}\right) \\
&= \max_{\boldsymbol{x} \in \mathbb{R}^n} \left(\prod_{i=1}^{n} \sqrt{\frac{\beta^2 + \sigma_i^2}{\beta^2}} \cdot \exp\left[-\pi x_i^2 \left(\frac{\sigma^2}{r^2\beta^2(\beta^2 + \sigma_i^2)}\right)\right]\right) \\
&= \prod_{i=1}^{n} \sqrt{\frac{\beta^2 + \sigma_i^2}{\beta^2}} \leq \left(1 + \frac{\sigma_1^2}{\beta^2}\right)^{n/2}.
\end{aligned}
$$

□

We now use the above lemma to show that there is a search MLWE to search MLWE reduction where the resulting error distribution is essentially spherical.

Corollary 4. *Let R be a ring with basis $\boldsymbol{B}_R$ in the canonical embedding and χ be a distribution satisfying*

$$\Pr_{s\leftarrow\chi}\left[\max_i |\sigma_i(s)| > B\right] \leq \delta$$

for some (B,δ). Also take any $\alpha > 0$, any $\epsilon \in (0,1/2)$, any $k > 1$ that divides d,

$$r \geq \frac{1}{q}\ \|\tilde{\boldsymbol{B}}_R\|\ \cdot\sqrt{2\ln(2nd(1+1/\epsilon))/\pi}$$

and define $t := \sqrt{\alpha^2 + (rB(mn)^{1/4})^2}$. Suppose there exists a PPT algorithm solving S-$\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,D_t}(\boldsymbol{G}\cdot\chi^d)$ where $\boldsymbol{G} = \boldsymbol{I}_{d/k}\otimes(1,q,\ldots,q^{k-1}) \in R^{d/k\times d}$ with probability p. Then there is a PPT algorithm solving S-$\mathsf{MLWE}^{(R^d)}_{m,q,D_\alpha}(\chi^d)$ with probability at least $\frac{(1-\delta)^d p^2}{2} - \big(1-(1-\delta)^d + (2d+6)\epsilon m\big)$. Alternatively, if we define $t := \sqrt{\alpha^2 + (rB(mn)^{1/2})^2}$, there is a PPT algorithm solving S-$\mathsf{MLWE}^{(R^d)}_{m,q,D_\alpha}(\chi^d)$ with probability at least $\frac{(1-\delta)^d p}{2} - \big(1-(1-\delta)^d + (2d+6)\epsilon m\big)$.

Proof. We run the reduction from Theorem 5 choosing $\beta = B(mn)^{1/4}\sqrt{d}$. The resulting error distribution has a diagonal covariance with i^{th} entry $(\boldsymbol{\alpha'})_i = \alpha^2 + r^2(\beta^2 + \sum_{j=1}^d |\sigma_i(s_j)|^2)$ where the s_j are the components of the original MLWE secret. In addition, the loss in success probability is at most $1-(1-\delta)^d + (4d+6)\epsilon$.

Next consider the loss in success probability when applying an algorithm solving S-$\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,D_{\boldsymbol{\alpha'}}}(\mathbf{G}\cdot\chi^d)$ with probability p to the S-$\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,D_t}(\mathbf{G}\cdot\chi^d)$ problem. We can apply Lemma 14 after setting $\mathbf{M}$ to be the diagonal matrix with $\sqrt{\sum_{j=1}^d |\sigma_i(s_j)|^2}$ in the i^{th} position. Conditioned on B being larger than $\max_i |\sigma_i(s_j)|$ for $j = 1,\ldots,d$ (which occurs with probability $(1-\delta)^d$), we find that

$$R_2\left((D_{r\beta})^m \| (D_{r(\beta^2\mathbf{I}+\mathbf{M}^T\mathbf{M})^{1/2}})^m\right) \leq \left(1+\frac{\sigma_1^4}{\beta^4}\right)^{mn/2} \leq \left(1+\frac{1}{mn}\right)^{mn/2}.$$

This quantity is upper bounded by $\exp(1/2) \leq 2$. By the data processing inequality, 2 is also an upper bound for the Renyi divergence between the MLWE distribution with error distributions D_t and $D_{\boldsymbol{\alpha'}}$. Therefore, a PPT algorithm solving S-$\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,D_{\boldsymbol{\alpha'}}}(\mathbf{G}\cdot\chi^d)$ with probability p also solves S-$\mathsf{MLWE}^{(R^{d/k})}_{m,q^k,D_t}(\mathbf{G}\cdot\chi^d)$ with probability at least $(1-\delta)^d\cdot p^2/2$. Repeating the analysis for Renyi divergences of order infinity completes the proof. □

Remark 8. *We can write Theorem 5 in terms of the Renyi divergence of order infinity instead of statistical distances to obtain slightly better parameters. Phrasing Theorem 5 in terms of Renyi divergence allows us to take $\epsilon = \mathcal{O}(1)$ allowing for smaller choices of r (by logarithmic factors). We do not spell this out to avoid repetition, but note that all statistical distances in the proof of Theorem 5 are derived by bounding the ratio between mass functions i.e. by essentially calculating order infinity Renyi divergences.*

4.3.6 Modulus Reduction

Although we have focused on reductions between MLWE problems of different module rank, we can also use Theorem 5 to derive a modulus reduction result by taking $d' = d$, $\mathbf{G} = \mathbf{I}$ and large enough $q' < q$. This is formalised in the following corollary.

Corollary 5. *Let R be a ring with basis $\boldsymbol{B}_R$ in the canonical embedding and χ be a distribution satisfying*

$$\Pr_{s \leftarrow \chi}\left[\max_i |\sigma_i(s)| > B\right] \leq \delta$$

for some (B, δ). Also take any $\alpha > 0$, any $\epsilon \in (0, 1/2)$, any $q' < q$ and

$$r \geq \frac{1}{q'} \cdot \|\tilde{\boldsymbol{B}}_R\| \cdot \sqrt{2\ln(2nd(1+1/\epsilon))/\pi},$$

there is an efficient reduction from $\mathsf{MLWE}^{(R^d)}_{m,q,\Psi_{\leq\alpha}}(\chi^d)$ to $\mathsf{MLWE}^{(R^d)}_{m,q',\Psi_{\leq\alpha'}}$ where

$$(\alpha')^2 \geq \alpha^2 + 2r^2B^2d.$$

Moreover, this reduction reduces the advantage by at most $[1-(1-\delta)^d]+(2d+10)\epsilon m$.

As an example instantiation, we can take the case of normal form MLWE over power-of-two cyclotomic rings. As discussed previously, $\|\tilde{\mathbf{B}}_R\| = \sqrt{n}$ and the choice $B = \alpha q n^c$ for any constant c allows for a negligible δ. Therefore, applying the corollary above to power of two cyclotomic rings, the reduction increases the bound on the error rate from α to roughly $\frac{qn^{c+1/2}\sqrt{d}}{q'} \cdot \alpha$.

4.4 Reducing RLWE in (n, q) to $(n/2, q^2)$

Throughout this entire section, we assume that n is a power of two. We will be using similar techniques to the above to show that there is an efficient reduction between RLWE in cyclotomic ring of dimension n and modulus q to RLWE in ring dimension $n/2$ and q^2. We also note that there is a less direct route leading to a stronger result than that given in this section. This alternative strategy uses recent results from [109] along with the reduction in the previous section. Details of the stronger result are given in Section 4.5.1.

4.4.1 Intuition

The reduction strategy is to represent polynomial multiplications in the *coefficient* embedding using $n \times n$ matrices. The reduction follows the same blueprint as in Section 4.3 apart from the fact that we are no longer working exclusively in the canonical embedding. Since we are considering power-of-two cyclotomic rings, polynomial multiplication is always represented by a matrix of the form given in Equation (2.4). Going from ring dimension n to $n/2$ just halves the dimension of these matrices. For clarity, we adopt the notation $R_{n,q} = \mathbb{Z}_q[X]/\langle X^n + 1\rangle$ and $R_n = \mathbb{Z}[X]/\langle X^n + 1\rangle$.

Our aim is to reduce RLWE in dimension and modulus (n, q) to RLWE in $(n/2, q^2)$ via some mapping $a \in R_{n,q} \longmapsto \tilde{a} \in R_{n/2,q^2}$, $b \in \mathbb{T}_{R_n^\vee} \longmapsto \tilde{b} \in \mathbb{T}_{R_{n/2}^\vee}$, $s \in R_{n,q}^\vee \longmapsto \tilde{s} \in R_{n/2,q^2}^\vee$. We can start by defining a relationship between $\mathrm{rot}(s)$ and $\mathrm{rot}(\tilde{s})$ where $\mathrm{rot}(\cdot)$ is a negacyclic matrix as in Equation (2.4). In order to make clear the distinction between the two rings, we denote $n \times n$ matrices associated with multiplications in $R_{n,q}$ by writing the subscript n, q. Given a particular choice of $\mathbf{G}, \mathbf{H} \in \mathbb{Z}^{n/2 \times n}$ (see below), the linear relationship will be defined via the equation

$$\mathrm{rot}(\tilde{s})_{n/2,q^2} = 2 \cdot \mathbf{H} \cdot \mathrm{rot}(s)_{n,q} \cdot \mathbf{G}^T. \tag{4.14}$$

Note that the factor of 2 is present to account for the fact that the new secret should be in the dual ring $R_{n/2,q^2}^\vee = \frac{2}{n} R_{n/2}$ and the matrix $\mathbf{H}$ ensures that we end up with a square matrix $\mathrm{rot}(\tilde{s})_{n/2,q^2}$. We also need to be careful that $\mathbf{G}$ and $\mathbf{H}$ are chosen so the matrix $\mathrm{rot}(\tilde{s})_{n/2,q^2}$ has the correct form. We will write $\boldsymbol{b} \in \mathbb{R}^n$ and $\tilde{\boldsymbol{b}} \in \mathbb{R}^{n/2}$ as the coefficient vectors of $b \in \mathbb{T}_{R_n^\vee}$ and $\tilde{b} \in \mathbb{T}_{R_{n/2}^\vee}$ respectively. Define the map

between b and $\tilde{b}$ (up to some Gaussian error) as

$$\tilde{\boldsymbol{b}} \approx 2\mathbf{H} \cdot \boldsymbol{b}.$$

In order for the reduction to work, we require that $\tilde{b} \approx \tilde{a} \cdot \tilde{s}/q^2 \bmod R^{\vee}_{n/2}$. Writing this in terms of coefficient vectors,

$$2\mathbf{H} \cdot \mathrm{rot}(s)_{n,q} \cdot \frac{1}{q}\boldsymbol{a} \approx 2 \cdot \mathbf{H} \cdot \mathrm{rot}(s)_{n,q} \cdot \mathbf{G}^T \cdot \frac{1}{q^2}\tilde{\boldsymbol{a}} \bmod 2/n.$$

It is easy to see that we can satisfy this requirement by choosing $\tilde{a}$ such that

$$\frac{1}{q}\boldsymbol{a} = \frac{1}{q^2}\mathbf{G}^T \cdot \tilde{\boldsymbol{a}} \bmod 1.$$

Explicit forms for our choice of $\mathbf{G}$ and $\mathbf{H}$ are

$$\mathbf{G} = \mathbf{I}_{n/2} \otimes (1, q) \in \mathbb{Z}^{n/2 \times n}, \tag{4.15}$$

$$\mathbf{H} = \mathbf{I}_{n/2} \otimes (1, 0) \in \mathbb{Z}^{n/2 \times n}. \tag{4.16}$$

Claim 3. *Take $\boldsymbol{G}$ and $\boldsymbol{H}$ as above. Then $\mathrm{rot}(\tilde{s})_{n/2,q^2}$ is of the correct form (i.e. represents multiplication by some polynomial in $(R_{n/2,q^2})$).*

Proof. We can write simple explicit forms $(\mathbf{G}^T)_{kl} = \delta_{k,2l-1} + q\delta_{k,2l}$ and $(\mathbf{H})_{ij} = \delta_{2i-1,j}$. Then the matrix multiplication $\mathbf{H} \cdot \mathrm{rot}(s)_{n,q} \cdot \mathbf{G}^T$ yields $(\mathrm{rot}(\tilde{s})_{n/2,q^2})_{il} = 2(\mathrm{rot}(s)_{n,q})_{2i-1,2l-1} + 2(q\mathrm{rot}(s)_{n,q})_{2i-1,2l}$ which is of the correct form. □

Note that the mapping between secrets is

$$s = \sum_{i=0}^{n-1} s_i \cdot X^i \longmapsto \tilde{s} = 2(s_0 - qs_{n-1}) + 2\sum_{i=1}^{n/2-1} (s_{2i} + qs_{2i-1}) \cdot X^i. \tag{4.17}$$

4.4.2 Proof of Correctness

Now the proof of correctness for this reduction is essentially the same as Theorem 5 with a few minor differences. The first is that it is convenient to perform part of the analysis in the coefficient embedding. The second is that we can only show a reduction resulting in an elliptical Gaussian error distribution that is *not* diagonal in canonical space. Such error distributions are non-standard. Therefore, we follow this theorem with a corollary that uses Renyi divergence analysis to obtain a spherical error distribution for search variants of RLWE.

Theorem 6. *Let n be a power of two, q be an integer, fix $s \in R^\vee_{n,q}$ and*

$$r \geq \frac{\sqrt{n}}{q} \cdot \sqrt{2\ln(2n(1+1/\epsilon))/\pi}.$$

Define $\boldsymbol{G}$, $\boldsymbol{H}$ and $\tilde{s}$ as in Equations (4.15), (4.16) and (4.17) respectively. Further, define $\sigma_i := |\sigma_i(s)|$.

For any $\alpha > 0$ and a particular $\boldsymbol{H'} \in \mathbb{R}^{n/2 \times n}$, there exists an efficient probabilistic mapping $\mathcal{F} : R_{n,q} \times \mathbb{T}_{R^\vee_{n,q}} \to R_{n/2,q} \times \mathbb{T}_{R^\vee_{n/2,q^2}}$ such that:

1. *The statistical distance between $\mathcal{F}(U(R_{n,q} \times \mathbb{T}_{R^\vee_{n,q}}))$ is within statistical distance 4ϵ of $U(R_{n/2,q} \times \mathbb{T}_{R^\vee_{n/2,q^2}})$.*
2. *The statistical distance between $\mathcal{F}(A^{R_n}_{q,s,D_\alpha})$ and $A^{R_{n/2}}_{q^2,\tilde{s},2\boldsymbol{H'}\cdot D_{\boldsymbol{\alpha'}}}$ is at most 8ϵ where $(\alpha')_i{}^2 = \alpha^2 + r^2(\beta^2 + |\sigma_i(s)|^2)$ for any $\beta \geq \max_i |\sigma_i(s)|$.*

Proof. Let all vectors denote coefficient vectors of the corresponding ring/field elements e.g. $\boldsymbol{s}$ is the coefficient vector of s. Suppose we are given $(a, b) \in R_{n,q} \times \mathbb{T}_{R^\vee_{n,q}}$ and take $\mathbf{G}, \mathbf{H} \in \mathbb{Z}^{n/2 \times n}$ as in Equations (4.15) and (4.16) respectively. The mapping $\mathcal{F}$ is performed as follows:

1. Sample $\boldsymbol{f} \leftarrow D_{\Lambda - \frac{1}{q}\boldsymbol{a}, \frac{r}{\sqrt{n}}}$ over the lattice $\Lambda = \frac{1}{q^2}\mathbf{G}^T\mathbb{Z}^{n/2} + \mathbb{Z}^n$. Note that the parameter $r/\sqrt{n}$ is large enough so we can sample the discrete Gaussian efficiently by Lemma 7 since $\|\tilde{\mathbf{B}}_\Lambda\| = q^{-1}$ (c.f. the lattice in Corollary 2).
2. Let $\boldsymbol{v} = \frac{1}{q}\boldsymbol{a} + \boldsymbol{f} \in \Lambda/\mathbb{Z}^n$ and set $\boldsymbol{x}$ to be a random solution of $\frac{1}{q^2}\mathbf{G}^T\boldsymbol{x} = \boldsymbol{v} \bmod 1$. Then set $\tilde{a} \in R_{n/2,q^2}$ to be the unique polynomial such that $\tilde{\boldsymbol{a}} = \boldsymbol{x}$.
3. Sample $\tilde{e} \leftarrow D_{r\beta/\sqrt{n}}$ over $K_\mathbb{R} \simeq H \simeq \mathbb{R}^{n/2}$ and set $\tilde{b}$ to be the element of $\mathbb{T}_{R_{n/2,q^2}{}^\vee}$ corresponding to $2\mathbf{H} \cdot (\boldsymbol{b} + \tilde{\boldsymbol{e}})$.
4. Finally, output $(\tilde{a}, \tilde{b}) \in (R_{n/2,q^2}) \times \mathbb{T}_{R^\vee_{n/2,q^2}}$.

Distribution of $\tilde{a}$: We follow the steps in the proof of Theorem 5 so keep details to a minimum. Since $r/\sqrt{n} \geq \eta_\epsilon(\Lambda)$ (which is implicitly shown in the proof of

Corollary 2), we can show using Lemma 6 that

$$\Pr[\boldsymbol{a} = \bar{\boldsymbol{a}} \wedge \boldsymbol{f} = \bar{\boldsymbol{f}}] \in C \cdot \left[1, \frac{1+\epsilon}{1-\epsilon}\right] \cdot \rho_{r/\sqrt{n}}(\bar{\boldsymbol{f}}) \tag{4.18}$$

for some constant C. Next, by applying Lemma 6 after noting that $r/\sqrt{n} \geq \eta_\epsilon(\Lambda)$ we can show that

$$\begin{aligned} \Pr[\boldsymbol{v} = \bar{\boldsymbol{v}}] &\in C \cdot \left[1, \frac{1+\epsilon}{1-\epsilon}\right] \cdot \rho_{r/\sqrt{n}}(q^{-1}\mathbb{Z}^n + \bar{\boldsymbol{v}}) \\ &\in C' \cdot \left[\frac{1-\epsilon}{1+\epsilon}, \frac{1+\epsilon}{1-\epsilon}\right] \end{aligned} \tag{4.19}$$

where $C' := C\rho_{r/\sqrt{n}}(q^{-1}\mathbb{Z}^n)$. This implies that the distribution of $\boldsymbol{v}$ and therefore $\tilde{a}$ is within statistical distance 4ϵ of uniform (since each value of $\boldsymbol{v}$ has the same number of possible values for $\bar{\boldsymbol{a}}$). If b is also uniform, then $\tilde{b}$ is also uniform. This shows that the reduction maps the uniform distribution over $R_{n,q} \times \mathbb{T}_{R^\vee_{n,q}}$ to the uniform distribution over $R_{n/2,q^2} \times \mathbb{T}_{R^\vee_{n/2,q^2}}$ to within statistical distance 4ϵ. For the rest of the proof, we condition on a fixed value $\tilde{a} = \bar{\tilde{a}}$ i.e. fixed $\boldsymbol{a} = \bar{\boldsymbol{a}}$ and thus a fixed $\boldsymbol{v} = \bar{\boldsymbol{v}} = \frac{1}{q^2}\mathbf{G}^T\bar{\boldsymbol{a}} \bmod \mathbb{Z}^n$. We also assume that b is from the RLWE distribution.

Distribution of the error: Similarly to Theorem 5, we can show that the distribution of $-\boldsymbol{f}$ is within statistical distance 2ϵ of $D_{\frac{1}{q}\mathbb{Z}^n - \bar{\boldsymbol{v}}, r/\sqrt{n}}$. Using the definition of $\mathrm{rot}(s)$ and $\tilde{s}$ given in Equation (4.14), we have that $\left(\tilde{b} - \frac{1}{q^2}\bar{\tilde{a}} \cdot \tilde{s}\right)$ written in coefficient vectors is

$$\tilde{\boldsymbol{b}} - \frac{1}{q^2}\mathrm{rot}(\tilde{s}) \cdot \bar{\tilde{\boldsymbol{a}}} = 2\mathbf{H} \cdot (\mathrm{rot}(s) \cdot (-\boldsymbol{f}) + \boldsymbol{e} + \tilde{\boldsymbol{e}}) \bmod \frac{2}{n} \cdot \mathbb{Z}^{n/2} \tag{4.20}$$

where $\tilde{\boldsymbol{e}}$ (resp. $\boldsymbol{e}$) is drawn from the spherical distribution $D_{r\beta/\sqrt{n}}$ (resp. $D_{\alpha/\sqrt{n}}$).

All that is left is to analyse the distribution of this error. We begin by analysing the bracketed term i.e. ignoring the $2\mathbf{H}$ factor. We will map the bracketed term into n-dimensional canonical space and analyse the distribution there. Note that for the power-of-two cyclotomic ring we are using, the mapping is a scaled isometry with scaling factor $\sqrt{n}$. Therefore, in canonical space, $\mathrm{rot}(s) \cdot (-\boldsymbol{f}) + \boldsymbol{e} + \tilde{\boldsymbol{e}}$ becomes $\mathbf{S} \cdot (-\boldsymbol{f}') + \boldsymbol{e}^* + \tilde{\boldsymbol{e}}^*$ where

- $(-\boldsymbol{f}')$ is distributed closely to $D_{\Lambda', r}$ for Λ' a scaled rotation of Λ with scaling factor $\sqrt{n}$,

- $\boldsymbol{e}^*$ and $\tilde{\boldsymbol{e}}^*$ are distributed as D_α and $D_{r\beta}$ respectively,
- $\mathbf{S}$ corresponds to field multiplication by $s \in R^\vee$ in the canonical embedding.

Since we have that $r/\sqrt{n} \geq \sqrt{2}\eta_\epsilon(\Lambda)$, we also have that $r \geq \sqrt{2}\eta_\epsilon(\Lambda')$. Applying Lemma 12, we find that the distribution of $\mathbf{S} \cdot (-\boldsymbol{f}') + \tilde{\boldsymbol{e}}^*$ is within statistical distance 4ϵ of $D_{\boldsymbol{t}}$ where $t_i = r\sqrt{\beta^2 + |\sigma_i(s)|^2}$. This implies that the distribution of $(\mathrm{rot}(s) \cdot (-\boldsymbol{f}) + \boldsymbol{e}^* + \tilde{\boldsymbol{e}}^*)$ is within statistical distance 4ϵ of $D_{\boldsymbol{\alpha}'}$ in the canonical embedding. Setting $\mathbf{H}'$ to be the matrix corresponding to $\mathbf{H}$ in canonical space completes the proof. □

Corollary 6. *Let R_n denote the power of two cyclotomic ring of dimension $n \geq 2$ and χ be a distribution over $R_n^\vee$ satisfying*

$$\Pr_{s \leftarrow \chi}[\|\sigma_H(s)\|_\infty > B] \leq \delta$$

for some (B, δ). Also, let $\alpha > 0$ and $\epsilon \in (0, 1/2)$. For any

$$r \geq \frac{\sqrt{n}}{q} \cdot \sqrt{2\ln(2n(1 + 1/\epsilon))/\pi},$$

let $(\alpha_t')^2 = 2\alpha^2 + 2r^2B^2(mn)^{2t}$. Suppose there exists a PPT algorithm solving S-$\mathsf{RLWE}^{(R_{n/2})}_{m,q^2,D_{\alpha_t'}}(U(R^\vee_{n/2,q^2}))$ for $t = 1/4$ (resp. $t = 1/2$) with success probability $p_{1/4}$ (resp. $p_{1/2}$). Then there exists a PPT algorithm solving S-$\mathsf{RLWE}^{(R_n)}_{m,q,D_\alpha}(\chi)$ with success probabilities at least $(1-\delta)\frac{p_{1/4}^2}{2} - (\delta + 8\epsilon)$ and $(1-\delta)\frac{p_{1/2}}{2} - (\delta + 8\epsilon)$.

Proof. We will only go through the proof for $t = 1/4$ since the $t = 1/2$ case follows in an identical fashion. We first run the reduction in Theorem 6 with $\beta = B \cdot (mn)^{1/4}$ to transform our S-$\mathsf{RLWE}^{(R_n)}_{m,q,D_\alpha}(\chi)$ instance with secret s to within statistical distance $\delta + 10\epsilon m$ of a S-$\mathsf{RLWE}^{(R_{n/2})}_{m,q^2,2\mathbf{H}' \cdot D_{\boldsymbol{\alpha}'}}(\chi')$ problem where $(\alpha_i')^2 = \alpha^2 + r^2(\beta^2 + |\sigma_i(s)|^2)$. Note that the mapping between secrets is reversible and that the δ term arises from the fact that the reduction analysis only holds when $\beta \geq \max_i |\sigma_i(s)|$. Therefore performing this reduction results in at most an additive loss of $\delta + 10\epsilon m$ in success probability. Note that we can further re-randomise to obtain a uniform secret distribution i.e. obtain instances of the S-$\mathsf{RLWE}^{(R_{n/2})}_{m,q^2,2\mathbf{H}' \cdot D_{\boldsymbol{\alpha}'}}(U(R^\vee_{n/2,q^2}))$ problem.

Next, conditioned on the event that $\max_i |\sigma_i(s)| \leq B$ we can use Lemma 14 (and

the data processing inequality) to show that

$$R_2\left(\left(2\mathbf{H}'\cdot D_{\sqrt{\alpha^2+r^2\beta^2}}\right)^m \| \left(2\mathbf{H}'\cdot D_{\boldsymbol{\alpha}}\right)^m\right) \leq R_2\left(\left(D_{\sqrt{\alpha^2+r^2\beta^2}}\right)^m \| \left(D_{\boldsymbol{\alpha}}\right)^m\right)$$
$$\leq \left(1+\frac{\sigma_1^4}{\beta^4}\right)^{mn/2} \leq \left(1+\frac{1}{mn}\right)^{mn/2}.$$

The above is upper bounded by 2. By the data-processing inequality and the probability preservation property of Renyi divergences, an algorithm solving

$$S\text{-}\mathsf{RLWE}^{(R_{n/2})}_{m,q^2,2\mathbf{H}'\cdot D_{\boldsymbol{\alpha}'}}(U(R^\vee_{n/2,q^2}))$$

with probability p solves

$$S\text{-}\mathsf{RLWE}^{(R_{n/2})}_{m,q^2,2\mathbf{H}'\cdot D_{\sqrt{\alpha^2+r^2\beta^2}}}(U(R^\vee_{n/2,q^2}))$$

with probability at least $p^2/2$. To complete the proof for $t = 1/4$, we note that in the coefficient embedding, the error distribution $2\mathbf{H}'\cdot D_{\sqrt{\alpha^2+r^2\beta^2}}$ is equal to $2\mathbf{H}\cdot D_{\sqrt{\alpha^2+r^2\beta^2}/\sqrt{n}}$ where $\mathbf{H}$ is defined in Equation (4.16). It should be clear that this is a spherical Gaussian distribution in the coefficient embedding with parameter $2\sqrt{\alpha^2+r^2\beta^2}/\sqrt{n}$ as $\mathbf{H}$ effectively deletes entries. Mapping back to the canonical embedding in dimension $n/2$ via the appropriate scaled isometry, we find that the error distribution $2\mathbf{H}'\cdot D_{\sqrt{\alpha^2+r^2\beta^2}}$ is spherical with Gaussian parameter $\sqrt{2(\alpha^2+r^2\beta^2)}$. □

We now discuss the consequences of this corollary. Taking normal form RLWE i.e. setting $\chi = D_{R_n^\vee,\alpha q}$ allows us to set $B = \alpha q n^c$ for any constant $c > 0$ whilst keeping δ negligible by Lemma 13. Further, we can set $\epsilon = n^{-\log(n)}$ (which is a negligible function), $m = O(1)$, $n = \mathsf{poly}(\lambda)$ and ignore logarithmic factors in this discussion. In doing so, the above corollary for $t = 1/4$ says that if we can solve RLWE in dimension $n/2$, modulus q^2 and error rate $\alpha\cdot n^{3/4+c}$ with non-negligible probability in polynomial time, then we can also solve RLWE with dimension n, modulus q and error rate α is polynomial time with non-negligible probability. This result is improved upon in the next section.

4.5 Related/Subsequent Work

The work of [139] revisited the problem over reducing MLWE to RLWE. In particular, the search to search variant of Theorem 5 (with slightly different parameters)

is extended to obtain decision to decision reductions.

There is also literature relating other algebraic variants of LWE via reductions. The algebraic variants considered include, Order-LWE (OLWE) [17], Middle-Product-LWE (MPLWE) [121] and Polynomial-LWE (PLWE) [122]. Very informally, PLWE is the analogue of RLWE over polynomial rings $\mathbb{Z}[X]/f(X)$ that are not rings of integers of algebraic fields, OLWE is the analogue of RLWE over rings that are "orders" of algebraic number fields, and MPLWE is concerned with LWE over rings where the multiplication is replaced with a so-called "middle-product". It is shown in [121] that MPLWE is at least as hard as PLWE for any polynomial f coming from a large class of polynomials. The blow-up in error-rate in this reduction is related to properties of the polynomial f. In turn, it was shown in [122] that the PLWE problem with a large class of polynomials is at least as hard as the RLWE problem. The growth in error rate for the class of polynomials considered is at least $n^{5/2}$ for this reduction. Another method of proving hardness of PLWE was presented in [17] via a reduction from OLWE (which was also shown to have a reduction from worst-case lattice problems).

Very recent work [109] unifies the above works by using an abstract framework and a collection of concise reductions encompassing many of the discussed hardness results. An advantage of this unification is that the growth in error rates are simpler, more concrete and smaller than previous work. It turns out that we can directly strengthen our search to search RLWE reduction by applying a result in the abstract framework laid out in [109]. In doing so, we can actually obtain a *decision to decision* RLWE reduction with a smaller polynomial blow-up in error rate than the search to search reduction in Section 4.4. We give details of this below.

4.5.1 An Improved RLWE to RLWE Reduction Result

The definition of OLWE is entirely analogous to the definition of RLWE. In particular, the decision variant with respect to an order $\mathcal{O}$ is parametrised by a modulus q, a secret distribution χ over $\mathcal{O}^\vee$ and an error distribution ψ over $K_\mathbb{R}$. Let $\mathcal{O}_q := \mathcal{O}/q\mathcal{O}$. The decisional OLWE problem $O\mathsf{LWE}^{\mathcal{O}}_{q,\psi}(\chi)$ asks to distinguish between uniform samples over $\mathcal{O}_q \times K_\mathbb{R}/\mathcal{O}^\vee$ and samples of the form $\left(a, b = \frac{1}{q}a \cdot s + e \bmod \mathcal{O}^\vee\right)$ where

$a \leftarrow U(\mathcal{O}_q)$, $s \leftarrow \chi$ and $e \leftarrow \psi$. The below is a result from [109] phrased using the convention that $b = \frac{1}{q}a \cdot s + e$ rather than $b = a \cdot s + e$ as was the case in [109]. We can also define the module variant of OLWE in rank d denoted $\mathsf{OLWE}^{\mathcal{O}^d}_{q,\psi}(\chi^d)$ by considering the order variant of the normal MLWE problem with secret distribution χ^d over $(\mathcal{O}^\vee)^d$. Consider any two rings R, R' such that R' is a finite-rank free R-module and any $x \in R'$. In the below, we define $\mathrm{Tr}_{R'/R}(x)$ to be the trace of the R-linear transformation acting on R' (viewed as a module over R) corresponding to multiplication by x. It is also useful to note that if a number field $K \cong \mathbb{Q}[X]/f(X)$, then we can view $K_\mathbb{R}$ as the *ring* $\mathbb{R}[X]/f(X)$.

Theorem 7 (Theorem 6.1 [109])**.** *Let K'/K be a number field extension; $\mathcal{O}$ be an order of K; $\mathcal{O}'$ be an order of K' that is a rank-d free $\mathcal{O}$-module with known basis $\vec{b} = (b_1, \ldots, b_d) \in (K')^d$; ψ' be a distribution over $K_\mathbb{R}$; and q be a positive integer. Then there is an efficient deterministic reduction preserving the number of samples from $\mathsf{OLWE}^{\mathcal{O}'}_{q,\psi'}(\chi')$ to $\mathsf{OLWE}^{\mathcal{O}^d}_{q,\psi}(\chi)$ where $\psi = \mathrm{Tr}_{K'_\mathbb{R}/K_\mathbb{R}}(\psi')$ and $\chi = \mathrm{Tr}_{K'/K}(\chi' \cdot \vec{b})$.*

Using the notation from Section 4.4, we get the following corollary.

Corollary 7. *For any $i \in \{1, 2, \ldots, \log_2 n\}$ for power-of-two n, there exists an efficient reduction from $\mathsf{RLWE}^{R_n}_{m,q,D_\alpha}(D_{R_n^\vee,\alpha q})$ to $\mathsf{MLWE}^{R_{n/2^i}^{2^i}}_{m,q,D_{2^i\alpha}}(D_{R_{n/2}^\vee,2^i\alpha q})$.*

Proof. We instantiate Theorem 7 by setting ξ to be a primitive $(2n)^{th}$ root of unity for power-of-two n. Then we set $K' = \mathbb{Q}(\xi)$ and $K = \mathbb{Q}(\xi^{2^i})$ to be cyclotomic fields. The rings of integers $R' := \mathbb{Z}(\xi) = R_n$ and $R := \mathbb{Z}(\xi^{2^i}) = R_{n/2^i}$ are cyclotomic rings of dimension n and $n/2^i$ respectively. Since rings of integers are orders, we may set $\mathcal{O}' = R'$ and $\mathcal{O} = R$. It is easy to see that R' is a rank 2^i R-module with basis $\vec{b} = (1, \xi, \ldots, \xi^{2^i-1})$. Finally, we show that $\mathrm{Tr}_{K'/K}(D_{R_n^\vee,\alpha q} \cdot \vec{b}) = D_{R_{n/2^i}^\vee,2^i\alpha q}$ and $\mathrm{Tr}_{K'_\mathbb{R}/K_\mathbb{R}}(D_\alpha) = D_{2^i\alpha}$. As the simplest example, take $i = 1$. It is not hard to see that for any $e = \sum_{j=0}^{n-1} e_j\xi^j \in K'$, the linear map representing multiplication by e in the K-module basis $(1, \xi)$ is given by the 2×2 matrix

$$\begin{bmatrix} e_0 + e_2\xi^2 + \cdots + e_{n-2}\xi^{n-2} & e_1\xi^2 + e_3\xi^4 + \cdots + e_{n-1}\xi^n \\ e_1 + e_3\xi^2 + \cdots + e_{n-1}\xi^{n-1} & e_0 + e_2\xi^2 + \cdots + e_{n-2}\xi^{n-2} \end{bmatrix}.$$

Therefore $\mathrm{Tr}_{K'/K}(e) = 2 \cdot (e_0 + e_2\xi^2 + \cdots + e_{n-2}\xi^{n-2})$. In a similar fashion, one may see that $\mathrm{Tr}_{K'/K}(e\xi) = 2 \cdot (-e_{n-1} + e_1\xi^2 + \cdots + e_{n-3}\xi^{n-2})$. It is straight-forward to show that $\mathrm{Tr}_{K'_\mathbb{R}/K_\mathbb{R}}$ takes exactly the same form. It can now be seen that if

$\psi' = D_\alpha$, then $\psi = D_{2\alpha}$ and if $\chi' = D_{(R')^\vee,\alpha q}$, then $\chi = (D_{R^\vee,2\alpha q})^2$. This analysis can straight-forwardly be generalised to consider the remaining values of i. □

We can now compose Corollary 7 with Corollary 3 in the context of power-of-two cyclotomic rings and use the two-step reduction below:

$$\mathsf{RLWE}^{R_n}_{m,q,D_\alpha}(D_{R_n^\vee,\alpha q}) \xrightarrow{\text{Cor.7}} \mathsf{MLWE}^{R_{n/2^i}^{2^i}}_{m,q,D_{2^i\alpha}}(D_{R_{n/2^i}^\vee,2^i\alpha q}) \xrightarrow{\text{Cor.3}} \mathsf{RLWE}^{R_{n/2^i}}_{m,q^{2^i},\Psi_{\leq 2^i\alpha'}}.$$

Importantly, composing the corollaries yields a decision to decision reduction, so the result here is stronger than that given in Section 4.4. In addition, reducing from dimension n to $n/2$ (i.e. taking $i = 1$) the error rate grows from α to $n^{c+1/2} \cdot \alpha$ for any constant $c > 0$ in this two step reduction (ignoring constant/logarithmic factors and choosing parameters such that losses in advantage are negligible). This growth factor is roughly $n^{1/4}$ smaller than the one given in Section 4.4. Another advantage of the two-step reduction is that we can reduce to any power of two dimension $n' < n$. Using normal form secrets, the error rate grows roughly by a factor of $\left(\frac{n}{n'}\right)^{3/2} \cdot (n')^{c+1/2} = \frac{n^{3/2}}{(n')^{1-c}}$. Informally, this shows the hardness of small power-of-two dimensional RLWE over cyclotomic rings, provided that the modulus is sufficiently large.

Chapter 5

A Lattice-Based VOPRF

Contents

5.1 Chapter Synopsis

This chapter is concerned with constructing a fully post-quantum secure round-optimal verifiable pseudorandom function (VOPRF) in the quantum random oracle

model. To our knowledge, this is the first construction of a post-quantum VOPRF. The pseudorandom function that will be obliviously evaluated is the RLWE based PRF from Banerjee and Peikert [18] (referred to as the BP14 PRF from now on). As a reminder, a VOPRF for a pseudorandom function F is a protocol between a server S and a client C. The server holds a key k and the client has an input x. The client C wants the guarantee of obtaining the correct value of $F_k(x)$ without leaking the value of x to S. On the other hand, the server wants to be able to provide the value of $F_k(x)$ without C learning anything about k. For applications of VOPRFs, see the introduction of this thesis. The rest of this synopsis aims to give the *intuition* behind our VOPRF design and security argument.

Intuition/Technical Overview We first informally overview the BP14 PRF in the ring setting. Specifically, for a particular *function* $\boldsymbol{a}^F : \{0,1\}^L \to R_q^{1\times\ell}$ where $R_q := \mathbb{Z}_q[X]/\langle X^n + 1\rangle$ and a particular choice of (p, q), we will design a VOPRF for the BP14 PRF

$$F_k(x) = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k \right\rceil$$

where the key $k \in R_q$ has small coefficients when represented in $\{-q/2, \ldots, q/2\}$. To provide intuition for our VOPRF design, we describe a basic protocol below that serves as a *starting point*. We assume that zero knowledge proofs of each message are implicitly provided to ensure that the protocol is followed. Our first attempt at building a VOPRF is to execute the following steps:

1. The server publishes some commitment to a small key $k \in R_q$.
2. On input x, the client picks *invertible* $s \in R_q$, small $\boldsymbol{e} \in R_q^{1\times\ell}$ and sends $\boldsymbol{c}_x = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}$.
3. On input small $k \in R_q$, the server sends $\boldsymbol{d}_x = \boldsymbol{c}_x \cdot k + \boldsymbol{e}'$ for small $\boldsymbol{e}' \in R_q^{1\times\ell}$.
4. The client outputs $\boldsymbol{y} = \left\lfloor \frac{p}{q} \cdot \boldsymbol{d}_x \cdot s^{-1} \right\rceil$.

For server security, note that $\boldsymbol{d}_x = \boldsymbol{a}^F(x) \cdot s \cdot k + \boldsymbol{e} \cdot k + \boldsymbol{e}'$. In a security proof, one strategy could be to artificially introduce an additional error term $\boldsymbol{e}_s$ that is distributed identically to $\boldsymbol{e}$. Suppose that we choose $\boldsymbol{e}'$ from a distribution that statistically hides addition of terms $\boldsymbol{e} \cdot k$ and $\boldsymbol{e}_s \cdot s$. Then, from the perspective

of the client, the server might as well have sent $\boldsymbol{d}_x = (\boldsymbol{a}^F(x) \cdot k + \boldsymbol{e}_s) \cdot s + \boldsymbol{e}'$. Picking $\boldsymbol{e}_s$ (and $\boldsymbol{e}$) from an appropriate distribution [18] makes the term in brackets i.e. $\boldsymbol{a}^F(x) \cdot k + \boldsymbol{e}_s$ computationally indistinguishable from uniform random under a RLWE assumption. This implies that the message $\boldsymbol{d}_x$ leaks nothing about the server's key k.

For client security in the first message, we pick s from a valid RLWE secret distribution and $\boldsymbol{e}$ from the same distribution as that of $\boldsymbol{e}_s$. Similarly to the above, this implies that $\boldsymbol{c}_x = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}$ is indistinguishable from uniform and doesn't leak information on s. Finally, we must show that the client does indeed recover $F_k(x)$ as its output $\boldsymbol{y}$. For correctness, we *would like to say* that

$$\left\lfloor \frac{p}{q} \cdot \boldsymbol{d}_x \cdot s^{-1} \right\rceil = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k + \frac{p}{q}(\boldsymbol{e} \cdot k + \boldsymbol{e}') \cdot s^{-1} \right\rceil = \left\lfloor \frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k \right\rceil.$$

Thus, we guarantee correctness if all coefficients of $\frac{p}{q} \cdot \boldsymbol{a}^F(x) \cdot k$ are at least

$$\left\| \frac{p}{q} \left(\boldsymbol{e} \cdot k + \boldsymbol{e}' \right) \cdot s^{-1} \right\|_\infty$$

away from $\mathbb{Z} + \frac{1}{2}$. It turns out that if all coefficients of s^{-1} are small, then this condition is satisfied with extremely high probability due to the 1-dimensional short integer solution (1D-SIS) assumption (see Definition 23). The form of $\boldsymbol{a}^F(x)$ is crucial to the connection with the 1D-SIS problem. In particular, we rely on the fact that we can decompose $\boldsymbol{a}^F(x)$ as $\boldsymbol{a}'_1 \cdot \boldsymbol{a}'_2$ where $\boldsymbol{a}'_1 \in R_q^{1 \times \ell}$ is uniform random and $\boldsymbol{a}'_2 \in R_q^{\ell \times \ell}$ has entries that are polynomials with *binary* coefficients.

Unfortunately, this simplified protocol cannot quite be realised using standard RLWE secret distributions. The problem is that (to our knowledge) there is no standard RLWE secret distribution where samples from the distribution are guaranteed to have *small inverses* in R_q. To overcome this issue, we apply a technique for sampling "full" NTRU keys [69, 116]. Firstly, we sample small ring elements s and t from a Gaussian distribution. Secondly, we use the extended GCD algorithm – in conjunction with Babai's rounding algorithm – to recover small u and v, such that $u \cdot s + v \cdot t = 1 \bmod R_q$. To adapt the basic protocol to our actual protocol, the client sends $\boldsymbol{c}_x^1 = \boldsymbol{a}^F(x) \cdot s + \boldsymbol{e}_1$, $\boldsymbol{c}_x^2 = \boldsymbol{a}^F(x) \cdot t + \boldsymbol{e}_2$ and receives back

$$\boldsymbol{d}_x^1 = \boldsymbol{c}_x^1 \cdot k + \boldsymbol{e}'_1, \quad \boldsymbol{d}_x^2 = \boldsymbol{c}_x^2 \cdot k + \boldsymbol{e}'_2.$$

The final output of the client is then calculated as $\left\lfloor \frac{p}{q} \left(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2 \right) \right\rceil$. In addition, the real protocol incorporates zero knowledge proofs of knowledge (that we show are

instantiable based on adaptations of Stern's protocol) to prevent malicious parties deviating from the protocol description. Although we focus on Stern's protocol for simplicity, we may also instantiate the proofs using the recent work of Beullens [24] to improve efficiency.

Ultimately, the security of our VOPRF construction using the Stern-style zero knowledge proofs holds in the QROM and relies on the hardness of RLWE and 1D-SIS which are both at least as hard as certain lattice problems using appropriate parameters. We discuss asymptotic parameter settings for which our protocol relies directly on assumed hard lattice problems in Section 5.4. In summary, we obtain a round-optimal lattice-based VOPRF in the QROM secure against malicious adversaries with polynomially sized messages.

Road Map We begin with chapter preliminaries in Section 5.2 that include the formal definition of a VOPRF that we use. This is followed by the actual VOPRF construction in Section 5.3 and security proof in Section 5.4. Details on how to instantiate the zero knowledge components of our construction in the QROM are given in Section 5.6.

5.2 Chapter Preliminaries

We begin by setting some RLWE/error distribution notation. This is followed by some results implying that with all but negligible probability, two RLWE samples is enough to ensure a unique secret. We then introduce the BP14 PRF formally along with a formal security definition of a VOPRF.

Important Notation: We use power of two cyclotomic rings $R = \mathbb{Z}[X]/\langle X^n + 1\rangle$ throughout this section. We will be using the practical decisional version of the RLWE problem given in Definition 20. To ease notation, we use χ_σ to denote the Gaussian distribution $D_{\mathbb{Z},\sigma}$. Furthermore, $R(\chi_\sigma)$ will denote the distribution over elements of R where each coefficient is distributed as χ_σ. In addition, we denote $\mathsf{RLWE}^R_{m,q,R(\chi_\sigma)}(R(\chi_\sigma))$ as $\mathsf{RLWE}_{m,n,q,\sigma}$ and omit m when we do not wish to specify

a particular number of samples. We also use coefficient embeddings in this section to define the norms of polynomials. For any positive real number $B \geq 0$, $R_{\leq B}$ is the set of degree $n-1$ polynomials whose coefficients are integers with absolute value at most B. We denote matrices whose entries are integers using non-bold capital letters and vectors whose entries are integers using an over-arrow. This is to differentiate them from matrices/vectors whose entries are ring elements. This convention is particularly useful in Section 5.6.

5.2.1 RLWE with Two (Invertible) Samples is Well-Defined

Note that we can apply the following claim to the Gaussian error distribution $R(\chi_\sigma)$ by setting $\bar{\sigma} = \sigma\sqrt{n}$ and applying Corollary 1.

Claim 4. *Let $|R_q^\times|$ denote the number of invertible elements in R_q and χ be a distribution that outputs polynomials with infinity norm $\leq \bar{\sigma}$. The S-$\mathsf{RLWE}_{2,q,n,\chi}$ problem restricting to invertible ring elements a_1, a_2 has a unique solution with probability at least $1 - \frac{(2\bar{\sigma})^{2n}}{|R_q^\times|^2} \cdot |R_q|$.*

Proof. Suppose we are given a RLWE challenge $b_1 = a_1 \cdot s + e_1, b_2 = a_2 \cdot s + e_2$. Then we can write any $s' \in R_q$ as $s' = s + u$ for some $u \in R_q$. Then it should be clear that

$$b_i - a_i \cdot s' = a_i \cdot u + e_i.$$

Since the error distribution χ outputs polynomials with infinity norm at most $\bar{\sigma}$, s' can only possibly be a valid solution when $|a_i \cdot u \bmod q|_\infty \leq 2\bar{\sigma}$. This means that our RLWE instance has a unique solution if there is no non-zero $u \in R_q$ such that $|a_i \cdot u|_\infty \leq 2\bar{\sigma}$.

We now upper bound the probability over uniformly chosen invertible elements a_1, a_2 that there exists a non-zero $u \in R_q$ resulting in the event that $|a_i \cdot u \bmod q|_\infty \leq \bar{\sigma}$ for $i = 1$ and 2. We refer to this event as E. For any fixed $u = \bar{u} \neq 0$, we have that

$$\Pr_{a_1, a_2 \leftarrow R_q^\times}[|a_1 \cdot \bar{u} \bmod q|_\infty \leq \bar{\sigma} \wedge |a_2 \cdot \bar{u} \bmod q|_\infty \leq \bar{\sigma}] = \left(\frac{(2\bar{\sigma})^n}{|R_q^\times|}\right)^2.$$

Using a union bound over all non zero choices of $\bar{u}$, we get that

$$\Pr_{a_1, a_2 \leftarrow R_q^\times}[E] \leq \frac{(2\bar{\sigma})^{2n}}{|R_q^\times|^2} \cdot |R_q|,$$

which is an upper bound for the corresponding RLWE instance having a non-unique solution. □

In order to qualitatively interpret this lemma, we next analyse how many elements of R_q are invertible for a particular choice of q that we will be using in our construction.

Claim 5. *Let $p_1 > \cdots > p_m$. Further, let $q = \prod_{i=1}^{m} p_i$. Then $\frac{|R_q^\times|}{|R_q|} \prod_{i=1}^{m} \left(1 - \frac{1}{p_i}\right)^n$ and $|R_q^\times| \geq \prod_{i=1}^{m} (p_i - 1)^n$.*

Proof. We use the chinese remainder theorem (CRT) isomorphism

$$R_q \cong R_{p_1} \times \cdots \times R_{p_m}.$$

Now consider R_{p_i} for some fixed i and suppose that $x^n + 1 = \prod_{j=1}^{n_i} r_j(x) \bmod p_i$ where $r_j(x)$ is an irreducible polynomial of degree d_j. Then by applying the CRT to R_{p_i} and the fact that $\mathbb{Z}_{p_i}(x)/\langle r_j(x)\rangle$ are fields, the number of invertible elements in R_{p_i} is $\prod_{j=1}^{n_i} \left(p_i^{d_j} - 1\right) \geq \prod_{j=1}^{n_i} (p_i - 1)^{d_j} \geq (p_i - 1)^n$.

It follows that there are at least $\prod_{i=1}^{m} (p_i - 1)^n$ invertible elements in $R_{p_1} \times \cdots \times R_{p_m}$ which means that the probability of a uniformly sampled element of R_q being invertible is at least $\prod_{i=1}^{m} \left(1 - \frac{1}{p_i}\right)^n$. □

To summarise, we will take $\bar{\sigma} = \sigma\sqrt{n}$ for $\sigma = \mathsf{poly}(\lambda)$ and $q = \prod_{i=1}^{m} p_i$ where p_1 is superpolynomial in λ. Note that Claims 4 and 5 imply that two RLWE samples restricted to invertible elements has a unique solution with probability at least

$$1 - (2\sigma\sqrt{n})^{2n} \cdot \prod_{i=1}^{m} \left(\frac{p_i}{(p_i - 1)^2}\right)^n \geq 1 - (2\sigma\sqrt{n})^{2n} \cdot \left(\frac{p_1}{(p_1 - 1)^2}\right)^{mn} \tag{5.1}$$

which is negligibly close to 1 when mn is polynomial in λ. In addition, the quantity $|R_q^\times|/|R_q|$ is also negligibly close to 1 implying that sampling two standard RLWE samples results in invertible a_1 and a_2 with all but negligible probability. Therefore, we can say that standard S-RLWE with two samples has a unique solution with all but negligible probability for the parameter setting discussed here.

5.2.2 The BP14 PRF

We will use an instantiation of the lattice PRF from [18]. Below, we present relevant definitions/results, all of which are particular cases of definitions/results from [18]. We set $\ell = \lceil \log_2 q \rceil$ throughout. The construction from [18] makes use of *gadget matrices* used in many previous works [108, 18, 33, 59].

Gadgets $\mathbf{G}, G^{-1}$ Define $\mathbf{G} : R_q^{\ell \times \ell} \to R_q^{1 \times \ell}$ to be the linear operation corresponding to left multiplication by $(1, 2, \ldots, 2^{\ell-1})$. Further, define $G^{-1} : R_q^{1 \times \ell} \to R_q^{\ell \times \ell}$ to be the (non-linear) bit decomposition operation that essentially inverts $\mathbf{G}$ i.e. the i^{th} column of $G^{-1}(\boldsymbol{a})$ is the bit decomposition of $a_i \in R_q$ into binary polynomials. To emphasise the non-linearity, we do not use bold font for G^{-1} as it cannot be expressed by a matrix.

For $x \in \mathbb{Z}_q$, define $\lfloor x \rceil_p := \left\lfloor \frac{p}{q} \cdot x \right\rceil$ and extend this definition coefficient-wise for polynomials/vectors of polynomials. The PRF from [18] that we focus on is defined as $F_k(x) = \lfloor \boldsymbol{a}_x \cdot k \rceil_p$ for $\boldsymbol{a}_x \in R_q^{1 \times \ell}$ as defined below. Throughout, we reserve L to denote the bit-length of the PRF input.

Definition 32. *Fix some $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1 \times \ell}$. For any $x = (x_1, \ldots, x_L) \in \{0,1\}^L$. We define $\boldsymbol{a}_x \in R_q^{1 \times \ell}$ as*

$$\boldsymbol{a}_x := \boldsymbol{a}_{x_1} \cdot G^{-1}\left(\boldsymbol{a}_{x_2} \cdot G^{-1}\left(\boldsymbol{a}_{x_3} \cdot G^{-1}\left(\ldots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\right)\right)\right) \in R_q^{1 \times \ell}.$$

The pseudorandomness of this construction follows from the RLWE assumption.

Theorem 8 ([18])**.** *Let $R = \mathbb{Z}[X]/\langle X^n + 1\rangle$ for power-of-two n and sample $k \leftarrow R(\chi_\sigma)$. If $q \gg p \cdot \sigma \cdot \sqrt{L} \cdot (n \cdot \ell)$, then the function $F_k(x) = \lfloor \boldsymbol{a}_x \cdot k \rceil_p$ is a PRF under the $\mathsf{RLWE}_{n,q,\sigma}$ assumption.*

When we eventually prove security of our VOPRF, it will be useful to define a special error distribution such that $\boldsymbol{a}_x \cdot k + \boldsymbol{e}$ remains indistinguishable from uniform (under a RLWE assumption) when $\boldsymbol{e}$ is sampled from this special error distribution. To this end, we introduce the distributions $\mathcal{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1, x, \sigma}$ followed by a lemma that is implicit in the pseudorandomness of the PRF from [18].

Definition 33. *For $\boldsymbol{a}_0, \boldsymbol{a}_1 \in R_q^{1\times\ell}$, define*

$$\boldsymbol{a}_{x\setminus i} := G^{-1}\left(\boldsymbol{a}_{x_{i+1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_{i+2}} \cdot G^{-1}\left(\ldots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\ldots\right)\right)\right) \in R_q^{\ell\times\ell}.$$

Furthermore, let $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ be the distribution that is sampled by choosing $\boldsymbol{e}_i \leftarrow R(\chi_\sigma)^{1\times\ell}$ for $i = 1,\ldots,L$ and outputting

$$\boldsymbol{e} = \sum_{i=1}^{L-1} \boldsymbol{e}_i \cdot a_{x\setminus i} + \boldsymbol{e}_L.$$

Lemma 15 (Implicit in [18])**.** *If $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}, \boldsymbol{e} \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ and $s \leftarrow R(\chi_\sigma)$, then for any fixed $x \in \{0,1\}^L$,*

$$(\boldsymbol{a}_0,\ \boldsymbol{a}_1,\ \boldsymbol{a}_x \cdot s + \boldsymbol{e})$$

is indistinguishable from uniform random by the $\mathsf{RLWE}_{q,n,\sigma}$ assumption.

In addition to introducing $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$, it will be useful to write down an upper bound on the infinity norm on errors drawn from this distribution. The following lemma follows from the fact that for $y \leftarrow \chi_\sigma$, $\|y\|_\infty \leq \sigma\sqrt{n}$ with all but negligible probability by Corollary 1. In fact, we could use the result that $\|y\|_\infty \leq \sigma n^{c'}$ with probability at least $1 - c \cdot \exp(-\pi n^{2c'})$ for any constant $c' > 0$ and some universal constant c (Corollary 1) to reduce the upper bound, but we choose not to for simplicity.

Lemma 16 (Bound on errors)**.** *Let $x \in \{0,1\}^L$, $\ell = \lceil \log_2 q \rceil$ and $n = \mathsf{poly}(\lambda)$. Samples from $\mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ have infinity norm at most $L \cdot \ell \cdot \sigma \cdot n^{3/2}$ with all but negligible probability.*

5.2.3 Verifiable Oblivious Pseudorandom Functions

Recall that the main goal of our work is to build a verifiable oblivious pseudorandom function (VOPRF). A VOPRF is a protocol between two parties: a server S and a client C, that in some sense realises the ideal functionality in Figure 5.1. The functionality consists of two phases, the initialisation phase and the query phase. In the event that the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ receives an input k from party S (i.e. the server) during the initialisation phase, it stores the key for use during the query phase. This models a server in a real protocol committing to a PRF key k. Next comes the query phase, where the client C sends some value x to $\mathcal{F}_{\mathsf{VOPRF}}$. Once this

value x has been received, the server S either sends the functionality an instruction to abort or to deliver the value $y = F_k(x)$ to C. Finally, the functionality carries out this instruction. Importantly, (assuming that no abort is triggered) the client has the guarantee that its output is indeed $F_k(x)$ i.e. the output of the client is *verifiably* correct when interacting with $\mathcal{F}_{\mathsf{VOPRF}}$. Importantly, when interacting via the ideal functionality, a client learns only the final evaluation and the server learns nothing about the client's input.

The Ideal Functionality $\mathcal{F}_{\mathsf{VOPRF}}$

This is a two party functionality between a server S and a client C. We assume there is a fixed PRF function defined by $F_k(x)$.

Init: On input of init from both parties the functionality waits for an input k from party S. If S returns abort then the functionality aborts. Otherwise the functionality stores the value k.

Query: On input of (query, x) from the client C, if $x \neq \perp$ then functionality waits for an input from party S. If S returns $\mathsf{deliver}$ then the functionality sends $y = F_k(x)$ to party C. If S returns abort then the functionality aborts.

Figure 5.1: The Ideal Functionality $\mathcal{F}_{\mathsf{VOPRF}}$

We now describe the distributions that arise in the security requirement. We consider malicious adversaries throughout that behave arbitrarily. We begin with the distributions of interest when a server has been corrupted. First, we consider a "real" world protocol Π between $\mathsf{C}(x)$ and $\mathsf{S}(k)$ along with an adversary $\mathcal{A}$. We denote $\mathsf{real}_{\Pi,\mathcal{A},\mathsf{S}}(x, k, 1^\lambda)$ to be the joint output distribution of $\mathcal{A}(k)$ when corrupting $\mathsf{S}(k)$ and $\mathsf{C}(x)$ where $\mathsf{C}(x)$ behaves as specified by Π. In this setting, $\mathcal{A}$ interacts directly with C. Now we introduce a simulator denoted Sim that lives in the "ideal" world. Specifically, still assuming $\mathcal{A}$ corrupts a server, Sim interacts with $\mathcal{A}$ on one hand and with $\mathsf{C}(x)$ via $\mathcal{F}_{\mathsf{VOPRF}}$ on the other hand. Considering this setting, for any client/server input pair (x, k), we define $\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathsf{S}}(x, k, 1^\lambda)$ to be the joint output distribution of $\mathcal{A}(k)$ and the honest client $\mathsf{C}(x)$ when $\mathcal{A}(k)$ interacts via Sim. Informally, one may interpret Sim as an attacker-in-the-middle between $\mathcal{A}$ and the outside world that interacts with $\mathcal{F}_{\mathsf{VOPRF}}$ external to the view of $\mathcal{A}$. Security will argue that whatever $\mathcal{A}$ can learn/affect in the real protocol can be emulated via Sim in the ideal world setting.

Next, we describe the distributions of interest when a client has been corrupted by an adversary $\mathcal{A}$. We let $\mathcal{K}$ denote the key distribution under which PRF security

of F holds. First, consider a "real" world case where $\mathcal{A}$ corrupts $\mathsf{C}(x)$ and directly interacts with honest $\mathsf{S}(k)$ which follows the specification of protocol Π. In this case, we use $\mathsf{real}_{\Pi,\mathcal{A},\mathsf{C}}(x,\mathcal{K},1^\lambda)$ to denote the joint output distribution of $\mathcal{A}(x)$ and $\mathsf{S}(k)$ where $k \leftarrow \mathcal{K}$. Now consider an alternative "ideal" world case where we introduce a simulator Sim interacting with $\mathcal{A}$ on one hand and with $\mathsf{S}(x)$ via $\mathcal{F}_{\mathsf{VOPRF}}$ on the other hand. Once again, one may wish to interpret the simulator as an attacker-in-the-middle interacting with $\mathcal{F}_{\mathsf{VOPRF}}$ external to the view of $\mathcal{A}$. In this alternative case, we denote the joint output distribution of $\mathcal{A}(x)$ and $\mathsf{S}(k)$ where $\mathcal{A}$ interacts via Sim and $k \leftarrow \mathcal{K}$ as $\mathsf{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathsf{C}}(x,\mathcal{K},1^\lambda)$.

Finally, for protocol Π, let $\mathsf{output}(\Pi,x,k)$ denote the output distribution of a *client* with input x running protocol Π with a server whose input key is k. Using the notation established above, we are ready to present our definition of a VOPRF.

Definition 34. *A protocol Π is a verifiable oblivious pseudorandom function for a PRF F if all of the following hold:*

1. ***Correctness:*** *For every pair of inputs (x,k),*

$$\Pr[\mathsf{output}(\Pi,x,k) \neq F_k(x)] \leq \mathsf{negl}(\lambda)\,.$$

2. ***Malicious server security:*** *For any PPT adversary $\mathcal{A}$ corrupting a server, there exists a PPT simulator Sim such that for every pair of inputs (x,k):*

$$\mathit{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathsf{S}}(x,k,1^\lambda) \approx_c \mathit{real}_{\Pi,\mathcal{A},\mathsf{S}}(x,k,1^\lambda).$$

3. ***Average-case malicious client security:*** *For any PPT adversary $\mathcal{A}$ corrupting a client, there exists a PPT simulator Sim such that for all client inputs x:*

   - $\mathit{ideal}_{\mathcal{F}_{\mathsf{VOPRF}},\mathsf{Sim},\mathcal{A},\mathsf{C}}(x,\mathcal{K},1^\lambda) \approx_c \mathit{real}_{\Pi,\mathcal{A},\mathsf{C}}(x,\mathcal{K},1^\lambda)$.
   - *If $\mathcal{A}$ correctly outputs $F_k(x)$ with all but negligible probability over the choice $k \leftarrow \mathcal{K}$ when interacting directly with $\mathsf{S}(k)$ using protocol Π, then $\mathcal{A}$ also outputs $F_k(x)$ with all but negligible probability when interacting via Sim.*

We now discuss this definition. Note that the correctness and malicious server security requirements are the standard ones used in MPC (for the standard security

notions, see [87]). Therefore, we restrict this discussion to the condition that we call average case malicious client security. The motivation for this non-standard property is that an honest server will always sample a key from distribution $\mathcal{K}$ as it wishes to provide pseudorandom function evaluations. In particular, PRF security holds with respect to this key distribution $\mathcal{K}$. Therefore, it makes sense to ask what a malicious client may learn/affect only in the case where $k \leftarrow \mathcal{K}$ which leads to the first point of our average case malicious client security requirement. The second point of the requirement captures the fact that adversaries may have access to an oracle that checks whether the PRF was evaluated correctly or not. Suppose that we give the adversary $\mathcal{A}$ access to an oracle which can check an input/output pair to the PRF is valid or not. Then $\mathcal{A}$ should not be able to distinguish whether it is interacting with a real server S or a simulation Sim. Note that our proof structure relies heavily on our alternative malicious client security definition. In particular, the definition above allows us to argue over the entropy of secret keys when making indistinguishability claims.

5.3 A VOPRF Construction From Lattices

In this section, we provide a construction emulating a Diffie-Hellman style blinding construction $g^a = ((g^r)^a)^{1/r}$. In what follows, we will initially ignore the zero-knowledge proofs of knowledge establishing that all computations are performed honestly. A detailed description of the protocol is in Figure 5.2 but the main high-level idea follows.

Recall that we are working with power-of-two cyclotomic rings. We begin by considering a slightly different scenario. In particular, suppose that a *client* wants to obtain $a \cdot k + e \in R_q$ (where e is relatively small) from a server holding a *short* k without revealing $a \in R_q$. One way to achieve this is for the client to sample $s, t, e_0, e_1 \leftarrow R(\chi_\sigma)$. The client could then also sample *short* u, v such that $u \cdot s + v \cdot t = 1 \in R_q$ (we discuss how in Section 5.3.1). The client submits $a \cdot s + e_0$ and $a \cdot t + e_1$ and obtains $(a \cdot s + e_0)k + e'_0$ and $(a \cdot t + e_1)k + e'_1$ from the server where

e_0, e'_0, e_1, e'_1 are small. Finally the client can compute:

$$\begin{aligned} r &= u \cdot ((a \cdot s + e_0) \cdot k + e'_0) + v \cdot ((a \cdot t + e_1)k + e'_1) \\ &= a \cdot (u \cdot s + v \cdot t) \cdot k + u \cdot e_0 \cdot k + u \cdot e'_0 + v \cdot e_1 \cdot k + v \cdot e'_1 \\ &= a \cdot k + u \cdot e_0 \cdot k + u \cdot e'_0 + v \cdot e_1 \cdot k + v \cdot e'_1 \\ &\approx a \cdot k. \end{aligned}$$

As mentioned above, a more detailed and precise formulation of our construction is given in Figure 5.2. In addition to the query phase informally outlined above, there is a set-up phase where system-wide parameters are sampled, and an initialisation phase where the server commits to its secret key. In the protocol description, P_i and V_i are prover and verifier algorithms for three different ZKPoK systems indexed by $i \in \{0, 1, 2\}$. Information on the languages underlying these zero knowledge proof systems is given in Section 5.3.2.

5.3.1 Sampling s, t, u, v

To compute tuples s, t, u, v such that $u \cdot s + v \cdot t = 1$ and all elements are short, we may use known techniques for sampling "full" NTRU private keys [69, 116] on input of $(s, t) \in R^2$. From now on we use: $\mathrm{Res}(\cdot, \cdot)$ to refer to the computation of the resultant of two polynomials; $\mathrm{xgcd}(\cdot, \cdot)$ to refer to the computation of the extended GCD of two integers; and $s^\star$ to refer to the conjugate of s in R. In particular, $\mathsf{fullNTRU}(s, t)$ runs the following steps.

1. Compute $r_s = \mathrm{Res}(s, X^n + 1) \in \mathbb{Z}$ and $u' \in R$ s.t. $u' \cdot s = r_s$
2. Compute $r_t = \mathrm{Res}(t, X^n + 1) \in \mathbb{Z}$ and $v' \in R$ s.t. $v' \cdot t = r_t$
3. Compute $r, u'', v'' = \mathrm{xgcd}(r_s, r_t)$. If $r \neq 1$: abort
4. Set $u = u'' \cdot u' \in R$ and $v = v'' \cdot v' \in R$.
5. Run Babai's inverting and rounding algorithm [14]:
   (a) Compute
   $$r = \left\lfloor \frac{v \cdot s^\star - u \cdot t^\star}{s \cdot s^\star + t \cdot t^\star} \right\rceil.$$

VOPRF construction

SetUp: To set up the various parameters execute the following steps:

- Pick $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$
- crs_0 contains $(a, b) \in R_q^2$
- crs_1 and crs_2 are for proof systems P_1 and P_2 respectively

Init: The initialization procedure is executed by the server S and the client C both with initial input crs_0.

1. The server S executes the following steps
   - $k, e, e' \leftarrow R(\chi_\sigma)$.
   - $c_1 \leftarrow a \cdot k + e \bmod q$.
   - $c_2 \leftarrow b \cdot k + e' \bmod q$.
   - $\pi_0 \leftarrow \mathsf{P}_0(k, e, e' : \mathsf{crs}_0)$.

   and sends (c_1, c_2, π_0) to the client C.
2. On receipt of (c_1, c_2, π_0), the client C executes
   - $b \leftarrow \mathsf{V}_0(\mathsf{crs}_0, c_1, c_2, \pi_0)$.
   - Output abort if $b = 0$; otherwise store (c_1, c_2).

Query: This is a two message protocol between the client and the server, with the client going first.

1. On input of $(x \in \{0,1\}^L, \mathsf{crs}_1, \mathsf{crs}_2)$ the client C executes the following steps
   - $s, t \leftarrow R(\chi_\sigma)$.
   - If $\mathsf{fullNTRU}(s, t)$ aborts: go back to previous step
     else: $(u, v) \leftarrow \mathsf{fullNTRU}(s, t)$.
   - $\boldsymbol{a}_x = \boldsymbol{a}_{x_1} \cdot G^{-1}\left(\ldots\left(\boldsymbol{a}_{x_{L-1}} \cdot G^{-1}\left(\boldsymbol{a}_{x_L}\right)\right)\ldots\right) \bmod q$.
   - $\boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1, x, \sigma}$.
   - $\boldsymbol{c}_x^1 \leftarrow \boldsymbol{a}_x \cdot s + \boldsymbol{e}_1 \bmod q$.
   - $\boldsymbol{c}_x^2 \leftarrow \boldsymbol{a}_x \cdot t + \boldsymbol{e}_2 \bmod q$.
   - $\pi_1 \leftarrow \mathsf{P}_1(x, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2 : \mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1)$.

   and sends $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ to the server S.
2. On receipt of $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ the server S executes the following steps
   - $b \leftarrow \mathsf{V}_1(\mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1, \pi_1)$.
   - Output abort if $b = 0$
   - $\boldsymbol{e}_1', \boldsymbol{e}_2' \leftarrow R(\chi_{\sigma'})^{1\times\ell}$.
   - $\boldsymbol{d}_x^1 = \boldsymbol{c}_x^1 \cdot k + \boldsymbol{e}_1' \bmod q$.
   - $\boldsymbol{d}_x^2 = \boldsymbol{c}_x^2 \cdot k + \boldsymbol{e}_2' \bmod q$.
   - $\pi_2 \leftarrow \mathsf{P}_2(k, \boldsymbol{e}_1', \boldsymbol{e}_2', e, e' : \mathsf{crs}_0, \mathsf{crs}_2, c_1, c_2, \boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2)$.

   and sends $(\boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \pi_2)$ to the client C.
3. On receipt of $(\boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \pi_2)$ the client C executes
   - $b \leftarrow \mathsf{V}_2(\mathsf{crs}_0, \mathsf{crs}_2, c_1, c_2, \boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_2)$.
   - Output abort if $b = 0$.
   - $\boldsymbol{y}_x = \left\lfloor u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2 \right\rceil_p$.
   - Output $\boldsymbol{y}_x$.

Figure 5.2: VOPRF construction

(b) Update $(u,\ v) = (u + r \cdot t,\ v - r \cdot s) \in R^2$.

6. Return u, v.

Note that it might be significantly more efficient to implement rational arithmetic using floating point arithmetic as in [116]. Finally, using the same heuristic arguments as in [69, Appendix A], we may expect the norm of u, v to satisfy $\|(u, v)\| \approx \sqrt{n/12} \cdot \|(s, t)\|$. However, for the purposes of our security proofs, we use the upper bound $\|(u, v)\|_\infty \leq n\sigma$ (see below for details).

Suppose we sample $(s, t) \leftarrow R(\chi_\sigma)^2$. Then there is a chance that s and t are not co-prime, causing the above algorithm to abort. However, it is shown in Lemma 4.4 in the full version[1] of [133] that discrete Gaussian s and t will be co-prime with non-negligible probability as long as $\sigma \geq 7 \cdot n^{3/2} \cdot \ln^{3/2}(n)$. Therefore, an algorithm solving RLWE with *two* discrete Gaussian coprime secrets (s, t) with non-negligible advantage would also solve RLWE where the two secrets are sampled independently from Gaussian distributions, with non-negligible probability. This implies that the sampling algorithm above results in a secret distribution for which RLWE is believed to be hard if $(s, t) \leftarrow R(\chi_\sigma)^2$.

5.3.1.1 Upper Bound on $\|u, v\|_\infty$

Babai's rounding technique is a very efficient way of obtaining a candidate solution to CVP. Given a target point $\boldsymbol{t} \in \mathbb{R}^n$ and a lattice Λ with basis $\mathbf{B}$ (which need not be a square matrix), Babai's rounding technique outputs the lattice vector $\boldsymbol{w} = \mathbf{B}\lfloor(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\boldsymbol{t}\rceil$. The offset vector obtained can therefore be written as

$$\boldsymbol{t} - \boldsymbol{w} = \mathbf{B} \cdot \left((\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\boldsymbol{t} - \lfloor(\mathbf{B}^T\mathbf{B})^{-1}\mathbf{B}^T\boldsymbol{t}\rceil\right) \in \mathbf{B} \cdot \left[-\frac{1}{2}, \frac{1}{2}\right]^n. \tag{5.2}$$

Let $\boldsymbol{b}'_i$ denote the i^{th} *row* of $\mathbf{B}$. From Equation (5.2), we have that

$$\|\boldsymbol{t} - \boldsymbol{w}\|_\infty \leq \frac{1}{2} \cdot \max_i \|\boldsymbol{b}'_i\|_1 \leq \frac{\sqrt{n}}{2} \cdot \max_i \|\boldsymbol{b}'_i\|_2. \tag{5.3}$$

We now use this analysis to give an upper bound on $\|(u, v)\|_\infty$ that is computed in the algorithm from Section 5.3. At a high level, the first four steps find a (poten-

[1] available at http://perso.ens-lyon.fr/damien.stehle/NTRU.html

tially very long) pair $(u, v) \in R$ such that $us + vt = 1 \mod R_q$ and the final two steps update this (u, v) using Babai's rounding technique. In particular, suppose we define $\mathbf{S}, \mathbf{T} \in \mathbb{Z}_q^{n \times n}$ to be the negacyclic matrices denoting multiplication by s and t respectively. Then the final two steps run Babai's rounding technique on the lattice $\Lambda = \{\boldsymbol{z} \in \mathbb{Z}^{2n} : [\mathbf{S}|\mathbf{T}] \cdot \boldsymbol{z} = \mathbf{0}\}$ and target point $\boldsymbol{t} = (u, v)$ (using the coefficient embedding), and update (u, v) to be the resulting offset. The basis for Λ used is $\mathbf{B} = [\mathbf{T}| - \mathbf{S}]^T \in \mathbb{Z}^{2n \times n}$ (which has linearly independent columns by invertibility of s, t in the field $\mathbb{Q}(X)/\langle X^n + 1\rangle$). Therefore, bounding the infinity norm of the offset (via Equation (5.3)) gives us a bound for the final value of $\|(u, v)\|_\infty$. Noting that each row of our basis consists of the coefficients of $s, t \leftarrow \chi_\sigma$, we obtain the bound

$$\|(u, v)\|_\infty \leq \frac{\sqrt{n}}{2} \cdot \|(s, t)\|_2 \leq \frac{\sqrt{n}}{2} \cdot 2\sigma\sqrt{n} = n\sigma \tag{5.4}$$

that holds with all but negligible probability over the choice of s and t.

5.3.2 Zero Knowledge Argument of Knowledge Statements

We now discuss the statements associated with the ZKAoKs in our construction given in Figure 5.2. The arguments of prover P_i fall into two groups separated by a colon. Arguments before a colon are intended as "secret" information pertaining to a witness for a statement. Arguments after a colon should be interpreted as "public" information describing the statement that is being proved.

A note on common reference strings (CRS) Our construction contains common reference strings (CRSs). A CRS shared by prover and verifier is commonly used in non-interactive ZKAoK/ZKPoKs as follows. An honestly sampled CRS ensures the soundness and completeness of the proof system. Additionally, there is an algorithm producing simulated CRSs from a distribution that is computationally indistinguishable from that of an honestly sampled CRS. It is a simulated CRS that is used to prove the zero-knowledge property. More specifically, there is an efficient algorithm producing simulated proofs with the property that the following two distributions are computationally indistinguishable for any instance in the underlying language:

1. The joint distribution of a simulated CRS and simulated proof,

2. The joint distribution of an honest CRS and honestly produced proof.

In order to accommodate a wider class of non-interactive ZKPoKs, we include CRSs explicitly in our construction.

5.3.2.1 Client Proof.

The client proof denoted $\mathsf{P}_1(x, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2 : \mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1)$ should prove knowledge of

- $x \in \{0,1\}^L$
- $s, t \in R$ where $|s|_\infty, |t|_\infty \leq \sigma \cdot \sqrt{n}$
- $\boldsymbol{e}_1, \boldsymbol{e}_2 \in R^{1\times\ell}$ where $|\boldsymbol{e}_1|_\infty, |\boldsymbol{e}_2|_\infty \leq L \cdot \ell \cdot \sigma \cdot n^{3/2}$

such that

$$\boldsymbol{c}_x^1 = \boldsymbol{a}_x \cdot s + \boldsymbol{e}_1 \bmod q,$$
$$\boldsymbol{c}_x^2 = \boldsymbol{a}_x \cdot t + \boldsymbol{e}_2 \bmod q.$$

5.3.2.2 Server Proofs.

The server proof in the *initialisation phase* denoted $\mathsf{P}_0(k, e, e' : \mathsf{crs}_0)$ has the purpose of proving knowledge of $k, e, e' \in R$ where $|k|_\infty, |e|_\infty, |e'|_\infty \leq \sigma \cdot \sqrt{n}$ such that

$$c_1 = a \cdot k + e \bmod q,$$
$$c_2 = b \cdot k + e' \bmod q,$$

where crs_0 contains (a, b).

The server proof in the *query phase* denoted by

$$\mathsf{P}_2(k, \boldsymbol{e}_1', \boldsymbol{e}_2', e, e' : \mathsf{crs}_0, \mathsf{crs}_2, c_1, c_2, \boldsymbol{d}_x^1, \boldsymbol{d}_x^2, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2)$$

has the purpose of proving that there is some

- $k, e, e' \in R$ where $|e|_\infty, |e'|_\infty \leq \sigma \cdot \sqrt{n}$
- $\boldsymbol{e}'_1, \boldsymbol{e}'_2 \in R^{1\times\ell}$ where $|\boldsymbol{e}'_1|_\infty, |\boldsymbol{e}'_2|_\infty \leq \sigma' \cdot \sqrt{n}$

such that

$$\begin{aligned} c_1 &= a \cdot k + e \bmod q, \\ c_2 &= b \cdot k + e' \bmod q, \\ \boldsymbol{d}^1_x &= \boldsymbol{c}^1_x \cdot k + \boldsymbol{e}'_1 \bmod q, \\ \boldsymbol{d}^2_x &= \boldsymbol{c}^2_x \cdot k + \boldsymbol{e}'_2 \bmod q. \end{aligned} \tag{5.5}$$

It is important to note that both $\boldsymbol{d}^1_x$ and $\boldsymbol{d}^2_x$ each consist of ℓ ring elements. Therefore, the above system consists of a total of $2 + 2\ell$ noisy products of public ring elements and k. We draw attention to the fact that the server need not prove a bound on k in the query phase, as this was dealt with in the initialisation phase. The fact that two RLWE samples are used during the initialisation phase forces the server to commit to a single k since RLWE with two samples has a unique secret with extremely high probability in the parameter setting that we will use (see Section 5.2.1). If only one RLWE sample was used, then a malicious server could use a different key k' in the query phase consistent with the single RLWE sample and produce a passing proof in the query phase using k' instead of k (since P_2 is not asked to give bound on the key). Adding a length bound requirement to k would allow for the use of a single RLWE sample. Nonetheless, we use the given formulation to reduce the burden of designing the proof system π_2. We do this in the hope of future work that designs more efficient alternatives to the realisation of π_2 given in this thesis. Moreover, π_2 in the query phase need not prove *knowledge* of k as this was also dealt with in the initialisation phase.

5.3.3 Correctness

Before proving correctness, we present a lemma that will prove useful.

Lemma 17. *Fix any $x \in \{0,1\}^L$. Suppose there exists a PPT algorithm $\mathcal{D}(x, \boldsymbol{a}_0, \boldsymbol{a}_1)$ that outputs $r \in R$ such that $\|r\| \leq B$ and at least one coefficient of $\boldsymbol{a}_x \cdot r$ is in the set $(q/p) \cdot \mathbb{Z} + [-T, T]$ with non-negligible probability (over a uniform choice of*

$\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^\ell$ *and its random coins). Then there exists an efficient algorithm solving* $\mathsf{1D\text{-}SIS}_{q/p,n\ell,\max\{n\ell B,T\}}$ *as defined in Definition 23.*

Proof. Consider the following algorithm $\mathcal{A}$ using $\mathcal{D}$ as a sub-routine that attempts to solve $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,n\ell,T}$ on uniform input $\boldsymbol{v} \in \mathbb{Z}_q^{n\ell}$:

1. Let $j \in \{0,1\}$ denote the first bit of x and set $\boldsymbol{w}^j := \boldsymbol{v} \in \mathbb{Z}_q^{n\ell}$.
2. Sample $\boldsymbol{w}^{\bar{j}} \leftarrow \mathbb{Z}_q^{n\ell}$
3. For $i = 0, \ldots, \ell - 1$:
   $(a_j)_i = \sum_{k=0}^{n-1} w_{in+k}^j X^k$
   $(a_{\bar{j}})_i = \sum_{k=0}^{n-1} w_{in+k}^{\bar{j}} X^k$
4. Run $r \leftarrow \mathcal{D}(x, \boldsymbol{a}_0, \boldsymbol{a}_1)$.
5. If there is no coefficient of $\boldsymbol{a}_x \cdot r$ in the set $(q/p) \cdot \mathbb{Z} + [-T, T]$, then abort.
6. Otherwise let x' be the input x with the first bit removed. There is a coefficient of $\boldsymbol{a}_x \cdot r = \boldsymbol{a}_j \cdot G^{-1}(\boldsymbol{a}_{x'}) \cdot r$ in $(q/p) \cdot \mathbb{Z} + [-T, T]$ meaning that for some k^*, there is a column of $G^{-1}(\boldsymbol{a}_{x'}) \cdot r$, say $\boldsymbol{y} \in R_q^\ell$ such that the X^{k^*} coefficient of $\langle \boldsymbol{a}_j, \boldsymbol{y} \rangle$ is in $(q/p) \cdot \mathbb{Z} + [-T, T]$.
7. Let $\mathbb{1}_{(\cdot)}$ be an indicator function. Noting that the coefficient of X^{k^*} of $\langle \boldsymbol{a}_j, \boldsymbol{y} \rangle$ is equal to
$$\sum_{i=0}^{\ell-1} \sum_{k=0}^{n-1} v_{in+k} \cdot (-1)^{\mathbb{1}_{k>k^*}} (y_i)_{k^*-k \bmod n},$$
output $\boldsymbol{z} \in \mathbb{Z}_q^{n\ell}$ where $z_{in+k} = (-1)^{\mathbb{1}_{k>k^*}} (y_i)_{k^*-k \bmod n}$ for $i = 0, \ldots, \ell - 1$, $k = 0, \ldots, n - 1$.

It is clear that if $\mathcal{A}$ does not abort, it outputs a vector $\boldsymbol{z} \in \mathbb{Z}_q^{n\ell}$ such that $\langle \boldsymbol{v}, \boldsymbol{z} \rangle \in (q/p) \cdot \mathbb{Z} + [-T, T]$. Furthermore, if no abort occurs, then the entries of $\boldsymbol{z}$ (up to a sign) correspond to the coefficients of a column of $r \cdot G^{-1}(\boldsymbol{a}_{x'})$ where $\|r\|_\infty \leq B$ with non-negligible probability. Recalling that $G^{-1}(\boldsymbol{a}_{x'}) \in R_q^{\ell \times \ell}$ is a binary decomposition of polynomials, we can see that,

$$\|\boldsymbol{z}\|_\infty \leq \ell \cdot n \cdot B$$

with non-negligible probability. In other words, $\mathcal{A}$ solves the $\mathsf{1D\text{-}SIS\text{-}R}_{q,p,n\ell,n\ell B}$ problem in polynomial time with non-negligible probability. To complete the proof, we use Lemma 9. □

Lemma 18 (Correctness)**.** *Adopt the notation of Figure 5.2, assuming an honest client and server. Define* $T := \sigma n^2 \cdot (L\ell\sigma^2 n^{5/2} + \sigma')$*. For any* $x \in \{0,1\}^L, k \in R_q$ *such that* $\|k\|_\infty \leq \sigma \cdot \sqrt{n}$*, we have that*

$$\Pr[\boldsymbol{y}_x \neq F_k(x)] \leq \mathsf{negl}(\lambda)$$

over the choice of PRF parameters $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$ *assuming the hardness of* $\mathsf{1D\text{-}SIS}_{q/p,n\ell,T}$*.*

Proof. Fix an arbitrary x. Assume that there exists a k such that $\|k\| \leq \sigma \cdot \sqrt{n}$ and $\Pr[\boldsymbol{y}_x \neq F_k(x)]$ is non-negligible over the choice of $\boldsymbol{a}_0, \boldsymbol{a}_1 \leftarrow R_q^{1\times\ell}$. Expanding $\boldsymbol{d}_x^1$ and $\boldsymbol{d}_x^2$, we have that

$$\boldsymbol{y}_x = \left\lfloor \boldsymbol{a}_x \cdot k + u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') \right\rceil_p .$$

Note that $\boldsymbol{e} := u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2')$ has infinity norm less than T (as defined in the lemma statement) with all but negligible probability. Therefore, if $\boldsymbol{y}_x \neq F_k(x)$ with non-negligible probability, it must be that at least one coefficient of $\boldsymbol{a}_x \cdot k$ in the set $(q/p)\cdot\mathbb{Z} + [T,T]$ with non-negligible probability. Applying Lemma 17 to the algorithm $\mathcal{D}(x)$ that ignores $\boldsymbol{a}_0, \boldsymbol{a}_1$ and simply outputs k implies an efficient algorithm solving $\mathsf{1D\text{-}SIS}_{q/p,n\ell,\max\{n^{3/2}\ell\sigma,T\}}$. □

5.4 VOPRF Security Proof

In this section, we show that the protocol in Figure 5.2 is a VOPRF achieving security against malicious adversaries. In particular, corrupted clients and servers that attempt to subvert the protocol learn/affect only as much as in an ideal world, where they interact via the functionality $\mathcal{F}_{\mathsf{VOPRF}}$.

Theorem 9. (Security) *Assume* $p|q$*. The protocol in Figure 5.2 is a secure VOPRF in the sense of Definition 34 provided that the following conditions hold:*

- $\mathsf{RLWE}_{q,n,\sigma}$ *is hard,*

- $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$,
- $\mathsf{1D\text{-}SIS}_{q/(2p),n\cdot\ell,4\cdot\sigma'\cdot\sigma\cdot n^{5/2}}$ *is hard.*

5.4.0.1 Correctness against non-aborting malicious client transcripts

During the malicious client proof, it will be useful to call upon the fact that any non-aborting protocol transcript allows for the computation of $F_k(x)$.

Lemma 19. *Assume that* $\mathsf{RLWE}_{q,n,\sigma}$ *is hard,* σ *and* n *are* $\mathsf{poly}(\lambda)$*, and* $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$*. For any* $x \in \{0,1\}^L$*, consider a non-aborting run of the protocol in Figure 5.2 between a (potentially malicious) efficient client* C^* *and honest server* S*. Consider any* $u, v \in R_q$*, such that* $\|u\|_\infty, \|v\|_\infty \leq \sigma n$ *and* $u \cdot s + v \cdot t = 1$*, where* s, t *are extracted from* C^**'s proof in its message to* S*. Then, the value of* $\left\lfloor u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2 \right\rceil_p$ *is equal to* $\lfloor \boldsymbol{a}_x \cdot k \rceil_p$ *with all but negligible probability over the choice of* $\boldsymbol{a}_0, \boldsymbol{a}_1$ *and* k*.*

Proof. We use the notation from Figure 5.2. First note that for a *non-aborting* protocol run, any *efficient* client C^* must have produced $\boldsymbol{c}_x^1$ and $\boldsymbol{c}_x^2$ correctly using some $x \in \{0,1\}^L, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2$ where $\|s\|_\infty, \|t\|_\infty \leq \sigma \cdot \sqrt{n}$ and $\|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq L \cdot \ell \cdot \sigma \cdot n^{3/2}$. To complete the proof, we will use the fact that $\frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e})$ is computationally indistinguishable from uniform random over $\frac{p}{q} \cdot R_q^{1\times\ell}$ when $\boldsymbol{e} \leftarrow \mathcal{E}_{a_0,a_1,x,\sigma}$ assuming the hardness of $\mathsf{RLWE}_{q,n,\sigma}$ (Lemma 15). This implies that every coefficient in $\frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e})$ is at least T' away from $\mathbb{Z} + 1/2$ with all but negligible probability for any $T' \ll 1$. We will use this fact twice to complete the proof. With this in mind, a client computing the output as prescribed in Figure 5.2 obtains

$$\left\lfloor \frac{p}{q}(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2) \right\rceil = \left\lfloor \frac{p}{q}\boldsymbol{a}_x \cdot k + \frac{p}{q}u \cdot (\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + \frac{p}{q}v \cdot (\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') \right\rceil. \tag{5.6}$$

The quantity $\left\lfloor \frac{p}{q}(\boldsymbol{a}_x \cdot k + \boldsymbol{e}) \right\rceil$ can be shown to be equal to Equation (5.6) (with all but negligible probability) using the negligible value of

$$T_0' = \frac{3p}{q}\sigma n^2(L \cdot \ell \cdot \sigma^2 \cdot n^{5/2} + \sigma') \geq \left\| \frac{p}{q}u(\boldsymbol{e}_1 \cdot k + \boldsymbol{e}_1') + \frac{p}{q}v(\boldsymbol{e}_2 \cdot k + \boldsymbol{e}_2') - \frac{p}{q}\boldsymbol{e} \right\|_\infty.$$

Furthermore, $\left\lfloor \frac{p}{q} \left(\boldsymbol{a}_x \cdot k + \boldsymbol{e} \right) \right\rceil$ is equal to $\left\lfloor \frac{p}{q} \boldsymbol{a}_x \cdot k \right\rceil$ with all but negligible probability, using the negligible value of

$$T_1' = \frac{p}{q} \cdot L \cdot \ell \cdot \sigma \cdot n^{3/2} \geq \left\| \frac{p}{q} \boldsymbol{e} \right\|_\infty .$$

. □

5.4.1 Malicious Client Proof

Lemma 20 (Average case malicious client security)**.** *Assume that σ and n are* $\mathsf{poly}(\lambda)$*, let $p|q$, and let conditions (i) and (ii) be as follows:*

(i) $\mathsf{RLWE}_{q,n,\sigma}$ *is hard,*

(ii) $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$.

If the above conditions hold, then the protocol in Figure 5.2 has average case security against malicious clients.

Proof. We describe a simulation $\mathcal{S}$ that communicates with the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ (environment) on one hand, and the malicious client C^* on the other. $\mathcal{S}$ carries out the following steps:

1. During CRS.SetUp, publish honest $\boldsymbol{a}_0, \boldsymbol{a}_1, \mathsf{crs}_1$ and (dishonest) simulated versions of crs_0 and crs_2. Denote the simulated CRS elements by crs_0' and crs_2'.
2. During the **Init** phase, send C^* a uniform $c_1, c_2 \leftarrow R_q$ with a simulated proof $\pi_{0,\mathsf{Sim}}$ and pass the init message onto $\mathcal{F}_{\mathsf{VOPRF}}$. Initialise an empty list $\mathsf{received}$.
3. During the **Query** stage, for each message $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ from C^*, do the following:
   (a) $b \leftarrow \mathsf{V}_1(\mathsf{crs}_1, \boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \boldsymbol{a}_0, \boldsymbol{a}_1, \pi_1)$. If $b = 0$ send abort to the functionality and abort the protocol with the malicious client. If $b = 1$ continue to the next step.

(b) Extract the values x, s, t from π_1 using the ZKAoK extractor and send (query, x) to the functionality.

(c)
- If $\mathcal{F}_{\mathsf{VOPRF}}$ aborts:
  $\mathcal{S}$ aborts.
- If $\mathcal{F}_{\mathsf{VOPRF}}$ returns $\boldsymbol{y} \in R_p^{1\times\ell}$ **and** $\forall \boldsymbol{y}^*, (x, \boldsymbol{y}^*) \notin \mathsf{received}$:
  (i.e. if this is the first time x is queried) uniformly sample
  $$\boldsymbol{y}_q \leftarrow R_q^{1\times\ell} \cap \left(\frac{q}{p}\boldsymbol{y} + R^{1\times\ell}_{\leq \frac{q}{2p}}\right)$$
  and do $\mathsf{received.add}(x, \boldsymbol{y}_q)$.
- If $\mathcal{F}_{\mathsf{VOPRF}}$ returns $\boldsymbol{y} \in R_p^{\ell}$ **and** $\exists \boldsymbol{y}^* \text{s.t.} (x, \boldsymbol{y}^*) \in \mathsf{received}$:
  (i.e. x was previously queried) Then set $\boldsymbol{y}_q = \boldsymbol{y}^*$.

(d) Next pick $\bar{\boldsymbol{e}}'_1, \bar{\boldsymbol{e}}'_2 \leftarrow \chi_{\sigma'}$ and set
$$\bar{\boldsymbol{d}}^1_x = \boldsymbol{y}_q \cdot s + \bar{\boldsymbol{e}}'_1 \bmod q,$$
$$\bar{\boldsymbol{d}}^2_x = \boldsymbol{y}_q \cdot t + \bar{\boldsymbol{e}}'_2 \bmod q.$$
Finally, produce a simulated proof $\pi_{2,\mathsf{Sim}}$ using crs'_2 and send $(\bar{\boldsymbol{d}}^1_x, \bar{\boldsymbol{d}}^2_x, \pi_{2,\mathsf{Sim}})$ to C^*.

We now argue that C^* cannot decide whether it is interacting with $\mathcal{S}$ or with a genuine server. Firstly, recognise that $\mathsf{crs}'_0, \mathsf{crs}'_2$ is indistinguishable from honestly created $\mathsf{crs}_0, \mathsf{crs}_2$. Secondly, the malicious client cannot distinguish the simulator's uniform c_1, c_2 that it sends during the **Init** phase from the real protocol by the $\mathsf{RLWE}_{q,n,\sigma}$ assumption (condition (i)). This implies that both the SetUp and **Init** phases that $\mathcal{S}$ performs are indistinguishable from the real protocol.

The most challenging step is arguing that the simulator's behaviour in the **Query** phase is indistinguishable from the real protocol from the malicious client's point of view. We will analyse the behaviour of the simulator assuming that no abort is triggered. We begin by arguing that the server message in the real protocol with respect to any triple (x, s, t) can be replaced by a related message $(\boldsymbol{a}_x \cdot k + \boldsymbol{e}^x) \cdot s + \hat{\boldsymbol{e}}'_1$ where $\boldsymbol{e}^x \leftarrow \mathcal{E}_{\boldsymbol{a}_0, \boldsymbol{a}_1, x, \sigma}$ and $\hat{\boldsymbol{e}}'_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$ (and similarly for the message depending on t) without detection by the following statistical argument. For brevity, consider the quantities that depend on s, i.e. $\boldsymbol{c}^1_x$ and $\boldsymbol{d}^1_x$ (a similar argument holds for the

quantities depending on t). We have that the server response in the *real* protocol has $\boldsymbol{d}_x^1$ of the form

$$(\boldsymbol{a}_x \cdot s + \boldsymbol{e}_1) \cdot k + \boldsymbol{e}_1' \tag{5.7}$$

where $\boldsymbol{e}_1 \leftarrow \mathcal{E}_{a_0,a_1,x,\sigma}$ and $\boldsymbol{e}_1' \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. By Lemma 3, the message distribution in Equation (5.7) is statistically indistinguishable (condition (ii)) from

$$\boldsymbol{a}_x \cdot k \cdot s + \boldsymbol{e}_1'' \tag{5.8}$$

where $\boldsymbol{e}_1'' \leftarrow R(\chi_{\sigma'})^{1\times\ell}$ due to the fact that $\sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$. By a similar argument, the quantity given in Equation (5.8) is statistically close in distribution to

$$(\boldsymbol{a}_x \cdot k + \boldsymbol{e}^x) \cdot s + \boldsymbol{e}_1'''. \tag{5.9}$$

where $\boldsymbol{e}^x \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ and $\boldsymbol{e}_1''' \leftarrow R(\chi_{\sigma'})^{1\times\ell}$.

Using Lemma 15 and condition (i), we have that the term in front of s in Equation (5.9) is indistinguishable from uniform by the hardness of $\mathsf{RLWE}_{q,n,\sigma}$ (Lemma 15). In particular, from an efficient C^*'s point of view, $\boldsymbol{d}_x^1$ cannot be distinguished from

$$\boldsymbol{u}_x \cdot s + \boldsymbol{e}_1$$

where $\boldsymbol{u}_x \leftarrow R_q^{1\times\ell}$ and $\boldsymbol{e}_1 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. Similarly, $\boldsymbol{d}_x^2$ cannot be distinguished from $\boldsymbol{u}_x \cdot t + \boldsymbol{e}_2$ for the same $\boldsymbol{u}_x$ as above and $\boldsymbol{e}_2 \leftarrow R(\chi_{\sigma'})^{1\times\ell}$. Note that on repeated queries, the errors sampled from $R(\chi_{\sigma'})^{1\times\ell}$ are fresh. The fact that $\mathcal{S}$ samples $\boldsymbol{y}_q$ from a uniform element of a uniform interval implies the indistinguishability part of the average-case malicious client security definition.

Next, we show that if the malicious client can indeed compute the correct value from the messages it receives from the honest server (in the real protocol), then it can do the same with the messages that it receives from the simulator. In Lemma 19, we show that a malicious client which does not cause an abort can compute $\lfloor \boldsymbol{a}_x \cdot k \rceil_p$ from the messages it receives from the honest server with all but negligible probability. We now show that this is also the case with the messages it receives from $\mathcal{S}$. Consider $\boldsymbol{y}_q$ sampled by $\mathcal{S}$ and the corresponding values $\bar{\boldsymbol{d}}_x^1$ and $\bar{\boldsymbol{d}}_x^2$. In addition, define $\boldsymbol{e} := \boldsymbol{y}_q - (q/p) \cdot \boldsymbol{y} \in R_{\leq \frac{q}{2p}}^{1\times\ell}$ so that $\boldsymbol{e}$ follows the uniform distribution over $R_{\leq \frac{q}{2p}}^{1\times\ell}$. We have that

$$\left\lfloor \frac{p}{q}(u \cdot \bar{\boldsymbol{d}}_x^1 + v \cdot \bar{\boldsymbol{d}}_x^2) \right\rceil = \left\lfloor \boldsymbol{y} + \frac{p}{q}\left(\boldsymbol{e} + u \cdot \bar{\boldsymbol{e}}_1' + v \cdot \bar{\boldsymbol{e}}_2'\right) \right\rceil. \tag{5.10}$$

We also know that with all but negligible probability, $\|u \cdot \bar{\boldsymbol{e}}'_1 + v \cdot \bar{\boldsymbol{e}}'_2\|_\infty \leq \sigma \cdot \sigma' \cdot n^{5/2}$ (since no abort occurred) and that $\|\boldsymbol{e}\|_\infty$ is less than $q/(2p) - T$ with all but negligible probability as long as $T \ll (q/2p)$. Taking $T = \sigma \cdot \sigma' \cdot n^{5/2}$, we get that with all but negligible probability,

$$\left\| \frac{p}{q} \cdot \left(\boldsymbol{e} + u \cdot \bar{\boldsymbol{e}}'_1 + v \cdot \bar{\boldsymbol{e}}'_2 \right) \right\|_\infty \leq \frac{p}{q} \|\boldsymbol{e}\|_\infty + \frac{p}{q} \|u \cdot \bar{\boldsymbol{e}}'_1 + v \cdot \bar{\boldsymbol{e}}'_2\|_\infty < \frac{1}{2},$$

implying that the quantity in Equation (5.10) rounds correctly to $\boldsymbol{y}$ with all but negligible probability. Therefore, both the real protocol and simulator enable correct evaluation of the PRF. □

5.4.2 Malicious Server Proof

Lemma 21. *Let conditions (i),(ii) and (iii) be as follows:*

(i) $\mathsf{RLWE}_{q,n,\sigma}$ *is hard,*

(ii) $\sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^{5/2}$,

(iii) $\mathsf{1D\text{-}SIS}_{q/(2p),n\cdot\ell,4\sigma'\sigma n^{5/2}}$ *is hard.*

If the above conditions hold, then the protocol in Figure 5.2 is secure in the presence of malicious servers.

Proof. We construct a simulator Sim interacting with the malicious server S^* on one hand and with the functionality $\mathcal{F}_{\mathsf{VOPRF}}$ on the other. The simulator Sim behaves as follows:

1. During the **SetUp** phase, publish honest $a_0, a_1, \mathsf{crs}_0, \mathsf{crs}_2$ and (dishonest) simulated crs'_1 to use with the proof systems.
2. During the **Init** phase, if S^* sends $c_1, c_2 \in R_q$ and an accepting proof π_0, then use the zero knowledge extractor to obtain a key k' from π_0 and forward this on to the functionality. If the message is not of the correct format, or the proof does not verify, then abort.

3. During the **Query** phase, select two uniform random values $\boldsymbol{u}_1, \boldsymbol{u}_2 \leftarrow R_q^{1\times\ell}$, and using the ZK simulator, produce a simulated proof $\pi_{1,\mathsf{Sim}}$ using crs_1'. Send the message $(\boldsymbol{u}_1, \boldsymbol{u}_2, \pi_{1,\mathsf{Sim}})$. Wait for a response of the form $(\tilde{\boldsymbol{d}}_x^1, \tilde{\boldsymbol{d}}_x^2, \tilde{\pi}_2)$ from S^*. If the proof $\tilde{\pi}_2$ verifies[2], forward on $\mathsf{deliver}$ to $\mathcal{F}_{\mathsf{VOPRF}}$. Otherwise, forward abort to $\mathcal{F}_{\mathsf{VOPRF}}$.

We will show that the joint output of an honest client C and S^* in the real world (where they interact directly) and the ideal world (where they interact via $\mathcal{F}_{\mathsf{VOPRF}}$ and $\mathcal{S}$) are computationally indistinguishable. We begin by arguing that the malicious server S^* cannot distinguish whether it is interacting with a real client or $\mathcal{S}$, as described above. Firstly, replacing crs_1 by crs_1' is indistinguishable from the point of view of S^* by definition of a simulated CRS. Importantly, if S^* can produce valid proofs in the **Init** phase, the key k' with $\|k\|_\infty \leq \sigma \cdot \sqrt{n}$ obtained by the simulator is the *unique* ring element consistent with c_1, c_2. This is due to the fact that RLWE is well-defined given two samples (see chapter preliminaries).

All that is left to consider is the **Query** phase. Note that in the real protocol, the client produces two values $\boldsymbol{c}_x^1, \boldsymbol{c}_x^2$ that are pseudorandom under the hardness of $\mathsf{RLWE}_{q,n,\sigma}$ by Lemma 15. Therefore, the malicious server S^* cannot distinguish a real $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2)$ from the pair $(\boldsymbol{u}_1, \boldsymbol{u}_2)$ that Sim uses. By the properties of a ZK simulator, it follows that a real client message $(\boldsymbol{c}_x^1, \boldsymbol{c}_x^2, \pi_1)$ and crs_1 is indistinguishable from $(\boldsymbol{u}_1, \boldsymbol{u}_2, \pi_{1,\mathsf{Sim}})$ and crs_1'. Next, if the response from S^* has a valid proof, then Sim forwards on $\mathsf{deliver}$. This means that the ideal functionality passes a PRF evaluation to the client using the server key k'. We now argue that this emulates the output on the client side when running the real protocol with malicious server S^*.

The case where the proof verification fails is trivial since the client aborts in the real and ideal worlds. As a result, we focus on the case where the zero knowledge proof produced by S^* in the query phase verifies correctly. Let $\boldsymbol{e}_1, \boldsymbol{e}_2 \leftarrow \mathcal{E}_{\boldsymbol{a}_0,\boldsymbol{a}_1,x,\sigma}$ be sampled by the honest client. For this honest client interacting with malicious S^* in the real protocol, observe that

$$\frac{p}{q}\left(u \cdot \boldsymbol{d}_x^1 + v \cdot \boldsymbol{d}_x^2\right) = \frac{p}{q}\boldsymbol{a}_x k' + \frac{p}{q}(\boldsymbol{e}_1 k' + \boldsymbol{e}_1')u + \frac{p}{q}(\boldsymbol{e}_2 k' + \boldsymbol{e}_2')v \tag{5.11}$$

for $k', \boldsymbol{e}_1', \boldsymbol{e}_2'$ chosen by S^* where $\|k'\|_\infty \leq \sigma \cdot \sqrt{n}$ and $\|\boldsymbol{e}_1'\|_\infty, \|\boldsymbol{e}_2'\|_\infty \leq \sigma' \cdot \sqrt{n}$.

[2] Alternatively, if $\tilde{\boldsymbol{d}}_x^1, \tilde{\boldsymbol{d}}_x^2$ is consistent with k'

Therefore, rounding the quantity in Equation (5.11) is guaranteed to result in the correct value if every coefficient of $\frac{p}{q} \cdot \boldsymbol{a}_x k'$ is further than

$$\left\| \frac{p}{q}(\boldsymbol{e}_1 k' + \boldsymbol{e}'_1)u + \frac{p}{q}(\boldsymbol{e}_2 k' + \boldsymbol{e}'_2)v \right\|_\infty$$

away from $\mathbb{Z} + 1/2$. In other words, if S^* can force incorrect evaluation, it must have found $k' \leq \sigma \cdot \sqrt{n}$ such that a coefficient of $\boldsymbol{a}_x k'$ is within a distance

$$\begin{aligned} &\left\| (\boldsymbol{e}_1 \cdot k' + \boldsymbol{e}'_1) \cdot u + (\boldsymbol{e}_2 \cdot k' + \boldsymbol{e}'_2) \cdot v \right\|_\infty \\ &\qquad \leq 2\left(L \cdot \ell \cdot \sigma \cdot n^{5/2} \cdot \sigma \cdot \sqrt{n} + \sigma' \cdot \sqrt{n}\right) \cdot \sigma \cdot n^2 \\ &\qquad \overset{condition(ii)}{\leq} 4 \cdot \sigma' \cdot \sigma \cdot n^{5/2} \end{aligned}$$

of $\frac{q}{p}\mathbb{Z} + \frac{q}{2p} \subset \frac{q}{2p}\mathbb{Z}$. At this point we apply Lemma 17 using $2 \cdot p$ and $T = 4 \cdot \sigma' \cdot \sigma \cdot n^{5/2}$ to show that S^* forcing incorrect evaluation with non-negligible probability violates the assumption that

$$\mathsf{1D\text{-}SIS}_{q/2p, n\cdot\ell, \max\{n^{3/2}\cdot\ell\cdot\sigma, 4\cdot\sigma'\cdot\sigma\cdot n^{5/2}\}}$$

is hard. Therefore, condition (iii) enforces correct evaluation with all but negligible probability when the parameters satisfy condition (ii). □

5.4.3 Setting the parameters

Let λ be the security parameter. Theorem 9 requires the following conditions:

- $\frac{q}{2p} \gg \sigma' \gg L \cdot \ell \cdot \sigma^2 \cdot n^3$
- $\mathsf{RLWE}_{q,n,\sigma}$ is hard
- $\mathsf{1D\text{-}SIS}_{q/(2p), n\ell, 4\cdot\sigma'\cdot\sigma\cdot n^{5/2}}$ is hard.

We will be using the presumed hardness of SIVP_γ for approximation factors $\gamma = 2^{o(\sqrt{n})}$. The SIVP_γ lattice dimension associated to RLWE will be $n = \lambda^c$ (for some constant c); the dimension associated to $\mathsf{1D\text{-}SIS}$ hardness will be $n' := \lambda$. We first choose $\sigma = \mathsf{poly}(n)$ and $\sigma' = \sigma \cdot \lambda^{\omega(1)}$, and then set $q = p \cdot \prod_{i=1}^{n'} p_i$ by picking coprime $p, p_1, \ldots, p_{n'} = 4\sigma'\sigma n^{5/2} \cdot \omega(\sqrt{nn' \log q \log n'})$. Having made these choices, it should

Parameter	Description	Requirement	Asymptotic
n	ring dimension	$n = \mathsf{poly}(\lambda)$	$\mathsf{poly}(\lambda)$
q	original modulus	$q = p \cdot \sigma' \cdot \lambda^{\omega(1)}$	$\lambda^{\omega(1)}$
p	rounding modulus	—	$\mathsf{poly}(\lambda)$
ℓ	$\log_2(q)$	—	$\log_2(\lambda^{\omega(1)})$
σ	secret/error distribution	$q/\sigma = 2^{o(\sqrt{n})}$	$\mathsf{poly}(\lambda)$
σ'	drowning distribution	$\sigma' = L\ell\sigma^2 n \cdot \lambda^{\omega(1)}$	$\lambda^{\omega(1)}$
L	bit-length of PRF input	—	—

Table 5.1: Parameters of our VOPRF

be clear that the first of the three conditions is satisfied. We can apply Theorem 2 to argue RLWE hardness via SIVP for sub-exponential approximation factors $2^{\tilde{\mathcal{O}}(n^{1/c})}$ (for $c > 2$), noting that $\sigma = \mathsf{poly}(n)$ and

$$\begin{aligned} q &= (4 \cdot \sigma' \cdot \sigma \cdot n^{5/2})^{n'} \omega((n \cdot n' \cdot \log q \cdot \log n')^{n'/2}) \\ &= 2^{(2+2\log\sigma+\omega(1)\log\lambda+(5/2)\log n)\cdot n^{1/c}} \cdot \omega((n \cdot n' \cdot \log q \cdot \log n')^{n'/2}) \\ &= 2^{\omega(1)\cdot n^{1/c}\cdot \log n} \cdot \omega((n^{1+\frac{1}{c}} \cdot \log q \cdot \log n)^{n^{1/c}/2}) \\ &= 2^{\tilde{\mathcal{O}}(n^{1/c})}. \end{aligned}$$

Finally for the 1D-SIS condition, we note that $q/p = \prod_{i=1}^{n'} p_i$ and

$$\begin{aligned} p_1 &= 4 \cdot \sigma' \cdot \sigma \cdot n \cdot \omega(\sqrt{n \cdot n' \log q \cdot \log n'}) \\ &= 4 \cdot \sigma^2 \cdot \lambda^{\omega(1)} n \cdot \omega(\sqrt{n \cdot n' \cdot \log q \cdot \log n'}) \\ &= (n')^{\omega(1)} \cdot \omega(\sqrt{n'^{1+c} \cdot \log q \cdot \log n'}). \end{aligned}$$

So applying Lemma 8, we get hardness of our 1D-SIS instance via the presumed hardness of SIVP on n'-dimensional lattices for $(n')^{\omega(1)} \cdot \mathsf{poly}(n')$ approximation factors. We summarise the parameters of our construction in Table 5.1.

5.5 Post-Quantum Zero Knowledge Instantiations (High level)

We now describe high-level instantiations of our zero knowledge proofs of knowledge. At a high level, we may use (parallel repetitions of) Stern-based proofs along with the Fiat-Shamir transform for all proof systems as in [86] (although there may be other alternatives e.g. using the optimised protocol of Beullens [24]). Recall that the Fiat-Shamir transform has recently been proven secure in the QROM [50, 91].

We place most of our attention on discussing how to instantiate Proof System 1, as the other proof systems may be derived straight-forwardly using a subset of the techniques arising in Proof System 1. For more precise details on how to instantiate Proof System 1 using Stern's protocol, see Section 5.6. From now on, we use non-bold capital letters to denote matrices with entries in $\mathbb{Z}_q$, and over-arrows to denote vectors with entries in $\mathbb{Z}_q$. Bold fonts are still used to represent matrices/vectors with entries in R_q.

Proof System 0: Two RLWE samples with a common small secret

Let $A, B \in \mathbb{Z}_q^{n \times n}$ be the negacyclic matrices associated to multiplication by $a, b \in R_q$ respectively. Further, let $\vec{c}_1, \vec{c}_2 \in \mathbb{Z}_q^n$ be the coefficient vectors of $c_1, c_2 \in R_q$ respectively. The first proof aims to prove in zero knowledge, knowledge of a short solution $\vec{x} := (\vec{x}_1, \vec{x}_2, \vec{x}_3)$, where $\|\vec{x}\|_\infty \leq \sigma \cdot \sqrt{n}$ to the system

$$\vec{c}_1 = A \cdot \vec{x}_1 + \vec{x}_2,$$
$$\vec{c}_2 = B \cdot \vec{x}_1 + \vec{x}_3.$$

The security of our VOPRF uses a very special form of q for security due to the use of the 1D-SIS assumption. In particular, q is neither an integer permitting an NTT, nor a prime power. This is unfortunate because the state-of-the-art for proving zero knowledge of short solutions to linear equations use the fact that $x(x-1) = 0 \bmod q$ if and only if $x \in \{0, 1\}$ to prove that witness vectors have binary entries [141] (or utilise NTTs and similar algebraic relations for ternary entries [30]). As a result, we can either use Stern's protocol as described in [89], or rejection sampling techniques [94, 95] to perform this zero knowledge proof. However, due to the soundness gap suffered when using rejection sampling (i.e. the fact that the infinity norm of the extractable witness may be a small constant factor times larger than intended), one can imagine the use of the less efficient Stern's protocol for the sake of keeping our VOPRF security proofs conceptually simpler. In actual fact, as with all of the proof systems, the analysis and linear systems derived when applying Stern's protocol can be reused to show compatibility with the more efficient protocol of Beullens [24]. However, for simplicity, we focus on the abstract version of Stern's protocol since our main aim is to show feasibility rather than efficiency.

Proof System 1: Non-interactive proofs of PRF evaluations

At a high level, we can run Stern's protocol [136] $\mathcal{O}(\lambda)$ times in parallel and apply the Fiat-Shamir heuristic in the QROM. We present the abstract Stern's protocol itself in Figure 5.3 and highlight the sufficient requirements for the use of the abstract protocol here. This abstraction is both presented and proven to be a ZKAoK in [84] with respect to computationally binding commitments. It is also easy to see that if the commitment scheme is perfectly binding, then the protocol becomes a ZKPoK. We will assume the availability of a perfectly binding post-quantum commitment scheme throughout. Note that perfectly binding lattice-based commitment schemes do exist e.g. [23, 22]. For some set VALID and a matrix M representing a set of linear equations over the *integers* modulo a natural number (e.g. q), the abstraction of Stern's protocol allows a prover to argue knowledge of a solution $\vec{w} \in \mathsf{VALID}$ to a system $M \cdot \vec{w} = \vec{y} \bmod q$ in zero knowledge. In order to apply Stern's protocol, there must be a set of permutations $\Gamma = \{\Gamma_\phi : \phi \in \mathcal{S}\}$ acting on the *entries* of $\vec{w}$ such that both of the following key properties hold.

Key properties:

1. For every $\phi \in \mathcal{S}, \vec{w} \in \mathsf{VALID} \iff \Gamma_\phi(\vec{w}) \in \mathsf{VALID}$.
2. For every $\vec{w} \in \mathsf{VALID}$, the distribution of $\Gamma_\phi(\vec{w})$ (for $\phi \leftarrow \mathcal{S}$) is uniform over the set VALID.

Therefore, in order to apply the abstract Stern's protocol, we must rewrite our problem as a linear system of equations and describe a set VALID alongside a set of permutations Γ possessing the key properties above. The details of how this is done are presented in Section 5.6, but we now give a short high-level summary of the technique.

First note that we can compute $\boldsymbol{a}_x$ recursively by setting variables $\mathbf{B}_i \in R_q^{\ell \times \ell}$ for $i = L-1, \ldots, 0$ via $\mathbf{B}_{L-1} = G^{-1}(\boldsymbol{a}_{x_{L-1}})$, and $\mathbf{B}_i = G^{-1}(\boldsymbol{a}_{x_i} \cdot \mathbf{B}_{i+1})$ for $i = L-2, \ldots, 0$. Using this, we have $\boldsymbol{a}_x = \mathbf{G} \cdot \mathbf{B}_0$. We can therefore use the system $\mathbf{G} \cdot \mathbf{B}_i = \boldsymbol{a}_{x_i} \cdot \mathbf{B}_{i-1}$ to facilitate computation of $\boldsymbol{a}_x$ along with the equation $\boldsymbol{y}_x = \mathbf{G} \cdot \mathbf{B}_0 \cdot k + \boldsymbol{e}$

Abstract Stern Protocol [86]

Objective: P wishes to prove knowledge of $\vec{w} \in \mathbb{Z}_q^n \cap \mathsf{VALID}$ such that $M \cdot \vec{w} = \vec{y} \bmod q$ to V.

Commitment: If $M \in \mathbb{Z}_q^{m \times n}$, P with witness $\vec{w}$ samples $\phi \leftarrow \mathcal{S}$, $\vec{r} \leftarrow \mathbb{Z}_q^n$ and sets $\vec{z} = \vec{w} + \vec{r} \bmod q$. Then P samples randomness ρ_1, ρ_2, ρ_3 and sends $\mathsf{Cmt} = (C_1, C_2, C_3)$ where

- $C_1 = \mathsf{Com}\left(\phi, M \cdot \vec{r}; \rho_1\right)$
- $C_2 = \mathsf{Com}\left(\Gamma_\phi(\vec{r}); \rho_2\right)$
- $C_3 = \mathsf{Com}\left(\Gamma_\phi(\vec{z}); \rho_3\right)$

Challenge: V sends a challenge $Ch \leftarrow \{1, 2, 3\}$

Response: P sends RSP according to Ch as follows:

- If $Ch = 1$: $\text{RSP} = (\vec{t}, \vec{s}, \rho_2, \rho_3)$ where $\vec{t} = \Gamma_\phi(\vec{w})$ and $\vec{s} = \Gamma_\phi(\vec{r})$.
- If $Ch = 2$: $\text{RSP} = (\pi, \vec{u}, \rho_1, \rho_3)$ where $\pi = \phi$ and $\vec{u} = \vec{z}$.
- If $Ch = 3$:$\text{RSP} = (\psi, \vec{v}, \rho_1, \rho_2)$ where $\psi = \phi$ and $\vec{v} = \vec{r}$.

Verify: V outputs the bit b according to the following:

- If $Ch = 1$:
  $b = 1 \iff \left(\vec{t} \in \mathsf{VALID} \wedge C_2 = \mathsf{Com}\left(\vec{s}; \rho_2\right) \wedge C_3 = \mathsf{Com}\left(\vec{t} + \vec{s}; \rho_3\right)\right)$
- If $Ch = 2$:
  $b = 1 \iff \left(C_1 = \mathsf{Com}\left(\pi, M \cdot \vec{u} - \vec{y}; \rho_1\right) \wedge C_3 = \mathsf{Com}\left(\Gamma_\pi(\vec{u}); \rho_3\right)\right)$
- If $Ch = 3$:
  $b = 1 \iff \left(\mathsf{Com}\left(\psi, M \cdot \vec{v}; \rho_1\right) \wedge C_2 = \mathsf{Com}\left(\Gamma_\pi(\vec{v}); \rho_2\right)\right)$

Figure 5.3: Abstract Stern Protocol [86]

(where $\boldsymbol{e}$ represents rounding) to fully describe a PRF evaluation. However, the resulting system is over ring elements and is not linear in unknowns. To solve these issues, we simply replace ring multiplication by integer matrix-vector products and then linearise the resulting system using known techniques [85, 86]. At this point, we carefully describe the set VALID, noting the structure that linearisation/ring structure introduces. We also make use of special bit-decompositions to bound the infinity norms of valid solutions. From this, we use known techniques [85, 86] (extended to the ring setting) to describe Γ satisfying the key properties above. We once again remind the reader that the resulting linear system obtained can be used along with the relatively efficient protocol of Beullens [24].

Proof System 2: Non-interactive proofs of secret equivalence

Recall that we wish to prove existence of a solution to Equations (5.5). Note that $\boldsymbol{d}_x^1, \boldsymbol{d}_x^2$ from the protocol in Section 5.3 are vectors holding ℓ ring elements. Therefore, Equations (5.5) can be expressed as a system

$$c_i = a_i k + e_i, \quad i = 1, \ldots, 2 + 2\ell$$

where $\|e_1\|_\infty, \|e_2\|_\infty \leq \sigma \cdot \sqrt{n}$, $\|e_3\|_\infty, \ldots, \|e_{2+2\ell}\|_\infty \leq \sigma' \cdot \sqrt{n}$. In order to instantiate this proof system, we may use the abstract Stern protocol again. Note that in Section 5.5, we implicitly show how to prove knowledge of RLWE secrets. Therefore, using the same techniques, we can straight-forwardly obtain abstract Stern proofs for Proof System 2.

5.6 Abstract Stern Protocol for Proof System 1

In this section we outline how we rewrite the statement of Proof System 1 as a linear system of equations and describe a set VALID alongside a set of permutations Γ possessing the key properties for the abstract Stern protocol stated in the previous section.

5.6.1 (Randomised) PRF Evaluation and the ZK Relation.

Recall that G^{-1} is the non-linear binary decomposition operation, and $\mathbf{G}$ is the powers of two matrix that undoes G^{-1}. Also recall that in the query phase, the client computes the function $F'_{k;\boldsymbol{e}} : \{0,1\}^L \to R_q^{1\times\ell}$ where

$$F'_{k;\boldsymbol{e}}(x) = \boldsymbol{a}_{x_0} \cdot G^{-1}\left(\boldsymbol{a}_{x_1} \cdot G^{-1}\left(\boldsymbol{a}_{x_2} G^{-1}\left(\ldots\right)\right)\right) \cdot k + \boldsymbol{e} \bmod q. \tag{5.12}$$

Note that this function is similar to, but not exactly the same as the PRF F from Section 5.2.2. In particular, the function F' is a *randomised* version of F where the error $\boldsymbol{e}$ is not obtained in a deterministic fashion. Note, however, that the techniques of this section can be straight-forwardly adapted to prove the analogous relation that uses F instead of F' (see [86]). In terms of the function F', the language we are interested in providing a ZKAoK/ZKPoK for is

$$\begin{aligned}\mathsf{L} = \{(\boldsymbol{a}_0, \boldsymbol{a}_1, \boldsymbol{y}_1, \boldsymbol{y}_2) \in (R_q^{1\times\ell})^4 : \exists (s,t,x,\boldsymbol{e}_1,\boldsymbol{e}_2) \in (R)^2 \times \{0,1\}^L \times (R^{1\times\ell})^2 \\ \boldsymbol{y}_1 = F'_{k;\boldsymbol{e}_1}(x), \boldsymbol{y}_2 = F'_{k;\boldsymbol{e}_2}(x) \\ \|s\|_\infty, \|t\|_\infty \leq \beta_1, \\ \|\boldsymbol{e}_1\|_\infty, \|\boldsymbol{e}_2\|_\infty \leq \beta_2\}.\end{aligned}$$

To begin with, we can describe the computation of $F'_{s,\boldsymbol{e}_1}(x)$ and $F'_{t,\boldsymbol{e}_2}(x)$ recursively using

$$\begin{aligned}\mathbf{B}_{L-1} &= G^{-1}(\boldsymbol{a}_{x_{L-1}}) \\ \mathbf{B}_{L-2} &= G^{-1}(\boldsymbol{a}_{x_{L-2}} \cdot \mathbf{B}_{L-1}) \\ \mathbf{B}_{L-3} &= G^{-1}(\boldsymbol{a}_{x_{L-3}} \cdot \mathbf{B}_{L-2}) \\ &\vdots \\ \mathbf{B}_0 &= G^{-1}(\boldsymbol{a}_{x_0} \cdot \mathbf{B}_1) \\ F_{s;\boldsymbol{e}_1}(x) &= \mathbf{G} \cdot \mathbf{B}_0 \cdot s + \boldsymbol{e}_1 \\ F_{t,\boldsymbol{e}_2}(x) &= \mathbf{G} \cdot \mathbf{B}_0 \cdot t + \boldsymbol{e}_2\end{aligned}$$

where each equation is considered over the ring R_q. Importantly, $\mathbf{B}_i \in R_2^{\ell\times\ell}$ represent binary decompositions and $\boldsymbol{a} \in R_q^{1\times\ell}$.

5.6.2 Evaluation of F' as a System of Linear Equations.

However, the system of equations above is not linear since G^{-1} is not a linear operator. In the hope of deriving a linear system of equations that we can use Stern's protocol on, we first multiply by the linear operator $\mathbf{G} \in R_q^{1\times\ell}$ or equivalently $\boldsymbol{g}^T = (1, 2, \ldots, 2^{\ell-1}) \in R_q^{1\times\ell}$. In doing so, we can set $\boldsymbol{b}_0 = (\boldsymbol{g}^T \cdot \mathbf{B}_0) \in R_q^{\ell}$ to obtain

$$\begin{aligned}
\boldsymbol{g}^T \cdot \mathbf{B}_{L-1} &= \boldsymbol{a}_{x_{L-1}} && (5.13)\\
\boldsymbol{g}^T \cdot \mathbf{B}_{L-2} &= \boldsymbol{a}_{x_{L-2}} \cdot \mathbf{B}_{L-1}\\
\boldsymbol{g}^T \cdot \mathbf{B}_{L-3} &= \boldsymbol{a}_{x_{L-3}} \cdot \mathbf{B}_{L-2}\\
&\vdots\\
\boldsymbol{b}_0^T &= \boldsymbol{a}_{x_0} \cdot \mathbf{B}_1 && (5.14)\\
F'_{s;\boldsymbol{e}_1}(x) &= \boldsymbol{b}_0 \cdot s + \boldsymbol{e}_1\\
F'_{t;\boldsymbol{e}_2}(x) &= \boldsymbol{b}_0 \cdot t + \boldsymbol{e}_2.
\end{aligned}$$

We now wish to come up with a ZKPoK allowing to prove knowledge of $\{(\mathbf{B}_i)_{i=1}^{L-1}, \boldsymbol{b}_0, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2\}$ (where $s, t, \boldsymbol{e}_1, \boldsymbol{e}_2$ are short, and $\mathbf{B}_i \in R_2^{\ell\times\ell}$) satisfying the above system of linear equations.

5.6.3 Three Problems with the Linear System.

In order to use Stern's protocol, the witness must be a vector with entries $\mathbb{Z}_q$ that solves some publicly known linear system. Considering the current formulation, we have three initial problems to solve:

1. The $\mathbf{B}_i$'s are matrices, rather than vectors,
2. The "witness" $\{(\mathbf{B}_i)_{i=1}^{L-1}, \boldsymbol{b}_0, s, t, \boldsymbol{e}_1, \boldsymbol{e}_2\}$ consists of vectors/matrices with entries in R_q rather than $\mathbb{Z}_q$.
3. The system is quadratic in unknowns, rather than linear.

Solving the First Problem.

To get the unknowns $\mathbf{B}_i \in R_2^{\ell \times \ell}$ in vector-form rather than matrix-form, we can introduce some tensor products. For $i = 1, \ldots, L-1$, define $\boldsymbol{b}_i \in R_2^{\ell^2}$ to be the vector consisting of the columns of $\mathbf{B}_i$ stacked on top of each other. Inserting the appropriate tensor products, Equations (5.13)-(5.14) end up being of the form

$$(\mathbf{I}_\ell \otimes \boldsymbol{g}^T) \cdot \boldsymbol{b}_i = (\mathbf{I}_\ell \otimes \boldsymbol{a}_{x_i}) \cdot \boldsymbol{b}_{i+1},$$
$$\boldsymbol{b}_0 = (\mathbf{I}_\ell \otimes \boldsymbol{a}_{x_0}) \cdot \boldsymbol{b}_1.$$

Solving the Second Problem.

We would like to replace all multiplications in R_q by a matrix-vector multiplication over $\mathbb{Z}_q$. To do so we simply use the well known negacyclic matrices over $\mathbb{Z}_q$ that represent multiplication in R_q. We define $A_0 \in \mathbb{Z}_q^{n \times n\ell}$ (and A_1) to be the *horizontal* concatenation of the negacyclic matrices corresponding to the entries of $\boldsymbol{a}_0 \in R_q^{1\times\ell}$ (resp. $\boldsymbol{a}_1$). Furthermore, we define $S \in \mathbb{Z}_q^{n\times n}$ (and T) to be the negacyclic matrices representing $s \in R_q$ (resp. t). Note that this turns part of our witness back into a matrices, but we will show how to deal with this using the techniques of [85] later. Also, for $i = 0, \ldots, L-1$, let $\vec{b}_i$ be the vertical concatenation of the coefficients in the ring entries of $\boldsymbol{b}_i$ and let $\vec{e}_1$ (resp. $\vec{e}_2$) be the vertical concatenation of coefficients in the entries of $\boldsymbol{e}_1$ (resp. $\boldsymbol{e}_2$). Further, let $\vec{y}_1$ and $\vec{y}_2$ be the vertical concatenation of the coefficients in $F'_{s,\boldsymbol{e}_1}(x)$ and $F'_{t,\boldsymbol{e}_2}(x)$ respectively. Let $\vec{g}^T = (1, 2, \ldots, 2^{\ell-1}) \in \mathbb{Z}_q^{1\times\ell}$. Setting $G^\otimes = I_\ell \otimes (\vec{g}^T \otimes I_n)$, $A_{x_i}^\otimes = I_\ell \otimes A_{x_i} \in \mathbb{Z}_q^{n\ell \times n\ell^2}$ and $\vec{b}_L$ to be the binary vector with a 1 in each block of $n\ell$ entries (at the $(i-1)n + 1^{th}$ position in the i^{th} block), we end up with the following system of equations mod q:

$$G^\otimes \cdot \vec{b}_{L-1} = A_{x_{L-1}}^\otimes \vec{b}_L \tag{5.15}$$
$$G^\otimes \cdot \vec{b}_{L-2} = A_{x_{L-2}}^\otimes \cdot \vec{b}_{L-1}$$
$$G^\otimes \cdot \vec{b}_{L-3} = A_{x_{L-3}}^\otimes \cdot \vec{b}_{L-2} \tag{5.16}$$
$$\vdots$$
$$\vec{b}_0 = A_{x_0}^\otimes \cdot \vec{b}_1 \tag{5.17}$$
$$\vec{y}_1 = (I_\ell \otimes S) \cdot \vec{b}_0 + \vec{e}_1 \tag{5.18}$$
$$\vec{y}_2 = (I_\ell \otimes T) \cdot \vec{b}_0 + \vec{e}_2. \tag{5.19}$$

where $\vec{b}_0 \in \mathbb{Z}_q^{n\ell}$, $\vec{b}_i \in \{0,1\}^{n\ell^2}$ for $i \neq 0$ and S, T have small entries.

Solving the Third Problem.

We very briefly overview the techniques of [86] to indicate how one can linearise Equations (5.15) to (5.17). The idea is to represent $A_{x_i}^{\otimes} \cdot \vec{b}_{i+1}$ by writing

$$A_{x_i}^{\otimes} \cdot \vec{b}_{i+1} = [A_0^{\otimes} | A_1^{\otimes}] \cdot \begin{bmatrix} \bar{x_i} \cdot \vec{b}_{i+1} \\ x_i \cdot \vec{b}_{i+1}. \end{bmatrix}$$

In order to make use of this, we treat unknowns x_i and $\vec{b}_{i+1}$ together by considering the single unknown

$$\vec{\tilde{b}}_i = \begin{bmatrix} \bar{x_i} \cdot \vec{b}_{i+1} \\ x_i \cdot \vec{b}_{i+1} \end{bmatrix} \tag{5.20}$$

In doing so, Equations (5.15) - (5.16) end up being of the form

$$[G^{\otimes} | G^{\otimes}] \cdot \vec{\tilde{b}}_{i-1} = [A_0^{\otimes} | A_1^{\otimes}] \cdot \vec{\tilde{b}}_i.$$

where for $i = 1, \ldots, L-1$, valid solutions $\vec{\tilde{b}}_i$ are of the form given in (5.20) i.e. a binary vector where the top half or bottom half of entries are 0. Equation (5.17) becomes

$$\vec{b}_0 = [A_0^{\otimes} | A_1^{\otimes}] \cdot \vec{\tilde{b}}_1. \tag{5.21}$$

Now we turn our attention to Equations (5.18) and (5.19). The high level idea for obtaining equations linear in unknowns is the same. We essentially rewrite the equations in terms of a new single unknown that depends quadratically in the old unknowns and then take note of the structure that this induces on valid solutions. We only consider the term $(I_\ell \otimes S) \cdot \vec{b}_0$ from Equation (5.18) since the quadratic term in Equation (5.19) can be dealt with in exactly the same way. For the ring element $s = \sum_{i=0}^{n-1} s_i x^i \in R_q$ corresponding to $S \in \mathbb{Z}_q^{n \times n}$, it is clear that if we know the products $s_i \cdot (\vec{b}_0)_j$ for every $(i,j) \in \{0, \ldots, n-1\} \times \{1, \ldots, n\ell\}$, then we can calculate $(I_\ell \otimes S) \cdot \vec{b}_0$ since every entry will be a linear combination of these products. Therefore, letting $\vec{s} \in \mathbb{Z}_q^n$ be the coefficient vector of s, we can write $\vec{z}_s = \vec{b}_0 \otimes \vec{s}$ so that

$$(I_\ell \otimes S) \cdot \vec{b}_0 = Q \cdot \vec{z}_s \bmod q$$

where $Q \in \mathbb{Z}_2^{n\ell \times n^2 \ell}$ is some known constant matrix. Note that this methodology is the same as in [85] apart from the fact that Q here is defined using the structure

of R_q. It is also useful here to express $\vec{b}_0$ in terms of its binary decomposition vector. In particular, we define $\vec{\tilde{b}}_0 \in \{0,1\}^{n\ell^2}$ to be the vertical concatenation of the binary decomposition of entries in $\vec{b}_0$. We can also rewrite $\vec{s}$ using a special binary decomposition. In particular, set $\delta_j = \lfloor(\beta_1 + 2^{j-1})/2^j\rfloor$ for $j = 1, \ldots, \lfloor\log\beta_1\rfloor + 1$, and $D_{\beta_1} = I_n \otimes (\delta_1, \ldots, \delta_{\lfloor\log\beta_1\rfloor+1})$. As in [86], we can efficiently find a vector $\vec{s}' \in \{-1,0,1\}^{n(\lfloor\log\beta_1\rfloor+1)}$ such that $D_{\beta_1}\vec{s}' = \vec{s}$ for any $\vec{s} \in \{-\beta_1, \ldots, \beta_1\}^n$. In addition, $\sum_{i=1}^{\lfloor\log\beta_1\rfloor+1}\delta_i = \beta_1$, implying that $\|D_{\beta_1}\cdot\vec{s}\|_\infty \leq \beta_1$ for any $\vec{s} \in \{-1,0,1\}^{n(\lfloor\log\beta_1\rfloor+1)}$. Defining $H_q := (1, 2, \ldots, 2^{\ell-1}) \otimes I_{n\ell}$, we have that $H_q\vec{\tilde{b}}_0 = \vec{b}_0$. Similarly, defining $P := Q \cdot (H_q \otimes D_\beta)$ and $\vec{z}'_s := \vec{\tilde{b}}_0 \otimes \vec{s}'$ we can write

$$(I_\ell \otimes S)\cdot\vec{b}_0 = P\cdot\vec{z}'_s \bmod q.$$

Note that we can derive a similar equation for t. We can also define D_{β_2} similarly to D_{β_2} to decompose $\vec{e}_1, \vec{e}_2$ into trinary $\vec{e}'_1, \vec{e}'_2$.

5.6.4 The Final Linear System.

Finally, we arrive at the following system modulo q:

$$\begin{aligned}
[G^\otimes|G^\otimes]\cdot\vec{\tilde{b}}_{L-1} &= [A_0^\otimes|A_1^\otimes]\cdot\vec{\tilde{b}}_L \qquad (5.22)\\
[G^\otimes|G^\otimes]\cdot\vec{\tilde{b}}_{L-2} &= [A_0^\otimes|A_1^\otimes]\cdot\vec{\tilde{b}}_{L-1}\\
&\vdots\\
H_q\cdot\vec{\tilde{b}}_0 &= [A_0^\otimes|A_1^\otimes]\cdot\vec{\tilde{b}}_1\\
\vec{y}_1 &= P\cdot\vec{z}'_s + D_{\beta_2}\cdot\vec{e}'_1\\
\vec{y}_2 &= P\cdot\vec{z}'_t + D_{\beta_2}\cdot\vec{e}'_2 \qquad (5.23)
\end{aligned}$$

where valid solutions are such that:

- $\vec{\tilde{b}}_L = (1,0)^T \otimes \vec{c}$ or $(0,1)^T \otimes \vec{c}$ where for $\hat{c}_i = \overbrace{(0,\ldots,0}^{i-1},1,\overbrace{0,\ldots,0)}^{\ell-i}$,

$$\vec{c} = (\hat{c}_1\|\hat{c}_2,\|\ldots\|\hat{c}_\ell)^T \otimes \overbrace{(1,0,\ldots,0)^T}^{n} \qquad (5.24)$$

- for $i = 1, \ldots, L-1$, $\vec{\tilde{b}}_i \in \{0,1\}^{2n\ell^2}$ and either the first or second $n\ell^2$ entries are 0

- $\vec{\tilde{b}}_0 \in \{0,1\}^{n\ell^2}$
- $\vec{z}'_s = \vec{\tilde{b}}_0 \otimes \vec{s}'$ for some $\vec{s}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$
- $\vec{z}'_t = \vec{\tilde{b}}_0 \otimes \vec{t}'$ for some $\vec{t}' \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)}$
- $\vec{e}'_1, \vec{e}'_2 \in \{-1,0,1\}^{n\ell \cdot (\lfloor \log \beta_2 \rfloor + 1)}$

5.6.5 The Building Block Extensions and Permutations.

Now we will show how to use Stern's protocol to prove knowledge of a *valid* solution/witness

$$\vec{\psi} = \begin{bmatrix} \vec{\tilde{b}}_L \\ \vdots \\ \vec{\tilde{b}}_0 \\ \vec{\tilde{b}}_0 \otimes \vec{s}' \\ \vec{\tilde{b}}_0 \otimes \vec{t}' \\ \vec{e}'_1 \\ \vec{e}'_2 \end{bmatrix} \tag{5.25}$$

to the system $M \cdot \vec{\psi} = \vec{y}$ implicit in Equations (5.22)-(5.23). We do this in the standard way by extending the witness vector (while updating the system of equations), and then defining a set VALID along with a set of permutations Γ such that the two key properties from Section 5.5 hold. We begin by describing an extension and permutation for each small section of the witness.

Extension for $\vec{\tilde{b}}_L$

It turns out that we do not need to extend the part of the witness comprising $\vec{\tilde{b}}_L$. All we need to do is define the permutations indexed by bit $b \in \{0,1\}$, $\pi_b : \{0,1\}^{2n\ell^2} \to \{0,1\}^{2n\ell^2}$. Writing $\vec{v} = (\vec{v}_0, \vec{v}_1)$ where $\vec{v}_0, \vec{v}_1 \in \{0,1\}^{n\ell^2}$, we define π_b via the equation $\pi_b(\vec{v}) = (\vec{v}_b, \vec{v}_{\bar{b}})$. In words, this permutation either does nothing or switches a valid $\vec{\tilde{b}}_L$ to the other valid option according to the value of $b \in \{0,1\}$.

Extension for $\vec{\tilde{b}}_1, \ldots, \vec{\tilde{b}}_{L-1}$

Recalling that either the second or first half of entries in each of $\vec{\tilde{b}}_1, \ldots, \vec{\tilde{b}}_{L-1}$ is

0, we define the extension Ext_0 to act as follows on a vector $\vec{v} \in \{0,1\}^{2n'}$. Writing $\vec{v} = (\vec{v}_1, \vec{v}_2)$ where $\vec{v}_1, \vec{v}_2 \in \{0,1\}^{n'}$ and letting h be the hamming weight of $\vec{v}$, we define

$$\mathsf{Ext}_1(\vec{v}) = \begin{cases} (\vec{v}_1, \overbrace{1,\ldots,1}^{n'-h}, \overbrace{0,\ldots,0}^{h}, \vec{v}_2, \vec{v}_2) \text{ if } \vec{v}_2 = \vec{0}, \\ (\vec{v}_1, \vec{v}_1, \vec{v}_2, \underbrace{1,\ldots,1}_{n'-h}, \underbrace{0,\ldots,0}_{h}) \text{ if } \vec{v}_1 = \vec{0}. \end{cases}$$

Define the permutation on vector entries τ_σ indexed by any $\sigma \in \mathcal{S}_{n'}$ where $\mathcal{S}_{n'}$ is the symmetric group on n' elements as follows. Writing $\vec{v}_1 = (v_{1,1}, \ldots, v_{1,n'})$ and $\vec{v}_2 = (v_{2,1}, \ldots, v_{2,n'})$, we let $\tau_\sigma(\vec{v}) := (v_{1,\sigma(1)}, \ldots, v_{1,\sigma(n')}, v_{2,\sigma(1)}, \ldots, v_{2,\sigma(n')})$. The corresponding permutations accompanying Ext_1 are given by $\pi_b \circ \tau_\sigma$ for any $b \in \{0,1\}, \sigma \in \mathcal{S}_{n'}$.

Extension for $\vec{\tilde{b}}_0$

For a vector $\vec{v} = (v_1, \ldots, v_{n'}) \in \{0,1\}^{n'}$, we define

$$\mathsf{Ext}_2(\vec{v}) = (v_1, \bar{v}_1, v_2, \bar{v}_2, \ldots, v_{n'}, \bar{v}_{n'}) \in \{0,1\}^{2n'}.$$

The corresponding permutations $\rho_{\vec{d}} : \{0,1\}^{2n'} \to \{0,1\}^{2n'}$ are indexed by $\vec{d} \in \{0,1\}^{n'}$. For $\vec{w} = (w_{1,0}, w_{1,1}, w_{2,0}, w_{2,1}, \ldots, w_{n',0}, w_{n',1})$, we define

$$\rho_{\vec{d}}(\vec{w}) := (w_{1,d_1}, w_{1,\bar{d}_1}, \ldots, w_{n',d_{n'}}, w_{n',\bar{d}_{n'}})$$

. The crucial observation is that

$$\vec{w} = \mathsf{Ext}_2(\vec{v}) \iff \rho_{\vec{d}}(\vec{w}) = \mathsf{Ext}_2(\vec{v} \oplus \vec{d}).$$

Product extensions for $\vec{\tilde{b}}_0 \otimes \vec{s}'$, $\vec{\tilde{b}}_0 \otimes \vec{t}'$

In [85], an extension and permutation for products of two bits compatible with Stern's protocol is presented. Inspired by this, we first show an extension and permutation that can handle products between $c_1 \in \{0,1\}$ and $c_2 \in \{-1,0,1\}$. For $c \in \{1,2\}$, we use the notation $c_2^{+c} = c_2 + c \bmod 3$ and define

$$\mathsf{Ext}_3(c_1, c_2) := (c_1c_2, c_1c_2^{+1}, c_1c_2^{+2}, \bar{c}_1c_2, \bar{c}_1c_2^{+1}, \bar{c}_1c_2^{+2}) \in \{-1,0,1\}^6. \tag{5.26}$$

Let cyc denote the clockwise cyclic permutation on entries of a 3 dimensional vector. The corresponding building-block permutations are indexed by $b_1 \in \{0,1\}, b_2 \in$

$\{-1,0,1\}$, and are defined by

$$T^3_{b_1,b_2} : (\vec{v}_0, \vec{v}_1) \longmapsto (\mathsf{cyc}^{b_2}(\vec{v}_{b_1}), \mathsf{cyc}^{b_2}(\vec{v}_{\bar{b}_1}))$$

where $\vec{v}_i \in \{-1,0,1\}^3$ for $i = 0, 1$. This ensures that

$$\vec{v} = \mathsf{Ext}_3(c_1, c_2) \iff T^3_{b_1,b_2}(\vec{v}) = \mathsf{Ext}_3(c_1 \oplus b_1, c_2 \oplus_3 b_2) \tag{5.27}$$

where $\oplus_3$ denotes addition modulo 3 and $\vec{v} = (\vec{v}_0, \vec{v}_1) \in \{-1,0,1\}^6$. This permutation essentially one time pads both c_1 and c_2. We can generalise Ext_3 and $T^3_{(,)}$ to act on vectors $\vec{a} = (a_1, \ldots, a_{n'}) \in \{0,1\}^{n'}$ and $\vec{b} = (b_1, \ldots, b_{n''}) \in \{-1,0,1\}^{n''}$ as follows. Informally, the generalised extension $\mathsf{Ext}^{\otimes}_3$ is the vertical concatenation of the $\mathsf{Ext}_3((a_i, b_j))$ ranging over $i = 1, \ldots, n'$ and $j = 1, \ldots, n''$. More precisely,

$$\begin{aligned}\mathsf{Ext}_{3,\otimes}(\vec{a}, \vec{b}) =& \mathsf{Ext}_3(a_1, b_1) \| \mathsf{Ext}_3(a_1, b_2) \| \ldots \| \mathsf{Ext}_3(a_1, b_{n''}) \| \ldots \| \ldots \| \ldots \\ & \ldots \quad \mathsf{Ext}_3(a_{n'}, b_1) \| \mathsf{Ext}_3(a_{n'}, b_2) \| \ldots \| \mathsf{Ext}_3(a_{n'}, b_{n''}).\end{aligned}$$

Importantly, this generalised extension contains all entries arising in the tensor product $\vec{a} \otimes \vec{b}$, so can be considered as an extension of $\vec{a} \otimes \vec{b}$. The generalised permutations are indexed by $\vec{c} = (c_1, \ldots, c_{n'}) \in \{0,1\}^{n'}$, $\vec{d} = (d_1, \ldots, d_{n''}) \in \{-1,0,1\}^{n''}$ and are denoted $T^{3,\otimes}_{\vec{c},\vec{d}}$. Writing $\vec{v} = (\vec{v}_{1,1}, \ldots, \vec{v}_{1,n''}, \ldots, \vec{v}_{n',1}, \ldots, \vec{v}_{n',n''})$ where each $\vec{v}_{i,j} \in \{-1,0,1\}^6$, we define

$$\begin{aligned}T^{3,\otimes}_{\vec{c},\vec{d}}(\vec{v}) :=& (T^3_{c_1,d_1}(\vec{v}_{1,1}) \| T^3_{c_1,d_2}(\vec{v}_{1,2}) \| \ldots \| T^3_{c_1,d_{n''}}(\vec{v}_{1,n''}) \| \ldots \| \ldots \| \ldots \\ & \ldots \quad \| T^3_{c_{n'},d_1}(\vec{v}_{n',1}) \| T^3_{c_{n'},d_2}(\vec{v}_{n',2}) \| \ldots \| T^3_{c_{n'},d_{n''}}(\vec{v}_{n',n''})).\end{aligned}$$

Using these definitions, we have

$$\vec{v} = \mathsf{Ext}_{3,\otimes}(\vec{a}, \vec{b}) \iff T^{3,\otimes}_{\vec{c},\vec{d}}(\vec{v}) = \mathsf{Ext}_{3,\otimes}(\vec{a} \oplus \vec{c}, \vec{b} \oplus_3 \vec{d}). \tag{5.28}$$

Extension for $\vec{e}'_1, \vec{e}'_2$

Here we use the technique from [89]. For any $\vec{v} \in \{-1,0,1\}^{n'}$ with h_{-1}, h_0, h_1 entries equal to $-1, 0, 1$ respectively, we define the extension

$$\mathsf{Ext}'(\vec{v}) = (\vec{v} \| \overbrace{-1, \ldots, -1}^{n'-h_{-1}}, \overbrace{0, \ldots, 0}^{n'-h_0}, \overbrace{1, \ldots, 1}^{n'-h_1}).$$

Note that this outputs a vector in $\{-1,0,1\}^{3n'}$ with exactly n' entries that take each of the values $-1, 0, 1$. The corresponding permutations τ'_σ are indexed by $\sigma \in \mathcal{S}_{3n'}$ where $\mathcal{S}_{3n'}$ is the symmetric group over $3n'$ elements. For $\vec{w} = (w_1, \ldots, w_{3n'})$, the permutation τ'_σ is defined via $\tau'_\sigma(\vec{w}) = (w_{\sigma(1)}, \ldots, w_{\sigma(3n')})$.

5.6.6 The Full Extension, Permutation and Valid Set.

Using the extensions in the previous section, we extend the witness of the form (5.25) to the following:

$$\vec{\psi}' = \begin{bmatrix} \vec{\tilde{b}}_L \\ \mathsf{Ext}_1(\vec{\tilde{b}}_{L-1}) \\ \vdots \\ \mathsf{Ext}_1(\vec{\tilde{b}}_1) \\ \mathsf{Ext}_2(\vec{\tilde{b}}_0) \\ \mathsf{Ext}_{3,\otimes}(\vec{\tilde{b}}_0 \otimes \vec{s}') \\ \mathsf{Ext}_{3,\otimes}(\vec{\tilde{b}}_0 \otimes \vec{t}') \\ \mathsf{Ext}'(\vec{e}'_1) \\ \mathsf{Ext}'(\vec{e}'_2) \end{bmatrix} \tag{5.29}$$

This forces us to update the system of equations to $M' \cdot \vec{\psi}' = \vec{y}$ where the columns of M make up a subset of the columns of M'. We now conclude by defining the set VALID and set of permutations Γ satisfying the key properties from Section 5.5.

We will say that $\vec{v} = (\vec{v}_1, \vec{v}_2, \ldots, \vec{v}_L, \vec{v}_{L+1}, \vec{v}_{L+2}, \vec{v}_{L+3}, \vec{v}_{L+4}, \vec{v}_{L+5}) \in \mathsf{VALID}$ if and only if:

- $\vec{v}_1 \in \{0,1\}^{2n\ell^2}$ is either $(1,0)^T \otimes \vec{c}$ or $(0,1)^T \otimes \vec{c}$ where $\vec{c}$ is defined in Equation (5.24).
- $\vec{v}_2, \ldots, \vec{v}_L \in \{0,1\}^{4n\ell^2}$ have Hamming weight $n\ell^2$ with either the first half or second half of entries all 0.
- $\vec{v}_{L+1} \in \{0,1\}^{2n\ell^2}$ and is consistent with the form of vector output by Ext_2.
- Letting $\vec{w}$ be the valid preimage of $\vec{v}_{L+1}$ under Ext_2, $\vec{v}_{L+2}$ and $\vec{v}_{L+3} \in \{-1,0,1\}^{6n^2\ell^2(\log\lfloor\beta_1\rfloor+1)}$ are of the form $\mathsf{Ext}_{3,\otimes}(\vec{w}, \vec{s}'')$ and $\mathsf{Ext}_{3,\otimes}(\vec{w}, \vec{t}'')$ respectively for some $s'', t'' \in \{-1,0,1\}^{n(\lfloor\log\beta_1\rfloor+1)}$
- $\vec{v}_{L+4}, \vec{v}_{L+5} \in \{-1,0,1\}^{3n\ell(\lfloor\log\beta_2\rfloor+1)}$ have an equal number of $-1, 0, 1$ entries

The permutation set Γ is

$$\left\{ \Gamma_{\boldsymbol{\phi}} \;\middle|\; \begin{array}{l} \boldsymbol{\phi} = (\phi_1, \phi_{2,1}, \phi_{2,2}, \ldots, \phi_{L,1}, \phi_{L,2}, \phi_{L+1}, \phi_{L+2}, \ldots, \phi_{L+5}), \\ \phi_1, \phi_{2,1}, \phi_{3,1}, \ldots, \phi_{L,1} \in \{0,1\} \\ \phi_{2,2}, \ldots, \phi_{L,2} \in \mathcal{S}_{n\ell^2} \\ \phi_{L+1} \in \{0,1\}^{n\ell^2} \\ \phi_{L+2}, \phi_{L+3} \in \{-1,0,1\}^{n(\lfloor \log \beta_1 \rfloor + 1)} \\ \phi_{L+4}, \phi_{L+5} \in \mathcal{S}_{3n\ell(\lfloor \log \beta_2 \rfloor + 1)} \end{array} \right\} \tag{5.30}$$

where

$$\Gamma_{\boldsymbol{\phi}} : \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_{L+1} \\ \vec{v}_{L+2} \\ \vec{v}_{L+3} \\ \vec{v}_{L+4} \\ \vec{v}_{L+5} \end{bmatrix} \longmapsto \begin{bmatrix} \pi_{\phi_1}(\vec{v}_1) \\ \pi_{\phi_{2,1}} \circ \tau_{\phi_{2,2}}(\vec{v}_2) \\ \vdots \\ \pi_{\phi_{L,1}} \circ \tau_{\phi_{L,2}}(\vec{v}_L) \\ \rho_{\phi_{L+1}}(\vec{v}_{L+1}) \\ T^{3,\otimes}_{\phi_{L+1},\phi_{L+2}}(\vec{v}_{L+2}) \\ T^{3,\otimes}_{\phi_{L+1},\phi_{L+3}}(\vec{v}_{L+3}) \\ \tau'_{\phi_{L+4}}(\vec{v}_{L+4}) \\ \tau'_{\phi_{L+5}}(\vec{v}_{L+5}) \end{bmatrix} \tag{5.31}$$

Finally, we note that $\Gamma_{\boldsymbol{\phi}}$ acts on elements of VALID in the following way:

$$\mathsf{VALID} \ni \begin{bmatrix} \vec{v}_1 \\ \vec{v}_2 \\ \vdots \\ \vec{v}_L \\ \mathsf{Ext}_2(\vec{\nu}_{L+1}) \\ \mathsf{Ext}_{3,\otimes}(\vec{\nu}_{L+1} \otimes \vec{s}'') \\ \mathsf{Ext}_{3,\otimes}(\vec{\nu}_{L+1} \otimes \vec{t}'') \\ \vec{v}_{L+4} \\ \vec{v}_{L+5} \end{bmatrix} \mapsto \begin{bmatrix} \pi_{\phi_1}(\vec{v}_1) \\ \pi_{\phi_{2,1}} \circ \tau_{\phi_{2,2}}(\vec{v}_2) \\ \vdots \\ \pi_{\phi_{L,1}} \circ \tau_{\phi_{L,2}}(\vec{v}_L) \\ \mathsf{Ext}_2(\vec{\nu}_{L+1} \oplus \phi_{L+1}) \\ \mathsf{Ext}_{3,\otimes}((\vec{\nu}_{L+1} \oplus \phi_{L+1}) \otimes (\vec{s}'' \oplus_3 \phi_{L+2}) \\ \mathsf{Ext}_{3,\otimes}((\vec{\nu}_{L+1} \oplus \phi_{L+1}) \otimes (\vec{t}'' \oplus_3 \phi_{L+3}) \\ \tau'_{\phi_{L+4}}(\vec{v}_{L+4}) \\ \tau'_{\phi_{L+5}}(\vec{v}_{L+5}) \end{bmatrix} \in \mathsf{VALID}.$$

The above implies both the first and second key properties required for the abstract version of Stern's protocol. In particular, for the second property, note that a random permutation in Γ essentially one-time pads arguments in the Ext sections of the witness and randomly permutes the remaining sections in a structure preserving manner.

Bibliography

[1] Michel Abdalla, Pierre-Alain Fouque, Vadim Lyubashevsky, and Mehdi Tibouchi. Tightly-secure signatures from lossy identification schemes. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 572–590. Springer, Heidelberg, April 2012.

[2] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Lattice basis delegation in fixed dimension and shorter-ciphertext hierarchical IBE. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 98–115. Springer, Heidelberg, August 2010.

[3] Martin Albrecht and Carlos Cid. Cold boot key recovery by solving polynomial systems with noise. In Javier Lopez and Gene Tsudik, editors, *ACNS 11*, volume 6715 of *LNCS*, pages 57–72. Springer, Heidelberg, June 2011.

[4] Martin R. Albrecht. On dual lattice attacks against small-secret LWE and parameter choices in HElib and SEAL. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part II*, volume 10211 of *LNCS*, pages 103–129. Springer, Heidelberg, April / May 2017.

[5] Martin R. Albrecht, Benjamin R. Curtis, Amit Deo, Alex Davidson, Rachel Player, Eamonn W. Postlethwaite, Fernando Virdia, and Thomas Wunderer. Estimate all the LWE, NTRU schemes! In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 351–367. Springer, Heidelberg, September 2018.

[6] Martin R. Albrecht and Amit Deo. Large modulus ring-LWE $\geq$ module-LWE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 267–296. Springer, Heidelberg, December 2017.

[7] Martin R. Albrecht, Amit Deo, and Kenneth G. Paterson. Cold boot attacks on ring and module LWE keys under the NTT. *IACR TCHES*, 2018(3):173–213, 2018. https://tches.iacr.org/index.php/TCHES/article/view/7273.

[8] Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the expected cost of solving uSVP and applications to LWE. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 297–322. Springer, Heidelberg, December 2017.

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

[10] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange - A new hope. In Thorsten Holz and Stefan Savage, editors, *25th USENIX Security Symposium, USENIX Security 16*, pages 327–343. USENIX Association, 2016.

[11] Yoshinori Aono, Phong Q. Nguyen, and Yixin Shen. Quantum lattice enumeration and tweaking discrete pruning. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 405–434. Springer, Heidelberg, December 2018.

[12] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast cryptographic primitives and circular-secure encryption based on hard learning problems. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 595–618. Springer, Heidelberg, August 2009.

[13] Hayo Baan, Sauvik Bhattacharya, Óscar García-Morchón, Ronald Rietman, Ludo Tolhuizen, Jose-Luis Torre-Arce, and Zhenfei Zhang. Round2: KEM and PKE based on GLWR. Cryptology ePrint Archive, Report 2017/1183, 2017. https://eprint.iacr.org/2017/1183.

[14] László Babai. On lovász'lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986.

[15] Shi Bai and Steven D. Galbraith. Lattice decoding attacks on binary LWE. In Willy Susilo and Yi Mu, editors, *ACISP 14*, volume 8544 of *LNCS*, pages 322–337. Springer, Heidelberg, July 2014.

[16] Wojciech Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(1):625–635, 1993.

[17] Gustavo Banegas, Paulo S. L. M. Barreto, Edoardo Persichetti, and Paolo Santini. Designing efficient dyadic operations for cryptographic applications. Cryptology ePrint Archive, Report 2018/650, 2018. `https://eprint.iacr.org/2018/650`.

[18] Abhishek Banerjee and Chris Peikert. New and improved key-homomorphic pseudorandom functions. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part I*, volume 8616 of *LNCS*, pages 353–370. Springer, Heidelberg, August 2014.

[19] Abhishek Banerjee, Chris Peikert, and Alon Rosen. Pseudorandom functions and lattices. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 719–737. Springer, Heidelberg, April 2012.

[20] Elad Barkan and Eli Biham. Conditional estimators: An effective attack on A5/1. In Bart Preneel and Stafford Tavares, editors, *SAC 2005*, volume 3897 of *LNCS*, pages 1–19. Springer, Heidelberg, August 2006.

[21] Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. In Dan Boneh, editor, *CRYPTO 2003*, volume 2729 of *LNCS*, pages 600–616. Springer, Heidelberg, August 2003.

[22] Carsten Baum, Ivan Damgård, Vadim Lyubashevsky, Sabine Oechsner, and Chris Peikert. More efficient commitments from structured lattice assumptions. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 368–385. Springer, Heidelberg, September 2018.

[23] Fabrice Benhamouda, Stephan Krenn, Vadim Lyubashevsky, and Krzysztof Pietrzak. Efficient zero-knowledge proofs for commitments from learning with errors over rings. In Günther Pernul, Peter Y. A. Ryan, and Edgar R. Weippl, editors, *ESORICS 2015, Part I*, volume 9326 of *LNCS*, pages 305–325. Springer, Heidelberg, September 2015.

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

[25] Marc Beunardeau, Aisling Connolly, Rémi Géraud, and David Naccache. On the hardness of the Mersenne low Hamming ratio assumption. Cryptology ePrint Archive, Report 2017/522, 2017. https://eprint.iacr.org/2017/522.

[26] Richard E Blahut. *Theory and practice of error control codes*. Addison-Wesley, 1983.

[27] Dan Boneh, Kevin Lewi, Hart William Montgomery, and Ananth Raghunathan. Key homomorphic PRFs and their applications. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 410–428. Springer, Heidelberg, August 2013.

[28] Xavier Bonnetain, Akinori Hosoyamada, María Naya-Plasencia, Yu Sasaki, and André Schrottenloher. Quantum attacks without superposition queries: the offline simon's algorithm. Cryptology ePrint Archive, Report 2019/614, 2019. https://eprint.iacr.org/2019/614.

[29] Jonathan Bootle, Claire Delaplace, Thomas Espitau, Pierre-Alain Fouque, and Mehdi Tibouchi. LWE without modular reduction and improved side-channel attacks against BLISS. In Thomas Peyrin and Steven Galbraith, editors, *ASIACRYPT 2018, Part I*, volume 11272 of *LNCS*, pages 494–524. Springer, Heidelberg, December 2018.

[30] Jonathan Bootle, Vadim Lyubashevsky, and Gregor Seiler. Algebraic techniques for short(er) exact lattice-based zero-knowledge proofs. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 176–202. Springer, Heidelberg, August 2019.

[31] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. In Shafi Goldwasser, editor, *ITCS 2012*, pages 309–325. ACM, January 2012.

[32] Zvika Brakerski, Adeline Langlois, Chris Peikert, Oded Regev, and Damien Stehlé. Classical hardness of learning with errors. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 575–584. ACM Press, June 2013.

[33] Zvika Brakerski and Vinod Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In Yevgeniy Dodis and Jesper Buus Nielsen, editors, *TCC 2015, Part II*, volume 9015 of *LNCS*, pages 1–30. Springer, Heidelberg, March 2015.

[34] Zvika Brakerski and Vinod Vaikuntanathan. Circuit-ABE from LWE: Unbounded attributes and semi-adaptive security. Cryptology ePrint Archive, Report 2016/118, 2016. http://eprint.iacr.org/2016/118.

[35] Leon Groot Bruinderink and Peter Pessl. Differential fault attacks on deterministic lattice signatures. Cryptology ePrint Archive, Report 2018/355, 2018. https://eprint.iacr.org/2018/355.

[36] Ran Canetti and Yilei Chen. Constraint-hiding constrained PRFs for NC^1 from LWE. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 446–476. Springer, Heidelberg, April / May 2017.

[37] Sanjit Chatterjee, Neal Koblitz, Alfred Menezes, and Palash Sarkar. Another look at tightness II: Practical issues in cryptography. Cryptology ePrint Archive, Report 2016/360, 2016. http://eprint.iacr.org/2016/360.

[38] Hao Chen, Kim Laine, and Rachel Player. Simple encrypted arithmetic library - SEAL v2.1. Cryptology ePrint Archive, Report 2017/224, 2017. http://eprint.iacr.org/2017/224.

[39] Yu-Ao Chen and Xiao-Shan Gao. Quantum algorithms for boolean equation solving and quantum algebraic attack on cryptosystems. Cryptology ePrint Archive, Report 2018/008, 2018. https://eprint.iacr.org/2018/008.

[40] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better lattice security estimates. In Dong Hoon Lee and Xiaoyun Wang, editors, *ASIACRYPT 2011*, volume 7073 of *LNCS*, pages 1–20. Springer, Heidelberg, December 2011.

[41] Jung Hee Cheon, Duhyeong Kim, Joohee Lee, and Yongsoo Song. Lizard: Cut off the tail! A practical post-quantum public-key encryption from LWE and LWR. In Dario Catalano and Roberto De Prisco, editors, *SCN 18*, volume 11035 of *LNCS*, pages 160–177. Springer, Heidelberg, September 2018.

[42] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Izabachène. TFHE: Fast fully homomorphic encryption over the torus. Cryptology ePrint Archive, Report 2018/421, 2018. https://eprint.iacr.org/2018/421.

[43] Ronald Cramer, Léo Ducas, and Benjamin Wesolowski. Short stickelberger class relations and application to ideal-SVP. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *EUROCRYPT 2017, Part I*, volume 10210 of *LNCS*, pages 324–348. Springer, Heidelberg, April / May 2017.

[44] Dana Dachman-Soled, Huijing Gong, Mukul Kulkarni, and Aria Shahverdi. On the leakage resilience of ideal-lattice based public key encryption. Cryptology ePrint Archive, Report 2017/1127, 2017. https://eprint.iacr.org/2017/1127.

[45] Dana Dachman-Soled, Huijing Gong, Mukul Kulkarni, and Aria Shahverdi. Partial key exposure in ring-lwe-based cryptosystems: Attacks and resilience. Cryptology ePrint Archive, Report 2018/1068, 2018. https://eprint.iacr.org/2018/1068.

[46] Alex Davidson, Ian Goldberg, Nick Sullivan, George Tankersley, and Filippo Valsorda. Privacy pass: Bypassing internet challenges anonymously. *PoPETs*, 2018(3):164–180, 2018.

[47] Koen de Boer, Léo Ducas, Stacey Jeffery, and Ronald de Wolf. Attacks on the AJPS Mersenne-based cryptosystem. In Tanja Lange and Rainer Steinwandt, editors, *Post-Quantum Cryptography*, pages 101–120, Cham, 2018. Springer.

[48] Jintai Ding, Tsuyoshi Takagi, Xinwei Gao, and Yuntao Wang. Ding key exchange. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[49] Yevgeniy Dodis, Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In Daniele Micciancio, editor, *TCC 2010*, volume 5978 of *LNCS*, pages 361–381. Springer, Heidelberg, February 2010.

[50] Jelle Don, Serge Fehr, Christian Majenz, and Christian Schaffner. Security of the Fiat-Shamir transformation in the quantum random-oracle model. In

Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 356–383. Springer, Heidelberg, August 2019.

[51] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal Gaussians. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 40–56. Springer, Heidelberg, August 2013.

[52] Thomas Espitau, Pierre-Alain Fouque, Benoît Gérard, and Mehdi Tibouchi. Side-channel attacks on BLISS lattice-based signatures: Exploiting branch tracing against strongSwan and electromagnetic emanations in microcontrollers. In Bhavani M. Thuraisingham, David Evans, Tal Malkin, and Dongyan Xu, editors, *ACM CCS 2017*, pages 1857–1874. ACM Press, October / November 2017.

[53] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. `http://eprint.iacr.org/2012/144`.

[54] Amos Fiat and Adi Shamir. How to prove yourself: Practical solutions to identification and signature problems. In Andrew M. Odlyzko, editor, *CRYPTO'86*, volume 263 of *LNCS*, pages 186–194. Springer, Heidelberg, August 1987.

[55] The FPLLL Development Team. FPLLL, a lattice reduction library. Available at `https://github.com/fplll/fplll`, 2017.

[56] The FPYLLL Development Team. FPYLLL, a Python interface for FPLLL. Available at `https://github.com/fplll/fpylll`, 2018.

[57] Michael J. Freedman, Yuval Ishai, Benny Pinkas, and Omer Reingold. Keyword search and oblivious pseudorandom functions. In Joe Kilian, editor, *TCC 2005*, volume 3378 of *LNCS*, pages 303–324. Springer, Heidelberg, February 2005.

[58] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th ACM STOC*, pages 197–206. ACM Press, May 2008.

[59] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *CRYPTO 2013, Part I*, volume 8042 of *LNCS*, pages 75–92. Springer, Heidelberg, August 2013.

[60] Oded Goldreich. *Foundations of cryptography: volume 1, basic tools*. Cambridge university press, 2007.

[61] Shafi Goldwasser, Yael Tauman Kalai, Chris Peikert, and Vinod Vaikuntanathan. Robustness of the learning with errors assumption. In Andrew Chi-Chih Yao, editor, *ICS 2010*, pages 230–240. Tsinghua University Press, January 2010.

[62] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 545–554. ACM Press, June 2013.

[63] Lov K Grover. A fast quantum mechanical algorithm for database search. *arXiv preprint quant-ph/9605043*, 1996.

[64] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: Cold boot attacks on encryption keys. In Paul C. van Oorschot, editor, *USENIX Security 2008*, pages 45–60. USENIX Association, July / August 2008.

[65] J. Alex Halderman, Seth D. Schoen, Nadia Heninger, William Clarkson, William Paul, Joseph A. Calandrino, Ariel J. Feldman, Jacob Appelbaum, and Edward W. Felten. Lest we remember: cold-boot attacks on encryption keys. *Commun. ACM*, 52(5):91–98, 2009.

[66] Wilko Henecka, Alexander May, and Alexander Meurer. Correcting errors in RSA private keys. In Tal Rabin, editor, *CRYPTO 2010*, volume 6223 of *LNCS*, pages 351–369. Springer, Heidelberg, August 2010.

[67] Nadia Heninger and Hovav Shacham. Reconstructing RSA private keys from random key bits. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 1–17. Springer, Heidelberg, August 2009.

[68] Jeff Hoffstein, Jill Pipher, John M. Schanck, Joseph H. Silverman, William Whyte, and Zhenfei Zhang. Choosing parameters for NTRUEncrypt. Cryptology ePrint Archive, Report 2015/708, 2015. http://eprint.iacr.org/2015/708.

[69] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In Marc Joye, editor, *CT-RSA 2003*, volume 2612 of *LNCS*, pages 122–140. Springer, Heidelberg, April 2003.

[70] Stanislaw Jarecki, Aggelos Kiayias, and Hugo Krawczyk. Round-optimal password-protected secret sharing and T-PAKE in the password-only model. In Palash Sarkar and Tetsu Iwata, editors, *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 233–253. Springer, Heidelberg, December 2014.

[71] Stanislaw Jarecki, Aggelos Kiayias, Hugo Krawczyk, and Jiayu Xu. Highly-efficient and composable password-protected secret sharing (or: How to protect your bitcoin wallet online). In *EuroS&P*, pages 276–291. IEEE, 2016.

[72] Stanislaw Jarecki, Hugo Krawczyk, and Jiayu Xu. OPAQUE: An asymmetric PAKE protocol secure against pre-computation attacks. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part III*, volume 10822 of *LNCS*, pages 456–486. Springer, Heidelberg, April / May 2018.

[73] Stanislaw Jarecki and Xiaomin Liu. Efficient oblivious pseudorandom function with applications to adaptive OT and secure computation of set intersection. In Omer Reingold, editor, *TCC 2009*, volume 5444 of *LNCS*, pages 577–594. Springer, Heidelberg, March 2009.

[74] Takayasu Kaida, Satoshi Uehara, and Kyoki Imamura. An algorithm for the k-error linear complexity of sequences over $GF(p^m)$ with period p^n, p a prime. *Information and Computation*, 151(1-2):134–147, 1999.

[75] Abdel Alim Kamal and Amr M Youssef. Applications of SAT solvers to AES key recovery from decayed key schedule images. In *Emerging Security Information Systems and Technologies (SECURWARE), 2010 Fourth International Conference on*, pages 216–220. IEEE, 2010.

[76] Ravi Kannan. Improved algorithms for integer programming and related lattice problems. In *15th ACM STOC*, pages 193–206. ACM Press, April 1983.

[77] Sriram Keelveedhi, Mihir Bellare, and Thomas Ristenpart. Dupless: Server-aided encryption for deduplicated storage. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, pages 179–194, Washington, D.C., 2013. USENIX.

[78] Elena Kirshanova, Erik Mårtensson, Eamonn W. Postlethwaite, and Subhayan Roy Moulik. Quantum algorithms for the approximate k-list problem and their application to lattice sieving. Cryptology ePrint Archive, Report 2019/1016, 2019. https://eprint.iacr.org/2019/1016.

[79] Thijs Laarhoven. *Search problems in cryptography.* PhD thesis, PhD thesis, Eindhoven University of Technology, 2015. http://www.thijs.com, 2015.

[80] Thijs Laarhoven, Michele Mosca, and Joop Van De Pol. Finding shortest lattice vectors faster using quantum search. *Designs, Codes and Cryptography*, 77(2-3):375–400, 2015.

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

[82] Adeline Langlois, Damien Stehlé, and Ron Steinfeld. GGHLite: More efficient multilinear maps from ideal lattices. In Phong Q. Nguyen and Elisabeth Oswald, editors, *EUROCRYPT 2014*, volume 8441 of *LNCS*, pages 239–256. Springer, Heidelberg, May 2014.

[83] Arjen Klaas Lenstra, Hendrik Willem Lenstra, and László Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261(4):515–534, 1982.

[84] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Signature schemes with efficient protocols and dynamic group signatures from lattice assumptions. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 373–403. Springer, Heidelberg, December 2016.

[85] Benoît Libert, San Ling, Fabrice Mouhartem, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for matrix-vector relations and lattice-based group encryption. In Jung Hee Cheon and Tsuyoshi Takagi, editors, *ASIACRYPT 2016, Part II*, volume 10032 of *LNCS*, pages 101–131. Springer, Heidelberg, December 2016.

[86] Benoît Libert, San Ling, Khoa Nguyen, and Huaxiong Wang. Zero-knowledge arguments for lattice-based PRFs and applications to E-cash. In Tsuyoshi Takagi and Thomas Peyrin, editors, *ASIACRYPT 2017, Part III*, volume 10626 of *LNCS*, pages 304–335. Springer, Heidelberg, December 2017.

[87] Yehuda Lindell. How to simulate it - A tutorial on the simulation proof technique. Cryptology ePrint Archive, Report 2016/046, 2016. `http://eprint.iacr.org/2016/046`.

[88] Richard Lindner and Chris Peikert. Better key sizes (and attacks) for LWE-based encryption. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 319–339. Springer, Heidelberg, February 2011.

[89] San Ling, Khoa Nguyen, Damien Stehlé, and Huaxiong Wang. Improved zero-knowledge proofs of knowledge for the ISIS problem, and applications. In Kaoru Kurosawa and Goichiro Hanaoka, editors, *PKC 2013*, volume 7778 of *LNCS*, pages 107–124. Springer, Heidelberg, February / March 2013.

[90] Mingjie Liu and Phong Q. Nguyen. Solving BDD by enumeration: An update. In Ed Dawson, editor, *CT-RSA 2013*, volume 7779 of *LNCS*, pages 293–309. Springer, Heidelberg, February / March 2013.

[91] Qipeng Liu and Mark Zhandry. Revisiting post-quantum Fiat-Shamir. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part II*, volume 11693 of *LNCS*, pages 326–355. Springer, Heidelberg, August 2019.

[92] Seth Lloyd. Quantum algorithm for solving linear systems of equations. In *APS March Meeting Abstracts*, 2010.

[93] Xianhui Lu, Yamin Liu, Dingding Jia, Haiyang Xue, Jingnan He, and Zhenfei Zhang. LAC. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[94] Vadim Lyubashevsky. Fiat-Shamir with aborts: Applications to lattice and factoring-based signatures. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 598–616. Springer, Heidelberg, December 2009.

[95] Vadim Lyubashevsky. Lattice signatures without trapdoors. In David Pointcheval and Thomas Johansson, editors, *EUROCRYPT 2012*, volume 7237 of *LNCS*, pages 738–755. Springer, Heidelberg, April 2012.

[96] Vadim Lyubashevsky, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Peter Schwabe, Gregor Seiler, and Damien Stehle. CRYSTALS-DILITHIUM. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[97] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *EUROCRYPT 2010*, volume 6110 of *LNCS*, pages 1–23. Springer, Heidelberg, May / June 2010.

[98] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. A toolkit for ring-LWE cryptography. In Thomas Johansson and Phong Q. Nguyen, editors, *EUROCRYPT 2013*, volume 7881 of *LNCS*, pages 35–54. Springer, Heidelberg, May 2013.

[99] James Massey. Shift-register synthesis and BCH decoding. *IEEE transactions on Information Theory*, 15(1):122–127, 1969.

[100] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems - a cryptograhic perspective*, volume 671 of *The Kluwer international series in engineering and computer science*. Springer, 2002.

[101] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Comput.*, 37(1):267–302, 2007.

[102] Daniele Micciancio and Oded Regev. Lattice-based cryptography in post quantum cryptography, bernstein dj, buchmann j., dahmen e, 2009.

[103] Daniele Micciancio and Michael Walter. Fast lattice point enumeration with minimal overhead. In Piotr Indyk, editor, *26th SODA*, pages 276–294. ACM-SIAM, January 2015.

[104] National Institute of Standards and Technology. Submission requirements and evaluation criteria for the Post-Quantum Cryptography standardization process. `http://csrc.nist.gov/groups/ST/post-quantum-crypto/documents/call-for-proposals-final-dec-2016.pdf`, December 2016.

[105] Jürgen Neukirch. *Algebraic Number Theory*, volume 322 of *Grundlehren der mathematischen Wissenschaften*. Springer-Verlag Berlin Heidelberg, 1999.

[106] Kenneth G. Paterson, Antigoni Polychroniadou, and Dale L. Sibborn. A coding-theoretic approach to recovering noisy RSA keys. In Xiaoyun Wang and Kazue Sako, editors, *ASIACRYPT 2012*, volume 7658 of *LNCS*, pages 386–403. Springer, Heidelberg, December 2012.

[107] Kenneth G. Paterson and Ricardo Villanueva-Polanco. Cold boot attacks on NTRU. In Arpita Patra and Nigel P. Smart, editors, *INDOCRYPT 2017*, volume 10698 of *LNCS*, pages 107–125. Springer, Heidelberg, December 2017.

[108] Chris Peikert. A decade of lattice cryptography. Cryptology ePrint Archive, Report 2015/939, 2015. `http://eprint.iacr.org/2015/939`.

[109] Chris Peikert and Zachary Pepin. Algebraically structured LWE, revisited. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 1–23. Springer, Heidelberg, December 2019.

[110] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of Ring-LWE for any ring and modulus. In *STOC 2017*, 2017.

[111] Chris Peikert and Sina Shiehian. Privately constraining and programming PRFs, the LWE way. In Michel Abdalla and Ricardo Dahab, editors, *PKC 2018, Part II*, volume 10770 of *LNCS*, pages 675–701. Springer, Heidelberg, March 2018.

[112] Chris Peikert and Sina Shiehian. Noninteractive zero knowledge for NP from (plain) learning with errors. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 89–114. Springer, Heidelberg, August 2019.

[113] Bertram Poettering and Dale L. Sibborn. Cold boot attacks in the discrete logarithm setting. In Kaisa Nyberg, editor, *CT-RSA 2015*, volume 9048 of *LNCS*, pages 449–465. Springer, Heidelberg, April 2015.

[114] David Pointcheval and Jacques Stern. Security proofs for signature schemes. In Ueli M. Maurer, editor, *EUROCRYPT'96*, volume 1070 of *LNCS*, pages 387–398. Springer, Heidelberg, May 1996.

[115] Thomas Poppelmann, Erdem Alkim, Roberto Avanzi, Joppe Bos, Leo Ducas, Antonio de la Piedra, Peter Schwabe, and Douglas Stebila. NewHope. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

[116] Thomas Pornin and Thomas Prest. More efficient algorithms for the NTRU key generation using the field norm. In Dongdai Lin and Kazue Sako, editors, *PKC 2019, Part II*, volume 11443 of *LNCS*, pages 504–533. Springer, Heidelberg, April 2019.

[117] Robert Primas, Peter Pessl, and Stefan Mangard. Single-trace side-channel attacks on masked lattice-based encryption. In Wieland Fischer and Naofumi Homma, editors, *CHES 2017*, volume 10529 of *LNCS*, pages 513–533. Springer, Heidelberg, September 2017.

[118] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th ACM STOC*, pages 84–93. ACM Press, May 2005.

[119] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56(6), 2009.

[120] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[121] Miruna Rosca, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. Middle-product learning with errors. In Jonathan Katz and Hovav Shacham, editors, *CRYPTO 2017, Part III*, volume 10403 of *LNCS*, pages 283–297. Springer, Heidelberg, August 2017.

[122] Miruna Rosca, Damien Stehlé, and Alexandre Wallet. On the ring-LWE and polynomial-LWE problems. In Jesper Buus Nielsen and Vincent Rijmen, editors, *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 146–173. Springer, Heidelberg, April / May 2018.

[123] Markku-Juhani O. Saarinen. HILA5. Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.

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

[125] Claus-Peter Schnorr. Lattice reduction by random sampling and birthday methods. In Helmut Alt and Michel Habib, editors, *STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science*, volume 2607 of *Lecture Notes in Computer Science*, pages 145–156. Springer, 2003.

[126] A. Schönhage and V. Strassen. Schnelle multiplikation großer zahlen. *Computing*, 7(3):281–292, Sep 1971.

[127] Peter Schwabe, Roberto Avanzi, Joppe Bos, Leo Ducas, Eike Kiltz, Tancrede Lepoint, Vadim Lyubashevsky, John M. Schanck, Gregor Seiler, and Damien Stehle. CRYSTALS-KYBER. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[128] Minhye Seo, Jong Hwan Park, Dong Hoon Lee, Suhri Kim, and Seung-Joon Lee. EMBLEM and R.EMBLEM. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[129] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM review*, 41(2):303–332, 1999.

[130] Daniel R Simon. On the power of quantum computation. *SIAM journal on computing*, 26(5):1474–1483, 1997.

[131] Nigel P. Smart, Martin R. Albrecht, Yehuda Lindell, Emmanuela Orsini, Valery Osheter, Kenny Paterson, and Guy Peer. LIMA. Technical report, National Institute of Standards and Technology, 2017. available at `https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions`.

[132] Mark Stamp and Clyde F Martin. An algorithm for the k-error linear complexity of binary sequences with period 2^n. *IEEE Transactions on Information Theory*, 39(4):1398–1401, 1993.

[133] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *EUROCRYPT 2011*, volume 6632 of *LNCS*, pages 27–47. Springer, Heidelberg, May 2011.

[134] Damien Stehlé, Ron Steinfeld, Keisuke Tanaka, and Keita Xagawa. Efficient public key encryption based on ideal lattices. In Mitsuru Matsui, editor, *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 617–635. Springer, Heidelberg, December 2009.

[135] William Stein et al. *Sage Mathematics Software Version 8.1.* The Sage Development Team, 2017. http://www.sagemath.org.

[136] Jacques Stern. A new identification scheme based on syndrome decoding. In Douglas R. Stinson, editor, *CRYPTO'93*, volume 773 of *LNCS*, pages 13–21. Springer, Heidelberg, August 1994.

[137] Alex Tsow. An improved recovery algorithm for decayed AES key schedule images. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *SAC 2009*, volume 5867 of *LNCS*, pages 215–230. Springer, Heidelberg, August 2009.

[138] Tim van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[139] Yang Wang and Mingqiang Wang. Module-lwe versus ring-lwe, revisited. Cryptology ePrint Archive, Report 2019/930, 2019. https://eprint.iacr.org/2019/930.

[140] Franz Winkler. *Polynomial Algorithms in Computer Algebra*. Texts & Monographs in Symbolic Computation. Springer, 1996.

[141] Rupeng Yang, Man Ho Au, Zhenfei Zhang, Qiuliang Xu, Zuoxia Yu, and William Whyte. Efficient lattice-based zero-knowledge arguments with standard soundness: Construction and applications. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part I*, volume 11692 of *LNCS*, pages 147–175. Springer, Heidelberg, August 2019.

[142] Yunlei Zhao, Zhengzhong Jin, Boru Gong, and Guangye Sui. KCL (pka OKCN/AKCN/CNKE). Technical report, National Institute of Standards and Technology, 2017. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-1-submissions.