University of California
Los Angeles

Candidate Multilinear Maps

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Sanjam Garg

2013

Abstract of the Dissertation

Candidate Multilinear Maps

by

Sanjam Garg
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2013
Professor Rafail Ostrovsky, Co-chair
Professor Amit Sahai, Co-chair

In this thesis, we describe plausible lattice-based constructions with properties that approximate the sought-after multilinear maps in hard-discrete-logarithm groups. The security of our constructions relies on seemingly hard problems in ideal lattices, which can be viewed as extensions of the assumed hardness of the NTRU function.

These new constructions radically enhance our tool set and open a floodgate of applications. We present a survey of these applications.

The dissertation of Sanjam Garg is approved.

Benny Sudakov

Eli Gafni

Amit Sahai, Committee Co-chair

Rafail Ostrovsky, Committee Co-chair

University of California, Los Angeles

2013

To my parents…

Table of Contents

Acknowledgments

Foremost, I would like to express my sincere gratitude to my advisors, Rafail Ostrovsky and Amit Sahai for their continuous support throughout my PhD. Rafi and Amit have very different styles of research and this served as an ideal learning experience for me. Rafi's breadth of knowledge and wealth of perspective have helped me shape my own. Amit helped me ungarble my garbled research ideas and with this slowly I have learnt to do it myself. Reasoning with him, taught me how to think.

I was very fortunate to have abundant opportunities of interacting with and learning from Yuval Ishai. His insightful opinions about even my own research, made conversations with him highly enlightening (and many a times made me feel highly ignorant).

This thesis is based on a joint work with Craig Gentry and Shai Halevi. I would like to thank them for introducing me to the beautiful area of lattices and particularly for patiently answering my stupid questions. I have learnt a lot in the process. I would like to thank Alice Silverberg for her technical questions that have helped improve the quality of this thesis. Finally I would like to thanks my dissertation committee members Eli Gafni and Benny Sudakov for their helpful comments on this work.

I am highly thankful to Tal Rabin and the entire cryptography group at IBM T.J. Watson research center – David Cash, Craig Gentry, Shai Halevi, Charanjit Jutla, Hugo Krawczyk, Mariana Rayokava and Daniel Wichs for hosting me there for a summer and making it an amazing experience. I would also like to thank my fellow intern Nir Bitansky for making this summer fun.

I would like to thank Yuval Ishai and Eyal Kushelvitz for hosting me in Technion for a summer. Outside of work Yuval took me to the best restaurants in Haifa and made sure that I had a wonderful time. I am also thankful to Ariel Gabizon, Daniel Genkin, Sigurd Meldgaard and Anat Paskin for making my stay in Haifa fun.

I am highly thankful to Masayuki Abe and Tatsuaki Okamoto for hosting me for a summer in NTT, Japan and giving me the opportunity to learn from them. I would like to thank everyone in the NTT Crypto group and especially Sherman Chow, Claudio Orlandi, Saho Uchida and Berkant Ustaoglu for making my stay in Japan memorable.

I also had the immense pleasure of collaborating with and learning from Nir Bitansky, Elette Boyle, Nishanth Chandran, Vipul Goyal, Yael Kalai, Eyal Kushelvitz, Ivan Visconti, Brent Waters and Daniel Wichs. I would like to thank them for that.

I would like to thank Raghav Bhaskar and Satya Lokam for hosting me for multiple internships at Microsoft Research India and helping me take my first steps as a researcher in Theoretical Computer Science. I thank them for their continued encouragement throughout my PhD career.

Probably the deepest mark on my life was made by my fellow grad students. My life at UCLA, personally or professionally would not have been the same if it wasn't for Chongwon Cho, Ran Gelles, Abhishek Jain, Abishek Kumarasubramanian, Hemanta Maji, Omkant Pandey, Alan Royatman and Akshay Wadia. I would also like to thank Claudio Orlandi

and Ivan Visconti for being great friends. Finally, I would like to thank my friends and my colleagues – Shweta Agrawal, Paul Bun, Nishanth Chandran, Divya Gupta, Bhavna Kathuri, Dakshita Khurana, Chen-Kuei (James) Lee, Anat Paskin, Vanishree Rao, Bhavani Shankar Garikapati, Alessandra Scafuro, Michael Schindler, Hakan Seyalioglu, Brian Tagiku and Vassilias Zikas, at UCLA for making grad school such a wonderful experience.

Vita

2008	B.Tech. in Computer Science and Engineering, Indian Institute of Technology Delhi.
2008	TCS Best B.Tech. Project Award.
2009	Chancellor's Fellowship, UCLA.
2013	Outstanding Graduating Ph.D. Student Award, UCLA.

Publications

Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai and Brent Waters. Attribute Based Encryption for Circuits from Multilinear Maps. In Ran Canetti and Juan Garay, editors, *Advances in Cryptology – CRYPTO (2) 2013*, *Lecture Notes in Computer Science*, pages 479–499, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany.

Elette Boyle, Sanjam Garg, Abhishek Jain, Yael Tauman Kalai and Amit Sahai. Secure Computation Against Adaptive Auxiliary Information. In Ran Canetti and Juan Garay, editors, *Advances in Cryptology – CRYPTO (1) 2013*, *Lecture Notes in Computer Science*, pages 316–334, Santa Barbara, CA, USA, August 18–22, 2013. Springer, Berlin, Germany.

Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness Encryption and Its Applications. In Dan Boneh, Tim Roughgarden and Joan Feigenbaum, editors, *45th Annual ACM Symposium on Theory of Computing*, pages 467–476, Palo Alto, CA, June 1–4, 2013. ACM Press.

Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate Multilinear Maps from Ideal Lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17, Athens, Greece, May 26–30, 2013. Springer, Berlin, Germany.

Nir Bitansky, Dana Dachman-Soled, Sanjam Garg, Abhishek Jain, Yael Tauman Kalai, Adriana López-Alt and Daniel Wichs. Why ”Fiat-Shamir for Proofs” Lacks a Proof. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 182-201, Tokyo, Japan, March 3-6, 2013. Springer, Berlin, Germany.

Sanjam Garg, Abishek Kumarasubramanian, Rafail Ostrovsky, and Ivan Visconti. Impossibility Results for Static Input Secure Computation. In Reihaneh Safavi-Naini and Ran

Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 424–442, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany.

Sanjam Garg and Amit Sahai. Adaptively Secure Multi-party Computation with Dishonest Majority. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, volume 7417 of *Lecture Notes in Computer Science*, pages 105–123, Santa Barbara, CA, USA, August 19–23, 2012. Springer, Berlin, Germany.

Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Concurrently Secure Computation in Constant Rounds. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 99–116, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany.

Sanjam Garg, Rafail Ostrovsky, Ivan Visconti, and Akshay Wadia. Resettable Statistical Zero Knowledge. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 494–511, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Berlin, Germany.

Sanjam Garg, Abhishek Jain, and Amit Sahai. Leakage-Resilient Zero Knowledge. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 297–315, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

Sanjam Garg, Vanishree Rao, Amit Sahai, Dominique Schröder, and Dominique Unruh. Round Optimal Blind Signatures. In Phillip Rogaway, editor, *Advances in Cryptology – CRYPTO 2011*, volume 6841 of *Lecture Notes in Computer Science*, pages 630–648, Santa Barbara, CA, USA, August 14–18, 2011. Springer, Berlin, Germany.

Sanjam Garg, Vipul Goyal, Abhishek Jain, and Amit Sahai. Bringing People of Different Beliefs Together to do UC. In Yuval Ishai, editor, *TCC 2011: 8th Theory of Cryptography Conference*, volume 6597 of *Lecture Notes in Computer Science*, pages 311–328, Providence, RI, USA, March 28–30, 2011. Springer, Berlin, Germany.

Sanjam Garg, Abishek Kumarasubramanian, Amit Sahai, and Brent Waters. Building Efficient Fully Collusion-Resilient Traitor Tracing and Revocation Schemes. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10: 17th Conference on Computer and Communications Security*, pages 121–130, Chicago, Illinois, USA, October 4–8, 2010. ACM Press.

Sanjam Garg, Raghav Bhaskar, and Satyanarayana V. Lokam. Improved Bounds on Security Reductions for Discrete Log based Signatures. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 93–107, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Berlin, Germany.

Sanjam Garg and Huzur Saran. Anti-DDOS Virtualized Operating System. In *Proceedings of the The Third International Conference on Availability, Reliability and Security, ARES 2008, March 4-7, 2008, Technical University of Catalonia, Barcelona, Spain*, pages 667–674. IEEE Computer Society, 2008.

Michael LeMay, George Gross, Carl A. Gunter, and Sanjam Garg. Unified Architecture for Large-Scale Attested Metering. In *40th Hawaii International International Conference on Systems Science (HICSS-40 2007), CD-ROM / Abstracts Proceedings, 3-6 January 2007, Waikoloa, Big Island, HI, USA*, page 115. IEEE Computer Society, 2007.

CHAPTER 1

Introduction

The aim of cryptography is to design primitives and protocols that withstand adversarial behavior. Information theoretic cryptography, how-so-ever desirable, is extremely restrictive and most non-trivial cryptographic tasks are known to be information theoretically impossible. In order to realize sophisticated cryptographic primitives, we forgo information theoretic security and assume limitations on what can be efficiently computed. In other words we attempt to build secure systems conditioned on some computational intractability assumption such as – factoring [RSA78], discrete log [Knu97], decisional Diffe-Hellman [DH76], learning with errors [Reg05] and many more (see [Ver13]).

Last decade has seen a push towards using structured assumptions such as the ones based on bilinear maps, for realizing sophisticated cryptographic goals otherwise considered impossible according to folklore. For example, bilinear pairings have been used to design ingenious protocols for tasks such as one-round three-party key exchange [Jou00], identity-based encryption [BF01], and non-interactive zero-knowledge proofs [GOS06]. By now the applications of bilinear maps have become too numerous to name.

Boneh and Silverberg [BS03] showed that cryptographic groups equipped with multilinear maps would have even more interesting applications, including one-round multi-party key exchange and very efficient broadcast encryption. However they presented strong evidence that such maps should be hard to construct. In particular, they attempted to construct multilinear maps from abelian varieties (extending known techniques for constructing bilinear maps), but identified serious obstacles, and concluded that "such maps might have to either come from outside the realm of algebraic geometry, or occur as 'unnatural' computable maps arising from geometry." Since then, the persistent absence of cryptographically useful multilinear maps has not stopped researchers from proposing applications of them. For example, Rückert and Schröder [RS09] use multilinear maps to construct efficient aggregate and verifiably encrypted signatures without random oracles. Papamanthou, Tamassia and

Triandopoulos [PTT10] show that “compact” multilinear maps give very efficient authenticated data structures. Recently, Rothblum [Rot13] used multilinear maps to construct a counterexample to the conjecture that all bit-encryption schemes are [CL01, BRS03] circularly secure (secure when bit-encryptions of the secret key are also given out).

1.1 Our Results

In this work [GGH13a, GGH12] we put forth new plausible lattice-based constructions with properties that approximate the sought after multilinear maps. The multilinear analog of the decision Diffie-Hellman problem appears to be hard in our construction, and this allows for their use in cryptography. These construction open doors to a providing solutions (see Section 2 for details) to a number of important open problems.

Functionality. Our multilinear maps are approximate in the sense that they are “noisy.” Furthermore they are bounded to a polynomial degree. For very high degree, in our maps, the “noisiness” overwhelms the signal, somewhat like for ciphertexts in *somewhat homomorphic encryption* [Gen09a] schemes. In light of their noisiness, one could say that our multilinear maps are indeed “unnatural” computable maps arising from geometry. As a consequence, our multilinear maps differ quite substantially from the “ideal” multilinear maps envisioned by Boneh and Silverberg[BS03].

The boundedness of our encodings has interesting consequences, both positive and negative. On the positive side, it hinders an attack based on Boneh and Lipton’s subexponential algorithm for solving the discrete logarithm in black box fields [BL96]. This attack cannot be used to solve the “discrete log” problem in our setting, since their algorithm requires exponentiations with exponential degree. On the negative size, the dependence between the degree and parameter-size prevents us from realizing applications such as the ones envisioned by [PTT10] because they need “compact” maps. Similarly, so far we were not able to use our maps to realize Rothblum’s counterexample to the circular security of bit encryption conjecture [Rot13]. That counterexample requires degree that is polynomial, but a polynomial that is always just out of our reach of our parameters.[1]

Security. The security of the multilinear-DDH problem in our constructions relies on new hardness assumptions, and we provide an extensive cryptanalysis to validate these assumptions. To make sure that our constructions are not “trivially” insecure, we prove that our constructions are secure against adversaries that merely run an arithmetic straight-line [Kal85a, Kal85b] program.

We also analyze our constructions with respect to the best known averaging, algebraic and lattice attacks. Many of these attacks have been published before [CS97, HKL$^+$00, Gen01,

[1] Note that our original multilinear maps were insufficient for these applications but however one can use obfuscation [GGH$^+$13b] along with fully homomorphic encryption to realize special multilinear maps that at least heuristically will suffice for these applications.

GS02, Szy03, HGS04, NR06, NR09, DN12b] in the context of cryptanalysis of the NTRU [HPS01, HHGP+03] and GGH [GGH97] signature scheme. We also present new attacks on *principal* ideal lattices, which arise in our constructions, that are more efficient than (known) attacks on general ideal lattices. Our constructions remain secure against all of the attacks that we present, both old and new.

Finally we note that some problems that are believed hard relative to contemporary bilinear maps are easy with our construction (see Section 7.5).

1.2 Brief Overview

In his breakthrough result, Gentry [Gen09a] constructed a *fully-homomorphic encryption* scheme that enabled arbitrary computation on encrypted data without being able to decrypt. However for many applications, the ability to perform arbitrary computation on encrypted data along with the ability to check if two ciphertexts encrypt the same message is essential. In his scheme, Gentry relied on "noise" to hide messages. The presence of noise, which helps hide messages without restricting arbitrary computation on them, seems to be in conflict with the goal of equality checking. In our constructions we overcome this obstacle by introducing techniques that enable equality testing even in the presence of noise. Here we present an overview of our construction.

Our constructions work in polynomial rings and use principal ideals in these rings (and their associated lattices). In a nutshell, an instance of our construction has a secret short ring element $\mathbf{g} \in R$, generating a principal ideal $\mathcal{I} = \langle \mathbf{g} \rangle \subset R$. In addition, it has an integer parameter q and another secret $\mathbf{z} \in R/qR$, which is chosen at random (and hence is not small).

We think of a term like g^x in a discrete-log system as an "encoding" of the "plaintext exponent" x. In our case the role of the "plaintext exponents" is played by the elements in $R/\mathcal{I}$ (i.e. cosets of $\mathcal{I}$), and we "encode" them via division by $\mathbf{z}$ in R_q. In a few more details, our system provides many levels of encoding, where a level-i encoding of the coset $\boldsymbol{e}_{\mathcal{I}} = \boldsymbol{e} + \mathcal{I}$ is an element of the form $\boldsymbol{c}/\mathbf{z}^i \bmod q$ where $\boldsymbol{c} \in \boldsymbol{e}_{\mathcal{I}}$ is short. It is easy to see that such encodings can be both added and multiplied, so long as the numerators remain short. More importantly, we show that it is possible to publish a "zero testing parameter" that enables to test if two elements encode the same coset at a given level, without violating security (e.g., it should still be hard to compute x from an encoding of x at higher levels). Namely, we add to the public parameters an element of the form $\mathbf{p}_{zt} = \boldsymbol{h} \cdot \boldsymbol{z}^{\kappa}/\mathbf{g} \bmod q$ for a not-too-large $\boldsymbol{h}$, where κ is the level of multilinearity. We show that multiplying an encoding of zero (at the κ^{th} level) by $\mathbf{p}_{zt}$ (mod q) yields a small element, while multiplying an encoding of a non-zero by $\mathbf{p}_{zt}$ (mod q) yields a large element. Hence we can distinguish zero from non-zero, and by subtraction we can distinguish two encodings of the same element from encodings of two different elements.

Our schemes are somewhat analogous to graded algebras, hence we sometimes call them *graded encoding schemes*. Our schemes are quite flexible, and for example can be modified to support the analog of asymmetric maps by using several different $\boldsymbol{z}$'s. On the other hand, other variants such as composite-order groups turn out to be insecure with our encodings (at least when implemented in a straightforward manner).

Other related work. Building upon our constructions Coron, Lepoint and Tibouchi [CLT13] provide an alternate construction of multilinear maps that works over the integers instead of ideal lattices, similar to the fully homomorphic encryption scheme of [vDGHV10]. The security of these constructions also relies on new assumptions.

1.3 Organization

We define formally our notion of a "approximate" multilinear maps which we call *graded encoding schemes* (termed after the notion of graded algebra), as well an abstract notion of our main hardness assumption (which is a multilinear analog of DDH) in Chapter 3. In Chapter 3 we restrict ourselves to the "symmetric setting" and then later in Appendix A we extend our definition to the "asymmetric" setting.

Then in Chapter 4 we provide some background on number theory and lattices necessary for understanding our construction and the security analysis. Our construction is presented in Chapter 6 and a high level security analysis provided in Chapter 7. We provide details on the cryptanalysis tools used and developed in this work (needed for Chapter 7) in Chapter 9. Additional number theory background useful for understanding this chapter is provided in Chapter 8.

Finally, as an example application of our multilinear maps we provide a construction of one-round multi-party key-exchange protocol in Chapter 10. Since their introduction multilinear maps have subsequently been used for realizing many new applications. A survey of all these applications is presented in Chapter 2.

CHAPTER 2

Survey of Applications

Albeit noisy, our multilinear maps radically enhance our tool set and open a floodgate of applications. For example, our multilinear maps provide as a special case a new candidate for bilinear maps that can be used to compile a countless number of applications based on bilinear maps to ones based on lattice assumptions. One-round multi-party key-exchange is another classical example. Diffie and Hellman in their seminal paper [DH76] provided the first construction of a one-round two-party key-exchange protocol which was then generalized to the three party setting by Joux [Jou00] using Weil and Tate pairings. Boneh and Silverberg [BS03] showed how this result could be extended to get a one-round n-party key-exchange protocol if multilinear maps existed. Our approximate multilinear maps suffice for instantiating this construction giving the first realization of this primitive. In Chapter 10 we provide details on this construction.

Our candidate construction of multilinear maps through a sequence of works have enabled realization of many cryptographic goals otherwise considered impossible according to folklore. This progress has ultimately led us to candidate constructions [GGH+13b] of general purpose *program obfuscation*, a fundamental concept in cryptography. Program obfuscation first formalized in [BGI+01, BGI+12], aims to make a computer program "unintelligible" while preserving its functionality. Researchers have contemplated many applications of general-purpose obfuscation, at least as far back as the work of Diffie and Hellman in 1976.[1] We will present the development of these ideas chronologically.

[1] Diffie and Hellman suggested the use of general-purpose obfuscation to convert private-key cryptosystems to public-key cryptosystems.

2.1 How flexible can we make access to encrypted data?

Starting with Access Control. Enabling encryption by arbitrary parties motivated the invention of public key encryption [DH76, RSA78]. However, enabling fine-grained decryption capabilities has remained an elusive goal [Sha85, SW05, GPSW06]. Shamir [Sha85] proposed the problem of non-interactively associating identities with encrypted data, and later Sahai and Waters [SW05] asked if an encrypter at the time of encryption can non-interactively embed any arbitrary decryption policy into his ciphertext. So far, the realizations of this primitive, referred to as *attribute based encryption*, were limited to access-control policies expressed by formulas. In [GGH$^+$13c] we showed how multilinear maps could be used to overcome these barriers and provided a construction that allows for arbitrary access-control policies. Concurrent and independent of this work Gorbnov et al. [GVW13] provided a solution without using our multilinear maps. This result is fascination as it relies only on the sub-exponential harness of the learning with errors (LWE) assumption.

Limits of Access Control – Witness Encryption. Encryption in all its myriad flavors has always been imagined with some known recipient in mind. But, what if the intended recipient of the message is not known and may never be known to the encrypter? For example, consider the task of encrypting to someone who knows a solution to a crossword puzzle that appeared in the *The New York Times*. Or, in general, a solution to some NP search problem which he might know or might acquire over a period of time. The encrypter on the other hand may even be unaware of the existence of a solution.

In [GGSW13] we proposed the concept of *witness encryption* which captures this intuition and realized it based on our noisy multilinear maps. Witness Encryption is closely related to the notion of computational secret sharing for NP-complete access structures, first posed by Rudich in 1989 [Rud89] (see [Bei11]). As observed by Rudich, this primitive already suffices for converting private-key cryptosystems to public-key ones.

Witness encryption has found applications elsewhere as well. Most prominently, Goldwasser et al. [GKP$^+$13] used (a variant of) witness encryption for constructing a variant of attribute-based encryption scheme for polynomial-time Turing machines, where the sizes of secret keys depend only on the size of the Turing machine (rather than its runtime). Furthermore in these constructions, the decryption algorithm has an input-specific runtime rather than worst-case runtime (at the price of revealing this runtime).

Computation in addition to access control – Functional Encryption. All primitive described above enabled encrypters with the ability to specify who can decrypt. However at the same time these tools do not provide for a mechanism to specific what a decrypter can learn. A decrypter learns either the entire message or nothing about it. Going further one could ask questions that combine non-interactively computing on encrypted data with its access management (or *functional encryption*) [BSW11, O'N10]. More specifically, in functional encryption, ciphertexts encrypt inputs x and keys are issued for functions f. The striking feature of this system is that given an encryption of x, the key corresponding to f

can be used to obtain $f(x)$ but nothing else about x. Furthermore, *any arbitrary* collusion of key holders relative to many functions f_i does not yield any more information about x beyond what is "naturally revealed" by each of them individually (i.e. $f_i(x)$ for all i). Prior work on functional encryption has been extremely limited in power, with the state of the art roughly limited to the inner-product construction of Katz et al. [KSW08].[2] Again using multilinear maps, in a recent work we [GGH+13b] resolved this long standing open problem giving a construction of functional encryption for general circuits.[3,4]

2.2 Program Obfuscation

Computing on encrypted data and revealing specific functions of it already has the flavor of *program obfuscation*, first studied formally by Barak et al. [BGI+01, BGI+12]. Despite its potential for far-reaching applications, positive results for obfuscation have largely been limited to relatively simple classes of functions such as point functions [Can97, CMR98, LPS04, Wee05, CD08, BC10], testing hyperplane membership [CRV10] and a few other simple programs [HRSV07, HMLS07, Had10, CCV12]. Multilinear maps have helped change this landscape dramatically:

- **Indistinguishability Obfuscation.** Multilinear maps have been used to construct new candidate constructions for a general purpose obfuscator [GGH+13b] satisfying the *indistinguishability obfuscation* notion. An indistinguishability obfuscator [BGI+01], denoted $i\mathcal{O}$, for a class of circuits $\mathcal{C}$ guarantees that given two *equivalent* circuits C_1 and C_2 (in the sense that they compute the same function) from the class $\mathcal{C}$, the two distribution of obfuscations $i\mathcal{O}(C_1)$ and $i\mathcal{O}(C_2)$ should be computationally indistinguishable.

  Goldwasser et al. [GR07], provide strong philosophical argument supporting the meaningfulness of this notion. In particular they show that (efficiently computable) indistinguishability obfuscators achieve the notion of Best-Possible Obfuscation: Informally, a best-possible obfuscator guarantees that its output hides as much about the input circuit as any other circuit (of a certain size).

- **Virtual Black-Box Obfuscation.** *Virtual black box* obfuscation [BGI+01] (VBB in short) is the strongest notion of obfuscation considered in the literature. This concept requires that the obfuscated program behaves like a "black-box," in the sense that it should not leak information about the program except its input output behaviour. Multilinear Maps have been used to realize VBB obfuscation for functions such as conjunctions [BR13b] and dynamic point function [GGHR13].

[2] However, there are constructions that achieve only limited-collusion notions [SS10, GVW12, GKP+12, GKP+13] of security.

[3] We note that the [GGH+13b] construction gets a weaker indistinguishability notion of security for functional encryption. However this can be upgraded to natural simulation-based definitions of security using the work of De Caro et al. [CIJ+13].

[4] The latest version of the paper builds functional encryption from indistingushability obfuscation but we note that historically speaking these results are were actually obtained in the opposite order.

Our inability to provide more general results can be explained by the negative results of [BGI+01], who showed that there exist families of "unobfuscatable" functions for which the VBB definition is impossible to achieve *in the plain model*. However this result does not apply to the setting of generic multilinear attacks, in which case the VBB notion can actually be realized [BR13a, BR13c, BGK+13]. These works provide evidence that no *algebraic* attacks (that respect multilinear maps) against these candidate constructions leak anything beyond what could be leaked in a black-box manner and provide heuristic evidence that these obfuscation mechanisms offer strong security for "natural" functions.

Other applications of Indistinguishability Obfuscation. Indistinguishability Obfuscation has been used in surprisingly unrelated settings (we refer the reader to [SW13] for a thorough survey) and has helped achieve many new feasibility results:

- **Deniable Encryption.** Deniable encryption, a primitive introduced by Canetti et al. [CDNO97], requires that a sender forced into revealing to the adversary its message and randomness, should be able to convincingly provide "fake" randomness that can explain any alternative message that it would like to pretend that it sent. All schemes for this in the literature requires some kind of pre-planning by the party that must later issue a denial. In a recent work, using indistinguishability obfuscation Sahai et al. [SW13] construct the first scheme that does not rely on pre-planning.

- **Round Optimal Multiparty Secure Computation.** One fundamental complexity measure of an MPC protocol is its *round complexity*. Asharov et al. [AJLA+12] recently constructed the first three-round protocol for general MPC in the CRS model. Using indistinguishability obfuscation [GGHR13] we show how the same result can be achieved with only two rounds of communication.

2.3 Other Applications

Constrained Pseudorandom Functions. In a recent work, Boneh et al. [BW13], have used multilinear maps to construct a new variant of pseudorandom functions (PRFs) that they call constrained PRFs. In a standard PRF there is a master key that enables one to evaluate the function at all points in the domain of the function. On the other hand, in a constrained PRF it is possible to derive constrained keys from the master key. A constrained key enables the evaluation of the PRF at a certain subset of the domain and nowhere else. In the same work Boneh et al. [BW13] show that constrained PRFs can be used to construct other useful primitives such as identity based key exchange and a broadcast encryption system with optimal ciphertext size.

Removing Random Oracles. A sequence of works [FHPS13, HSW13] have used multilinear maps to provide standard model constructions of primitives previously known only using random oracles [BR93, CGH98]. In particular, Freire et al. [FHPS13] give new constructions of programmable hash functions (PHFs), an abstraction of random oracles that can also be instantiated in the standard model [HK08]. They then use these constructions to realize standard model versions of several primitive, such as Boneh-Franklin identity-based encryption scheme [BF01], the Boneh-Lynn-Shacham [BLS04] signature scheme, and the Sakai-Ohgishi-Kasahara identity-based non-interactive key exchange (ID-NIKE) scheme [SOK00]. These constructions can also be made hierarchical.

In the same vein, Hohenberger et al. [HSW13] provide standard model proofs for schemes with full domain hash structure [BR93, BR96] again in an attempt to avoid the random oracle heuristic [BR93, CGH98]. In particular they build an identity-based aggregate signature scheme that admits unrestricted aggregation.

CHAPTER 3

Multilinear Maps and Graded Encoding Systems

In this chapter we define formally our notion of a "approximate" multilinear maps, which we call *graded encoding schemes* (termed after the notion of graded algebra).

To make the analogy and differences from multilinear maps more explicit, we begin by recalling the notion of *cryptographic multilinear maps* of Boneh and Silverberg [BS03] (using a slightly different syntax).

3.1 Cryptographic Multilinear Maps

Below we define cryptographic multilinear maps.

Definition 3.1 (Multilinear Map)**.** *For $\kappa + 1$ cyclic groups $G_1, \ldots, G_\kappa, G_T$ (written additively) of the same order p, an κ-*multilinear map *$e : G_1 \times \cdots \times G_\kappa \to G_T$ has the following properties:*

1. *For elements $\{g_i \in G_i\}_{i=1,\ldots,\kappa}$, index $i \in [\kappa]$ and integer $\alpha \in \mathbb{Z}_p$, it holds that*
$$e(g_1, \ldots, \alpha \cdot g_i, \ldots, g_\kappa) = \alpha \cdot e(g_1, \ldots, g_\kappa).$$
2. *The map e is non-degenerate in the following sense: if the elements $\{g_i \in G_i\}_{i=1,\ldots,\kappa}$, are all generators of their respective groups, then $e(g_1, \ldots, g_\kappa)$ is a generator of G_T.*

Boneh and Silverberg considered in [BS03] only the *symmetric* case $G_1 = \cdots = G_\kappa$. The asymmetric case with different G_i's (as defined above) has also been considered in the literature, e.g., by Rothblum in [Rot13]. Unlike the above notion that allows for pairing of only batches of κ encodings at the time, we can consider a more general setting that allows for pairing any subset of encodings together as explained later in Section 3.2.

3.1.1 Efficient Procedures

To be useful for cryptographic applications, we need to be able to manipulate (representations of) elements in these groups efficiently, and at the same time we need some other manipulations to be computationally hard. Specifically, a cryptographic multilinear map scheme consists of efficient procedures for instance-generation, element-encoding validation, group-operation and negation, and multilinear map, $\mathcal{MMP} = (\mathsf{InstGen}, \mathsf{EncTest}, \mathsf{add}, \mathsf{neg}, \mathsf{map})$. These procedures are described below.

Instance Generation. A randomized algorithm $\mathsf{InstGen}$ that takes the security parameter λ and the multi-linearity parameter κ (both in unary), and outputs $(G_1, \ldots, G_T, p, e, g_1, \ldots, g_\kappa)$. Here the G_i's and G_T describe the groups, $p \in \mathbb{Z}$ is their order, $e : G_1 \times \cdots \times G_\kappa \to G_T$ describes an κ-multilinear map as above, and $g_i \in \{0,1\}^*$ for $i = 1, \ldots, \kappa$ encode generators in these groups. To shorten some of the notations below, we denote $\mathsf{params} = (G_1, \ldots, G_T, p, e)$.

Element Encoding. Given the instance params from above, an index $i \in [\kappa]$, and a string $x \in \{0,1\}^*$, $\mathsf{EncTest}(\mathsf{params}, i, x)$ decides if x encodes an element in G_i (and of course the g_i's output by the instance-generator are all valid encodings). Similarly $\mathsf{EncTest}(\mathsf{params}, \kappa + 1, x)$ efficiently recognizes description of elements in G_T.

It is usually assumed that elements have unique representation, namely for every i there are only p different strings representing elements in G_i. Below we therefore identify elements with their description, e.g. referring to "$x \in G_i$" rather than "x is a description of an element in G_i".

Group Operation. Given $x, y \in G_i$, $\mathsf{add}(\mathsf{params}, i, x, y)$ computes $x + y \in G_i$ and $\mathsf{neg}(\mathsf{params}, i, x)$ computes $-x \in G_i$. This implies also that for any $\alpha \in \mathbb{Z}_p$ we can efficiently compute $\alpha \cdot x \in G_i$.

Multilinear Map. For $\{x_i \in G_i\}_{i=1,\ldots,\kappa}$, $\mathsf{map}(\mathsf{params}, x_1, \ldots, x_\kappa)$ computes $e(x_1, \ldots, x_n) \in G_T$.

Another property, which was used by Papamanthou et al. [PTT10], is *compactness*, which means that the size of elements in the groups (as output by the instance generator) is independent of κ. Looking ahead we note that our multilinear maps do not satisfy this requirement, and are therefore unsuitable for the application in [PTT10]. For the same reasons we find our multilinear maps unsuitable for application of [Rot13].

3.1.2 Hardness Assumptions

For the multilinear map to be cryptographically useful, at least the discrete logarithm must be hard in the respective groups, and we usually also need the multilinear-DDH to be hard.

Multilinear Discrete-log (MDL). The Multilinear Discrete-Log problem is hard for a scheme $\mathcal{MMP}$, if for all $\kappa > 1$, all $i \in [\kappa]$, and all probabilistic polynomial time algorithms, the discrete-logarithm advantage of $\mathcal{A}$,

$$\mathsf{AdvDlog}_{\mathcal{MMP},\mathcal{A},\kappa}(\lambda) \overset{\text{def}}{=} \Pr\left[\mathcal{A}(\mathsf{params}, i, g_i, \alpha \cdot g_i) = \alpha \ : \ (\mathsf{params}, g_1, \ldots, g_l) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa), \alpha \leftarrow \mathbb{Z}_p\right],$$

is negligible in λ

Multilinear DDH (MDDH). For a symmetric scheme $\mathcal{MMP}$ (with $G_1 = G_2 = \cdots$), the Multilinear Decision-Diffie-Hellman problem is hard for $\mathcal{MMP}$ if for any κ and every probabilistic polynomial time algorithms $\mathcal{A}$, the advantage of $\mathcal{A}$ in distinguishing between the following two distributions is negligible in λ:

$$\begin{aligned} &(\mathsf{params}, g, \alpha_0 g, \alpha_1 g, \ldots, \alpha_\kappa g, \ (\prod_{i=0}^{\kappa} \alpha_i) \cdot e(g \ldots, g)) \\ \text{and} \quad &(\mathsf{params}, g, \alpha_0 g, \alpha_1 g, \ldots, \alpha_\kappa g, \ \alpha \cdot e(g, \ldots, g)) \end{aligned}$$

where $(\mathsf{params}, g) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$ and $\alpha, \alpha_0, \alpha_1, \ldots, \alpha_\kappa$ are uniformly random in $\mathbb{Z}_p$.

3.2 Graded Encoding Schemes

The starting point for our new notion is viewing group elements in multilinear-map schemes as just a convenient mechanism of encoding the exponent: Typical applications of bilinear (or more generally the envisioned multilinear) maps use $\alpha \cdot g_i$ as an "obfuscated encoding" of the "plaintext integer" $\alpha \in \mathbb{Z}_p$. This encoding supports limited homomorphism (i.e., linear operations and a limited number of multiplications) but no more.

In our setting we retain this concept of a somewhat homomorphic encoding, and have an algebraic ring (or field) R playing the role of the exponent space $\mathbb{Z}_p$. However we will dispose of the algebraic groups, replacing them with "unstructured" sets of encodings of ring elements.

Perhaps the biggest difference between our setting and the setting of cryptographic multilinear maps, is that our encodings are randomized, which means that the same ring-element can be encoded in many different ways. In our notion we do not even insist that the "plaintext version" of a ring element has a unique representation. This means that checking if two strings encode the same element may not be trivial, indeed our constructions rely heavily on this check being feasible for some encodings and not feasible for others.

Another important difference is that our system lets us multiply not only batches of κ encodings at the time, but in fact any subset of encodings. This stands in stark contrast to the sharp threshold in multi-linear maps, where one can multiply exactly κ encodings, no more and no less. A consequence of the ability to multiply any number of encodings is that we no longer have a single target group, instead we have a different "target group" for any number of multiplicands. This yields a richer structure, roughly analogous to *graded algebra*.

In its simplest form (analogous to symmetric maps with a single source group), we have levels of encodings: At level zero we have the "plaintext" ring elements $\alpha \in R$ themselves, level one corresponds to $\alpha \cdot g$ in the source group, and level-i corresponds to a product of i level-1 encodings (so level-κ corresponds to the target group from multilinear maps).

For the sake of simplicity, in this section we will restrict to the case of symmetric multilinear maps and provide the extensions of these definitions to the asymmetric setting in Appendix A.

Definition 3.2 (κ-Graded Encoding System). *A κ-Graded Encoding System consists of a ring R and a system of sets $\mathcal{S} = \{S_i^{(\alpha)} \subset \{0,1\}^* : \alpha \in R,\ 0 \le i \le \kappa,\}$, with the following properties:*

1. *For every fixed index i, the sets $\{S_i^{(\alpha)} : \alpha \in R\}$ are disjoint (hence they form a partition of $S_i \stackrel{\text{def}}{=} \bigcup_\alpha S_{\boldsymbol{v}}^{(\alpha)}$).*
2. *There is an associative binary operation '$+$' and a self-inverse unary operation '$-$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every index $i \le \kappa$, and every $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, it holds that*
$$u_1 + u_2 \in S_i^{(\alpha_1+\alpha_2)} \quad \text{and} \quad -u_1 \in S_i^{(-\alpha_1)}$$
where $\alpha_1 + \alpha_2$ and $-\alpha_1$ are addition and negation in R.
3. *There is an associative binary operation '$\times$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every i_1, i_2 with $i_1 + i_2 \le \kappa$, and every $u_1 \in S_{i_1}^{(\alpha_1)}$ and $u_2 \in S_{i_2}^{(\alpha_2)}$, it holds that $u_1 \times u_2 \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$. Here $\alpha_1 \cdot \alpha_2$ is multiplication in R, and $i_1 + i_2$ is integer addition.*

Clearly, Definition 3.2 implies that if we have a collection of n encodings $u_j \in S_{i_j}^{(\alpha_j)}$, $j = 1, 2 \ldots, n$, then as long as $\sum_j i_j \le \kappa$ we get $u_1 \times \cdots \times u_n \in S_{i_1+\cdots+i_n}^{(\prod_j \alpha_j)}$.

3.2.1 Efficient Procedures, the Dream Version

To be useful, we need efficient procedures for manipulating encodings well as as hard computational tasks. To ease the exposition, we first describe a "dream version" of the efficient procedures (which we do not know how to realize), and then explain how to modify them to deal with technicalities that arise from our use of lattices in the realization.

Instance Generation. The randomized $\mathsf{InstGen}(1^\lambda, 1^\kappa)$ takes as inputs the parameters λ, κ, and outputs $(\mathsf{params}, \mathbf{p}_{zt})$, where params is a description of a κ-Graded Encoding System as above, and $\mathbf{p}_{zt}$ is a zero-test parameter for level κ (see below).

Ring Sampler. The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_0^{(\alpha)}$ for a nearly uniform element $\alpha \in_R R$. (Note that we require that the "plaintext" $\alpha \in R$ is nearly uniform, but not that the encoding a is uniform in $S_0^{(\alpha)}$.)

Encoding. The (possibly randomized) $\mathsf{enc}(\mathsf{params}, i, a)$ takes a "level-zero" encoding $a \in S_0^{(\alpha)}$ for some $\alpha \in R$ and index $i \leq \kappa$, and outputs the "level-i" encoding $u \in S_i^{(\alpha)}$ for the same α.

Addition and negation. Given params and two encodings relative to the same index, $u_1 \in S_i^{(\alpha_1)}$ and $u_2 \in S_i^{(\alpha_2)}$, we have $\mathsf{add}(\mathsf{params}, i, u_1, u_2) = u_1 + u_2 \in S_i^{(\alpha_1+\alpha_2)}$, and $\mathsf{neg}(\mathsf{params}, i, u_1) = -u_1 \in S_i^{(-\alpha_1)}$.

Multiplication. For $u_1 \in S_{i_1}^{(\alpha_1)}$, $u_2 \in S_{i_2}^{(\alpha_2)}$ such that $i_1+i_2 \leq \kappa$, we have $\mathsf{mul}(\mathsf{params}, i_1, u_1, i_2, u_2) = u_1 \times u_2 \in S_{i_1+i_2}^{(\alpha_1 \cdot \alpha_2)}$.

Zero-test. The procedure $\mathsf{isZero}(\mathsf{params}, u)$ output 1 if $u \in S_\kappa^{(0)}$ and 0 otherwise. Note that in conjunction with the subtraction procedure, this lets us test if $u_1, u_2 \in S_\kappa$ encode the same element $\alpha \in R$.

Extraction. This procedure extracts a "canonical" and "random" representation of ring elements from their level-κ encoding. Namely $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u)$ outputs (say) $s \in \{0,1\}^\lambda$, such that:

(a) For any $\alpha \in R$ and two $u_1, u_2 \in S_\kappa^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_1) = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_2)$,
(b) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u) \ : \ \alpha \in_R R, u \in S_\kappa^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.

3.2.2 Efficient Procedures, the Real-Life Version

Our realization of the procedures above over ideal lattices uses noisy encodings, where the noise increases with every operation and correctness is only ensured as long as it does not increase too much. We therefore modify the procedures above, letting them take as input (and produce as output) also a bound on the noise magnitude of the encoding in question. The procedures are allowed to abort if the bound is too high (relative to some maximum value which is part of the instance description params). Also, they provide no correctness guarantees if the bound on their input is "invalid." (When B is a noise-bound for some encoding u, we say that it is "valid" if it is at least as large as the bound produced by the procedure that produced u itself, and moreover any encoding that was used by that procedure (if any) also came with a valid noise bound.) Of course we also require that these procedure do not always abort, i.e. they should support whatever set of operations that the application calls for, before the noise becomes too large. Finally, we also relax the requirements on the zero-test and the extraction routines. Some more details are described next:

Zero-test. We sometime allow false positives for this procedure, but not false negatives. Namely, $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1$ for every $u \in S_\kappa^{(0)}$, but we may have $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1$ also for some $u \notin S_\kappa^{(0)}$. The weakest functionality requirement that we make is that for a uniform random choice of $\alpha \in_R R$, we have

$$\Pr_{\alpha \in_R R}\left[\exists\, u \in S_\kappa^{(\alpha)} \text{ s.t } \mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1\right] = \mathsf{negl}(\lambda). \tag{3.1}$$

Additional requirements are considered security features (that a scheme may or may not possess), and are discussed later in this section.

Extraction. We replace[1] the properties (a)-(b) from above dream version by the weaker requirements:

(a′) For a randomly chosen $a \leftarrow \mathsf{samp}(\mathsf{params})$, if we run the encoding algorithm twice to encode a at level κ and then extract from both copies then we get:

$$\Pr\left[\begin{array}{l} \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_1) \\ = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_2) \end{array} \;:\; \begin{array}{l} a \leftarrow \mathsf{samp}(\mathsf{params}) \\ u_1 \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a) \\ u_2 \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a) \end{array}\right] \;\geq\; 1 - \mathsf{negl}(\lambda).$$

(b′) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u) : a \leftarrow \mathsf{samp}(\mathsf{params}), u \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, a)\}$ is nearly uniform over $\{0,1\}^\lambda$.

We typically need these two conditions to hold even if the noise bound that the encoding routine takes as input is larger than the one output by samp (upto some maximum value).

3.2.3 Hardness Assumptions

Our hardness assumptions are modeled after the discrete-logarithm and MDDH assumptions in multilinear groups. For example, the most direct analog of the discrete-logarithm problem is trying to obtain a level-zero encoding $a \in S_0^{(\alpha)}$ for $\alpha \in R$ from an encoding relative to some other index $i > 0$.

The analog of MDDH in our case roughly says that given $\kappa + 1$ level-one encoding of random elements it should be infeasible to generate a level-κ encoding of their product, or even to distinguish it from random. To formalize the assumption we should specify how to generate level-κ encodings of the "the right product" and of a random element. One way to formalize it is by the following process. (Below we suppress the noise bounds for readability):

```
1. $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$
2. For $i = 0, \ldots, \kappa$:
3.     Choose $a_i \leftarrow \mathsf{samp}(\mathsf{params})$   // level-0 encoding of random $\alpha_i \in_R R$
4.     Set $u_i \leftarrow \mathsf{enc}(\mathsf{params}, 1, a_i)$     // level-1 encoding of the $\alpha_i$'s
5. Set $\tilde{a} = \prod_{i=0}^{\kappa} a_i$                      // level-0 encoding of the product
6. Choose $\hat{a} \leftarrow \mathsf{samp}(\mathsf{params})$         // level-0 encoding of a random element
```

[1] Our construction from Chapter 6 does not support full canonicalization. Instead, we settle for $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u)$ that has a good chance of producing the same output when applied to different encoding of the same elements.

```
7. Set $\tilde{u} \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, \tilde{a})$        // level-$\kappa$ encoding of the product
8. Set $\hat{u} \leftarrow \mathsf{enc}(\mathsf{params}, \kappa, \hat{a})$        // level-$\kappa$ encoding of random
```

(We note that with the noise bound, it may be important that the encoding routines for both $\tilde{a}$ and $\hat{a}$ get as input the same bound, i.e., the largest of the bounds for $\tilde{a}$ and $\hat{a}$.) The GDDH distinguisher gets all the level-one u_i's and either $\tilde{u}$ (encoding the right product) or $\hat{u}$ (encoding a random element), and it needs to decide which is the case. In other words, the GDDH assumption says that for any setting of the parameters, the following two distributions, defined over the experiment above, are computationally indistinguishable:

$$\mathcal{D}_{GDDH} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \tilde{u})\} \text{ and } \mathcal{D}_{RAND} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \hat{u})\}.$$

Zero-test security. In some settings we may be concerned with adversaries that can generate encodings in a malicious way and submit them to the zero-test procedure. In such settings, the statistical property from Equation (3.1) is not sufficient, instead we would like the zero-test to accept *only* encoding of zero at the right level. This can be statistical (i.e. no false positive exist) or computational (i.e. it is hard to find them).

Definition 3.3. *A graded-encoding system enjoys* statistical *zero-test security if the only strings that pass the zero-test are encodings of zero, except with a negligible probability over the instance generation. That is, for every κ:*

$$\Pr_{\mathsf{params}, \mathbf{p}_{zt}}[\exists\ u \notin S_\kappa^{(0)} \text{ s.t. } \mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1] \leq \textit{negligible}(\lambda),$$

where the probability is taken over $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$. And we say that the graded-encoding system enjoys computational *zero-test security if for every adversary $\mathcal{A}$ and parameters as above:*

$$\Pr_{\substack{(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa) \\ u \leftarrow \mathcal{A}(\mathsf{params}, \mathbf{p}_{zt})}}\left[u \notin S_\kappa^{(0)} \text{ but } \mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, u) = 1\right] \leq \textit{negligible}(\lambda).$$

CHAPTER 4

Preliminaries I : Lattices

We denote set of complex number by $\mathbb{C}$, real numbers by $\mathbb{R}$, the rationals by $\mathbb{Q}$ and the integers by $\mathbb{Z}$. For a positive integer n, $[n]$ denotes the set $\{1, \ldots, n\}$. We extend any real function $f(\cdot)$ to a countable set A by defining $f(A) = \sum_{x \in A} f(x)$.

By convention, vectors are assumed to be in column form and are written using bold lower-case letters, e.g. $\boldsymbol{x}$. The ith component of $\boldsymbol{x}$ will be denoted by x_i. We will use $\boldsymbol{x}^T$ to denotes the transpose of $\boldsymbol{x}$. For a vector $\boldsymbol{x}$ in $\mathbb{R}^n$ or $\mathbb{C}^n$ and $p \in [1, \infty]$, we define the ℓ_p norm as $\|\boldsymbol{x}\|_p = \left(\sum_{i \in [n]} |x_i|^p\right)^{1/p}$ where $p < \infty$, and $\|\boldsymbol{x}\|_\infty = \max_{i \in [n]} |x_i|$ where $p = \infty$. Whenever p is not specified, $\|\boldsymbol{x}\|$ is assumed to represent the ℓ_2 norm (also referred to as the Euclidean norm).

Matrices are written as bold capital letters, e.g. $\boldsymbol{X}$, and the ith column vector of a matrix $\boldsymbol{X}$ is denoted $\boldsymbol{x}_i$. The length of a matrix is the norm of its longest column: $\|\boldsymbol{X}\| = \max_i \|\boldsymbol{x}_i\|$. For notational convenience, we sometimes view a matrix as simply the set of its column vectors. Finally we will denote the transpose and the inverse (if it exists) of a matrix $\boldsymbol{X}$ with $\boldsymbol{X}^T$ and $\boldsymbol{X}^{-1}$ respectively.

The natural security parameter throughout the thesis is λ, and all other quantities are implicitly assumed to be functions of λ. We use standard big-O notation to classify the growth of functions, and say that $f(\lambda) = \tilde{O}(g(\lambda))$ if $f(\lambda) = O(g(\lambda) \cdot log^c\ \lambda)$ for some fixed constant c. We let $\mathsf{poly}(\lambda)$ denote an unspecified function $f(\lambda) = O(\lambda^c)$ for some constant c. A *negligible* function, denoted generically by $\mathsf{negl}(\lambda)$, is an $f(\lambda)$ such that $f(\lambda) = o(\lambda^{-c})$ for every fixed constant c. We say that a function is *overwhelming* if it is $1 - \mathsf{negl}(\lambda)$.

The *statistical distance* between two distributions X and Y over a domain D is defined to be $\frac{1}{2} \sum_{d \in D} |\Pr[X = d] - \Pr[Y = d]|$. We say that two ensembles of distributions $\{X_\lambda\}$ and $\{Y_\lambda\}$ are *statistically indistinguishable* if for every λ the statistical distance between X_λ

and Y_λ is negligible in λ.

Two ensembles of distributions $\{X_\lambda\}$ and $\{Y_\lambda\}$ are *computationally indistinguishable* if for every probabilistic poly-time (in λ) machine $\mathcal{A}$, $|\Pr[\mathcal{A}(1^\lambda, X_\lambda) = 1] - \Pr[\mathcal{A}(1^\lambda, Y_\lambda) = 1]|$ is negligible in λ. The definition is extended to non-uniform families of poly-sized circuits in the standard way.

4.1 Lattices

A lattice Λ is an additive discrete sub-group of $\mathbb{R}^n$, i.e., it is a subset $\Lambda \subset \mathbb{R}^n$ satisfying the following properties:

(subgroup) λ is closed under addition and subtraction,

(discrete) there is an $\epsilon > 0$ such that any two distinct lattice points $\boldsymbol{x} \neq \boldsymbol{y} \in \Lambda$ are at distance at least $\|\boldsymbol{x} - \boldsymbol{y}\| \geq \epsilon$.

Let $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_k\} \subset \mathbb{R}^n$ consist of k linearly independent vectors in $\mathbb{R}^n$. The lattice generated by the $\boldsymbol{B}$ is the set

$$\mathcal{L}(\boldsymbol{B}) = \{\boldsymbol{B}\boldsymbol{z} = \sum_{i=1}^{k} z_i \boldsymbol{b}_i : \boldsymbol{z} \in \mathbb{Z}^k\},$$

of all the integer linear combinations of the columns of $\boldsymbol{B}$. The matrix $\boldsymbol{B}$ is called a *basis* for the lattice $\mathcal{L}(\boldsymbol{B})$. The integers n and k are called the *dimension* and *rank* of the lattice. If $n = k$ then $\mathcal{L}(\boldsymbol{B})$ is called a *full-rank* lattice. We will only be concerned with full-rank lattices, hence unless otherwise mentioned we will assume that the lattice considered is full-rank. Notice the similarity in the definition of a lattice with the definition of vector space generated by $\boldsymbol{B}$:

$$\text{span}(\boldsymbol{B}) = \{\boldsymbol{B} \cdot \boldsymbol{x} : \boldsymbol{x} \in \mathbb{R}^n\}.$$

Also the *fundamental parallelepiped* of $\boldsymbol{B}$, denoted as $\mathcal{P}(\boldsymbol{B})$ is defined as

$$\mathcal{P}(\boldsymbol{B}) = \{\boldsymbol{B}\boldsymbol{x} : x \in [0,1)^k\}.$$

The *minimum distance* $\lambda_1(\Lambda)$ of a lattice Λ is the length (in the Euclidean ℓ_2 norm, unless otherwise indicated) of its shortest nonzero vector: $\lambda_1(\Lambda) = \min_{\boldsymbol{x} \neq \boldsymbol{0}, \boldsymbol{x} \in \Lambda} \|\boldsymbol{x}\|$. More generally, the *ith successive minimum* $\lambda_i(\Lambda)$ is the smallest radius r such that Λ contains i linearly independent vectors of norm at most r. We write λ_1^∞ to denote the minimum distance measured in the ℓ_∞ norm (which as mentioned earlier, is defined as $\|\boldsymbol{x}\|_\infty = \max |x_i|$).

For lattices $\Lambda' \subseteq \Lambda$, the quotient group Λ/Λ' (also written as $\Lambda \mod \Lambda'$) is well-defined as the additive group of distinct *cosets* $\boldsymbol{v} + \Lambda'$ for $\boldsymbol{v} \in \Lambda$, with addition of cosets defined in the usual way.

The *dual lattice* of a full-rank lattice Λ, denoted Λ^*, is defined to be

$$\Lambda^* = \{\boldsymbol{x} \in \mathbb{R}^n : \forall \boldsymbol{v} \in \Lambda, \langle \boldsymbol{x}, \boldsymbol{v} \rangle \in \mathbb{Z}\}.$$

In general, we define

$$\Lambda^* = \{\boldsymbol{x} \in \mathrm{span}(\boldsymbol{B}) : \forall \boldsymbol{v} \in \Lambda, \langle \boldsymbol{x}, \boldsymbol{v} \rangle \in \mathbb{Z}\},$$

where $\boldsymbol{B}$ is a basis for Λ. If $\boldsymbol{B}$ is a basis of Λ, then we have that $\boldsymbol{B}^* = \boldsymbol{B}(\boldsymbol{B}^T\boldsymbol{B})^{-1}$ is a basis of Λ^*. For the special case, when Λ is a full rank lattice we have that $\boldsymbol{B}^* = (\boldsymbol{B}^{-1})^T$ is a basis of Λ^*.

4.2 Gaussians on Lattices

Review of Gaussian measure over lattices presented here follows the development by prior works [Reg04, AR05, MR07, GPV08, AGHS12]. For any real $s > 0$, define the (spherical) *Gaussian function* $\rho_s : \mathbb{R}^n \to (0, 1]$ with[1] parameter s as:

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \rho_s(\boldsymbol{x}) = \exp(-\pi\langle \boldsymbol{x}, \boldsymbol{x}\rangle/s^2) = \exp(-\pi\|\boldsymbol{x}\|^2/s^2).$$

For any real $s > 0$, and n-dimensional lattice Λ, define the (spherical) *discrete Gaussian distribution* over Λ as:

$$\forall \boldsymbol{x} \in \Lambda, D_{\Lambda,s}(\boldsymbol{x}) = \frac{\rho_s(\boldsymbol{x})}{\rho_s(\Lambda)}.$$

This generalizes to ellipsoid Gaussians, where the different coordinates are jointly Gaussian but not independent, where we replace the parameter $s^2 \in \mathbb{R}$ by a symmetric positive definite[2] covariance matrix in $\mathbb{R}^{n\times n}$. For any rank-n matrix $\boldsymbol{S} \in \mathbb{R}^{m\times n}$, the *ellipsoid Gaussian function* on $\mathbb{R}^n$ parameterized by a nonsingular matrix $\boldsymbol{S}$ is defined by

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \rho_{\boldsymbol{S}}(\boldsymbol{x}) = \exp\big(-\pi \cdot \langle \boldsymbol{S}^{-1}\boldsymbol{x}, \boldsymbol{S}^{-1}\boldsymbol{x}\rangle\big) = \exp\big(-\pi \cdot \boldsymbol{x}^T(\boldsymbol{S}^T\boldsymbol{S})^{-1}\boldsymbol{x}\big).$$

Clearly this function only depends on $\boldsymbol{S}^T\boldsymbol{S}$ and not on the particular choice of $\boldsymbol{S}$. Note that for any nonsingular matrix $\boldsymbol{S}$ the symmetric matrix $\boldsymbol{S}^T\boldsymbol{S}$ is positive definite because

$$\boldsymbol{x}^T\boldsymbol{S}^T\boldsymbol{S}\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{S}^T(\boldsymbol{x}^T\boldsymbol{S}^T)^T = \langle \boldsymbol{x}^T\boldsymbol{S}^T, \boldsymbol{x}^T\boldsymbol{S}^T\rangle = \|(\boldsymbol{x}^T\boldsymbol{S}^T)\|^2 > 0$$

for all $\boldsymbol{x} \in \mathbb{R}^n$. Notice that the spherical case can be obtained by setting $\boldsymbol{S} = s\boldsymbol{I}_n$, with $\boldsymbol{I}_n$ the n-by-n identity matrix. Normalizing, *ellipsoid discrete Gaussian distribution* over lattice Λ with parameter $\boldsymbol{S}$ is

$$\forall\ \boldsymbol{x} \in \Lambda, D_{\Lambda,\boldsymbol{S}}(\boldsymbol{x}) = \frac{\rho_{\boldsymbol{S}}(\boldsymbol{x})}{\rho_{\boldsymbol{S}}(\Lambda)}.$$

[1] The Gaussian function can be defined more generally as being centered around a specific vector $\boldsymbol{c}$ instead of $\boldsymbol{0}$ as done here. The simpler definition considered here suffices for our purposes.

[2] A *symmetric* matrix is a square matrix that is equal to its transpose. A symmetric $n \times n$ real matrix $\boldsymbol{M}$ is said to be *positive definite*, written $\boldsymbol{M} > 0$, if $\boldsymbol{z}^T\boldsymbol{M}\boldsymbol{z}$ is positive for all non-zero $\boldsymbol{z} \in \mathbb{R}^n$.

Smoothing Parameter. Micciancio and Regev [MR07] introduced a lattice quantity called the *smoothing parameter*, and related it other lattice parameters.

Definition 4.1 (Smoothing Parameter, [MR07, Definition 3.1])**.** *For an n-dimensional lattice Λ, and positive real $\epsilon > 0$, we define its* smoothing parameter *denoted $\eta_\epsilon(\Lambda)$, to be the smallest s such that $\rho_{1/s}(\Lambda^* \setminus \{\mathbf{0}\}) \le \epsilon$.*

Intuitively, for a small enough ϵ, the number $\eta_\epsilon(\Lambda)$ is sufficiently larger than a fundamental parallelepiped of Λ so that sampling from the corresponding Gaussian "wipes out the internal structure" of Λ. The following Lemma 4.3 and Corollary 4.4 formally provide this claim. The bounds on $\eta_\epsilon(\Lambda)$ are specified by Lemma 4.2. Finally Lemma 4.5 provides bounds on the length of a vector sampled from a Gaussian.

Lemma 4.2 ([MR07, Lemma 3.3])**.** *For any n-dimensional lattice Λ and positive real $\epsilon > 0$, we have that*

$$\eta_\epsilon(\Lambda) \le \sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \lambda_n(\Lambda).$$

The following lemma explains the name "smoothing parameter."

Lemma 4.3 ([MR07, Lemma 4.1])**.** *For any lattice Λ, $\epsilon > 0$, $s \ge \eta_\epsilon(\Lambda)$, and $\boldsymbol{c} \in \mathbb{R}^n$, the statistical distance between $D_s + \boldsymbol{c} \mod \Lambda$ and the uniform distribution modulo Λ is at most $\epsilon/2$.*

Corollary 4.4 ([GPV08, Corollary 2.8])**.** *Let Λ, Λ' be n-dimensional lattices, with $\Lambda' \subseteq \Lambda$. Then for any $\epsilon \in (0, \frac{1}{2})$, any $s \ge \eta_\epsilon(\Lambda')$, the distribution of $(D_{\Lambda,s} \mod \Lambda')$ is within a statistical distance at most 2ϵ of uniform over $(\Lambda \mod \Lambda')$.*

Lemma 4.5 ([MR07, Lemma 4.4] and [BF11b, Proposition 4.7])**.** *For any n-dimensional lattice Λ, and $s \ge \eta_\epsilon(\Lambda)$ for some negligible ϵ, then for any constant $\delta > 0$ we have*

$$\Pr_{\boldsymbol{x} \leftarrow D_{\Lambda,s}}\left[(1-\delta)s\sqrt{\frac{n}{2\pi}} \le \|\boldsymbol{x}\| \le (1+\delta)s\sqrt{\frac{n}{2\pi}}\right] \ge 1 - \mathsf{negl}(n).$$

Next we present a generalization of Lemma 4.5 to the setting of ellipsoidal Gaussians [AGHS12]. Specifically Lemma 4.6 claims that the size of vectors drawn from $D_{\Lambda,\boldsymbol{S}}$ is roughly bounded by the largest singular value of $\boldsymbol{S}$. Recall that the largest and least singular values of a full rank matrix $\boldsymbol{S} \in \mathbb{R}^{m \times n}$ are defined as $\sigma_1(\boldsymbol{S}) = \sup(U_{\boldsymbol{S}})$ and $\sigma_n(\boldsymbol{S}) = \inf(U_{\boldsymbol{S}})$, respectively, where $U_{\boldsymbol{S}} = \{\|\boldsymbol{S}\boldsymbol{u}\| : \boldsymbol{u} \in \mathbb{R}^n, \|\boldsymbol{u}\| = 1\}$.

Lemma 4.6 ([AGHS12, Lemma 3])**.** *For a rank-n lattice Λ, constant $0 < \epsilon < 1$ and matrix $\boldsymbol{S}$ such that $\sigma_n(\boldsymbol{S}) \ge \eta_\epsilon(L)$, we have:*

$$\Pr_{\boldsymbol{x} \leftarrow D_{\Lambda,\boldsymbol{S}}}\left[\|\boldsymbol{x}\| \le \sigma_1(\boldsymbol{S})\sqrt{n}\right] \ge 1 - \mathsf{negl}(n).$$

4.3 Sampling from Discrete Gaussian

In this section we will recall different mechanisms of sampling from discrete gaussian distributions and some of their properties.

GPV Sampling Algorithm. The GPV sampler [GPV08] provides a polynomial-time procedure for sampling from the discrete Gaussian distribution over a lattice Λ. More precisely, given a basis $\boldsymbol{B}$ of Λ, and a sufficiently large s (related to the "quality" of $\boldsymbol{B}$), the GPV algorithm outputs a sample from a distribution statistically close to $D_{\Lambda,s}$. Informally speaking, the sampling algorithm is "zero-knowledge" in the sense that it leaks no information about its input basis $\boldsymbol{B}$ (aside from a bound on its quality), because $D_{\Lambda,s}$ is defined without reference to any particular basis. This zero-knowledge property accounts for its broad utility in lattice-based cryptography. This sampling algorithm has been improved by Peikert [Pei10], however for concreteness we stick with the GPV sampling algorithm.

Theorem 4.7 ([GPV08, Theorem 4.1])**.** *There is a probabilistic polynomial-time algorithm that, given a basis $\boldsymbol{B}$ of an n-dimensional lattice $\Lambda = \mathcal{L}(\boldsymbol{B})$, a parameter $s \geq \|\tilde{\boldsymbol{B}}\| \cdot \omega(\sqrt{\log n})$, outputs a sample from a distribution that is statistically close to $D_{\Lambda,s}$. Here $\tilde{\boldsymbol{B}}$ denotes the Gram-Schmidt orthogonalization of $\boldsymbol{B}$.*[3]

Discrete Gaussian Leftover Hash Lemma. A recent work [AGHS12] considers an alternate way of sampling from a gaussian distribution. The process begins by choosing "once and for all" m points in a lattice Λ, drawn independently from a "wide enough discrete Gaussian" choosing an appropriate parameter s, namely $\boldsymbol{x}_i \leftarrow D_{\Lambda,s}$ for $i \in [m]$. Once the $\boldsymbol{x}_i$'s are fixed, they are arranged as the rows of an m-by-n matrix $\boldsymbol{X} = (\boldsymbol{x}_1|\boldsymbol{x}_2|\ldots|\boldsymbol{x}_m)^T$, and we consider the distribution $\mathcal{E}_{\boldsymbol{X},s'}$, induced by choosing an integer vector $\boldsymbol{v}$ from a discrete spherical Gaussian over $\mathbb{Z}^m$ with parameter s' and outputting $\boldsymbol{y} = \boldsymbol{X}^T\boldsymbol{v}$,

$$\mathcal{E}_{\boldsymbol{X},s'} \overset{\text{def}}{=} \{\boldsymbol{X}^T\boldsymbol{v} : \boldsymbol{v} \leftarrow D_{\mathbb{Z}^m,s'}\}.$$

[AGHS12] proved that with high probability over the choice of $\boldsymbol{X}$, the distribution $\mathcal{E}_{\boldsymbol{X},s'}$ is statistically close to the ellipsoid Gaussian $D_{\Lambda,s'\boldsymbol{X}}$.

Theorem 4.8 ([AGHS12, Theorem 3])**.** *Let Λ be a lattice $\Lambda \subset \mathbb{R}^n$ and $\boldsymbol{B}$ a matrix whose rows form a basis of Λ, and denote $\chi = \sigma_1(\boldsymbol{B})/\sigma_n(\boldsymbol{B})$. Also let ϵ be negligible in n, and let m, s, s' be parameters such that $s \geq \eta_\epsilon(\mathbb{Z}^n)$, $m \geq 10n\log(8(mn)^{1.5}s\chi)$ and $s' \geq 4mn\chi\ln(1/\epsilon)$.*

Then, when choosing the rows of an m-by-n matrix X from the spherical Gaussian over Λ, $\boldsymbol{X} \leftarrow (D_{\Lambda,s})^m$, we have with all but probability $2^{-O(m)}$ over the choice of $\boldsymbol{X}$, that the statistical distance between $\mathcal{E}_{\boldsymbol{X},s'}$ and the ellipsoid Gaussian $D_{\Lambda,s'\boldsymbol{X}}$ is bounded by 2ϵ.

[3] *In the Gram-Schmidt orthogonalization $\tilde{\boldsymbol{B}}$ of $\boldsymbol{B}$, the vector $\tilde{\boldsymbol{b}}_i$ is the projection of $\boldsymbol{b}_i$ orthogonally to span$(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_{i-1})$. As a point of comparison, $\|\tilde{\boldsymbol{B}}\|$ is always at most $\|\boldsymbol{B}\|$, and in some cases can be substantially smaller.*

Lemma 4.9 ([AGHS12, Lemma 8])**.** *There exists a universal constant $K > 1$ such that for all $m \geq 2n$, $\epsilon > 0$ and every n-dimensional real lattice $\Lambda \subset \mathbb{R}^n$, the following holds: Choosing the rows of an m-by-n matrix $\boldsymbol{X}$ independently at random from a spherical discrete Gaussian on Λ with parameter $s > 2K\eta_\epsilon(\Lambda)$, namely $\boldsymbol{X} \leftarrow (D_{\Lambda,s})^m$, we have*

$$\Pr\left[s\sqrt{2\pi m}/K < \sigma_n(\boldsymbol{X}) \leq \sigma_1(\boldsymbol{X}) < sK\sqrt{2\pi m}\right] > 1 - (4m\epsilon + O(\exp(-m/K))).$$

CHAPTER 5

Preliminaries II : Algebraic Number Theory Background

Algebraic number theory is the study of *number fields*. Here we review the background essential for understanding our encoding scheme. We consider the special case of *cyclotomic* number fields as a special example of particular interest. Much of our description here follows [LPR10], and we refer the reader to [Jan96, Ste04, Oss08, Wes99] for detailed background reading. Additional background will be necessary for our study of cryptanalysis and is recalled later in Chapter 8.

5.1 Number Fields and Ring of Integers

An algebraic number field (or simply number field) K is a finite (and hence algebraic) field extension of the field of rational numbers $\mathbb{Q}$. In this section we will recall definition of some of these elementary notions.

Definition 5.1 (Algebraic Number and Algebraic Integer)**.** *We say that $\zeta \in \mathbb{C}$ is an* algebraic number *if it is a root of a polynomial $f(x) \in \mathbb{Q}[x]$. Furthermore, we say that that ζ is an* algebraic integer *if additionally $f(x)$ is a monic (a polynomial whose leading coefficient is 1) polynomial in $\mathbb{Z}[x]$.*

Definition 5.2 (Minimal Polynomial)**.** *The* minimal polynomial *of ζ is the monic polynomial $f(x) \in \mathbb{Q}[x]$ of least positive degree such that $f(\zeta) = 0$.*

The *conjugates* of ζ are defined by all the roots of its minimal polynomial.

Proposition 5.3 ([Ste04, Lemma 5.1.3])**.** *If ζ is an algebraic integer, then the minimal polynomial of ζ is in $\mathbb{Z}[x]$.*

Proposition 5.4 ([Ste04, Proposition 5.1.5]). *The set of all algebraic integers form a ring, i.e., the sum and product of two algebraic integers is again an algebraic integer.*

Now we are ready to define the notion of a number field and its ring of integers.

Definition 5.5 (Number Field and Ring of Integers). *A* number field *is a field extension $K = \mathbb{Q}(\zeta)$ obtained by adjoining an algebraic number ζ to the field of rationals $\mathbb{Q}$. The* ring of integers *of a number field K is the ring*

$$\mathcal{O}_K = \{x \in K : x \textit{ is an algebraic integer.}\}$$

Let the minimal polynomial $f(x)$ of ζ have degree n. Then because $f(\zeta) = 0$, there is a natural isomorphism between $\mathbb{Q}[x] \mod f(x)$ and K, given by $x \mapsto \zeta$, and the number field K can be seen as an n-dimensional vector space over $\mathbb{Q}$ with basis $\{1, \zeta, \ldots, \zeta^{n-1}\}$. This is called the *power basis* of K.

The case of Cyclotomic Number Fields. Let $\zeta_m = e^{2\pi\sqrt{-1}/m} \in \mathbb{C}$ denote a *primitive* m-th root of unity. (Recall that an mth root of unity is said to be a *primitive* root if it is not a kth root for some $0 < k < m$.)

Definition 5.6 (Cyclotomic Polynomial). *The m-th* cyclotomic polynomial*, denote by $\Phi_m(x)$, is defined as the product*

$$\Phi_m(x) = \prod_{k \in \mathbb{Z}_m^*} (x - \zeta_m^k).$$

Observe that the values ζ^k run over all the primitive m^{th} roots of unity in $\mathbb{C}$, thus $\Phi_m(x)$ has degree $n = \varphi(m)$, where $\varphi(m)$ denotes the *Euler's totient* or *phi function*. Recall that if m is a positive integer, then $\varphi(m)$ is the number of integers in the set $\{1, 2, \ldots, m\}$ that are relatively prime to m.

It is easy to see that $\Phi_m(x)$ is monic. It is also known (a nontrivial result due to Gauss) that $\Phi_m(x)$ is in $\mathbb{Z}[x]$ and is irreducible over $\mathbb{Q}$. Therefore ζ_m is an algebraic integer with the minimal polynomial $\Phi_m(x)$.

The cyclotomic polynomial $\Phi_m(x)$ may be computed by (exactly) dividing $x^n - 1$ by the cyclotomic polynomials of the proper divisors of n previously computed recursively (setting, $\Phi_1(x) = x - 1$) by the same method:

$$\Phi_m(x) = \frac{x^m - 1}{\prod_{\substack{d|m \\ d<m}} \Phi_d(x)}.$$

Two useful facts about cyclotomic polynomials are that $\Phi_m(x) = \frac{x^m-1}{x-1} = x^{m-1} + \ldots + x + 1$ for prime m, and $\Phi_m(x) = \Phi_{m_0}(x^{m/m_0})$ where m_0 is the radical of m, i.e., the product of all primes diving m. For instance, $\Phi_8(c) = x^4 + 1$ and $\Phi_9(x) = x^6 + x^3 + 1$. We will be most interested in the case when $m \geq 2$ is a power of 2 in which case $\Phi_m(x) = x^{m/2} + 1$. (However, not all cyclotomic polynomials have 0-1, or even small coefficients: e.g., $\Phi_6(x) = x^2 - x + 1$, $\Phi_{3\cdot5\cdot7}$ has a -2 coefficient, and $\Phi_{3\cdot5\cdot7\cdot11\cdot13}(x)$ has coefficients with magnitudes as large as 22.)

Definition 5.7. *The mth* cyclotomic field $\mathbb{Q}(\zeta_m)$ *(with $m > 2$) is obtained by adjoining ζ_m to $\mathbb{Q}$.*

Proposition 5.8 ([Jan96, p 48, Proposition 4.3])**.** *The ring of integers in $\mathbb{Q}(\zeta_m)$ is $\mathbb{Z}(\zeta_m)$. This ring $\mathbb{Z}(\zeta_m)$ is called the* cyclotomic ring.

5.2 Embeddings and Geometry

In this section we will recall various geometric interpretations of a number field and most importantly define different notion of norm essential for our study.

Canonical Embedding. A number field $K = \mathbb{Q}(\zeta)$ of degree[1] n has [Wes99, p 9, Proposition 2.1] exactly n field homomorphisms $\sigma_i = K \hookrightarrow \mathbb{C}$ that fix every element of $\mathbb{Q}$. Concretely, these embeddings map ζ to each of its conjugates; it can be verified that these are the only field homomorphisms from K to $\mathbb{C}$ because ζ's conjugates are the only roots of ζ's minimal polynomial $f(x)$. An embedding whose image lies in $\mathbb{R}$ (corresponding to a real root of $f(x)$) is called a *real embedding*; otherwise (for a complex root of $f(x)$) it is called a *complex embedding*. Because complex roots of $f(x)$ come in conjugate pairs, so too do the complex embeddings. The number of real embeddings is denoted s_1 and the number of *pairs* of complex embeddings is denoted by s_2, so we have $n = s_1 + 2s_2$. The pair (s_1, s_2) is called the *signature* of K. By convention, we let $\{\sigma_j\}_{j\in[s_1]}$ be the real embeddings, and order the complex embeddings so that $\sigma_{s_1+s_2+j} = \overline{\sigma_{s_1+j}}$ for $j \in [s_2]$. The *canonical embedding* $\sigma : K \to \mathbb{R}^{s_1} \times \mathbb{C}^{2s_2}$ is defined as

$$\sigma(x) = (\sigma_1(x), \ldots, \sigma_n(x)).$$

The canonical embedding σ is a field homomorphism from K to $\mathbb{R}^{s_1} \times \mathbb{C}^{2s_2}$, where multiplication and addition in $\mathbb{R}^{s_1} \times \mathbb{C}^{2s_2}$ are component-wise (since σ is a ring homomorphism). Due to the pairing of the complex embeddings, σ maps into the following space $H \subseteq \mathbb{R}^{s_1} \times \mathbb{C}^{2s_2} \subset \mathbb{C}^n$:

$$H = \{(x_1, \ldots, x_n) \in \mathbb{R}^{s_1} \times \mathbb{C}^{2s_2} : x_{s_1+s_2+j} = \overline{x_{s_1+j}}, \forall j \in [s_2]\}.$$

By identifying elements of K with their canonical embeddings in H, we can speak of geometric *canonical norms* on K. Specifically, we define the ℓ_p canonical norm of x, denoted as $\|x\|_p^{can}$ as $\|\sigma(x)\|_p = \left(\sum_{i\in[n]} |\sigma_i(x)|^p\right)^{\frac{1}{p}}$ for $p < \infty$, and as $\max_{i\in[n]} |\sigma_i(x)|$ for $p = \infty$. (As always we assume the ℓ_2 norm when p is omitted.)

[1] Recall that a number field $K = Q(\zeta)$ is isomorphic to $\mathbb{Q}[x]/f(x)$ where $f(x)$ is the minimal polynomial of ζ. The degree of K defined to be the value $[K : \mathbb{Q}]$, is same as [Ste04, p 28] the degree of the polynomial $f(x)$. (More generally, if $K \subset L$ are number fields, we let $[L : K]$ denote the dimension of L viewed as a K-vector space.)

Field Norm. The (field) *norm* of an element $a \in K$ is defined as $\mathsf{N}(a) = \mathsf{N}_{K/\mathbb{Q}}(a) = \prod_{i\in[n]} \sigma_i(a)$.[2] Note that the [Wes99, p 43, proof of Lemma 3.2] norm of an algebraic integer is in $\mathbb{Z}$.

Coefficient Embedding. There is also a *coefficient embedding* $\tau : K \to \mathbb{Q}^n$. As mentioned earlier, since $f(\zeta) = 0$, there is an isomorphism between $\mathbb{Q}[x] \mod f(x)$ and K given by $x \to \zeta$. So, K can be represented as a n-dimensional vector space over $\mathbb{Q}$ using the *power basis* $\{1, \zeta, \dots, \zeta^{n-1}\}$, and τ maps an element of K to its associated coefficient vector. When identifying an element $a \in K$ as a coefficient vector, i.e., $\tau(a)$ we denote it as a boldface vector $\mathbf{a}$. Note that the addition of vectors is done component-wise, while the multiplication is done as polynomials modulo $f(x)$. We define the *coefficient norm* of a as the norm of the vector $\mathbf{a}$. Specifically, we define the ℓ_p coefficient norm of a, denoted as $\|a\|_p$ or $\|\mathbf{a}\|_p$ as $\left(\sum_{i\in[n]} a_i^p\right)^{\frac{1}{p}}$ for $p < \infty$, and as $\max_{i\in[n]} |a_i|$ for $p = \infty$. (As always we assume the ℓ_2 norm when p is omitted.)

Relationship between Coefficient and Canonical Embeddings. The conversion of an element in $K = \mathbb{Q}[\zeta_m]$ $(n = \phi(m))$ from its coefficient representation to the canonical one can be seen as the multiplication of the coefficients of the polynomial by a specific Vandermonde matrix. More specifically, if $\mathbf{a}$ is an element of K in the coefficient representation then $V_m \cdot \mathbf{a}$ is exactly the canonical representation where $V_m \in \mathbb{C}^{n\times n}$ such that its i^{th} row is the vector $(1, \zeta_m^{j_i}, \zeta_m^{2j_i}, \dots, \zeta_m^{(n-1)j_i})$ for all $j_i \in \mathbb{Z}_m^*$. The matrix V_m when m is a power of 2 is special in the sense that the matrix $\frac{1}{n}V_m$ is unitary. This means that conversions between the canonical embedding and the coefficient representation corresponds to just a rigid rotation and a scaling.

Multiplicative Expansion Factor. We define the *multiplicative expansion factor* γ_{Mult} to denote (as in [Gen09a, p. 71]) the maximal value of $\frac{\|\mathbf{a}\times\mathbf{b}\|}{\|\mathbf{a}\|\cdot\|\mathbf{b}\|}$ for any $\boldsymbol{a}, \boldsymbol{b} \in K$. (See [LM06] for a different definition of the expansion factor for multiplication.) The dependence of γ_{Mult} value on the underlying field K is understood.

Next we will argue (also see [Gen09a, Lemma 7.4.3] and [GH10, Section 2.2]) that for the field $K = \mathbb{Q}[x]/(x^n + 1)$, γ_{Mult} can be upper bounded by $\sqrt{n}$.

Lemma 5.9. *Let $K = \mathbb{Q}[x]/(x^n + 1)$, for any positive integer n. $\forall \boldsymbol{a}, \boldsymbol{b} \in K$ and $\boldsymbol{c} = \boldsymbol{a} \times \boldsymbol{b}$ we have that*

$$\|\boldsymbol{c}\| \leq \sqrt{n} \cdot \|\boldsymbol{a}\| \cdot \|\boldsymbol{b}\|.$$

Proof. Consider the ith coefficient c_i of $\boldsymbol{c}$. First observe that for each i, c_i is obtained as a dot product of $\boldsymbol{a}$ and some reordering of entries of $\boldsymbol{b}$ (additionally the signs of some entries can also be reversed). Therefore we have $c_i \leq \|\boldsymbol{a}\| \cdot \|\boldsymbol{b}\|$. This allows us to conclude that $\|\boldsymbol{c}\| \leq \sqrt{n} \cdot \|\boldsymbol{a}\| \cdot \|\boldsymbol{b}\|$. □

[2] More generally, the *relative norm* $\mathsf{N}_{K/L}(a)$ of an element $a \in K$ over a subfield $L \subset K$ is $\prod_{\sigma_i \in S} \sigma_i(a)$, where S consists of the K-embeddings σ_i that fix every element in L.

Example. Continuing with our example of the mth cyclotomic number field where $K = \mathbb{Q}(\zeta_m)$ for $m > 2$, there are $2s_2 = n = \varphi(m)$ complex canonical embeddings (and no real ones), which are given by $\sigma_i(\zeta_m) = \zeta_m^i$ for $i \in \mathbb{Z}_m^*$. (It is convenient to index the embeddings by elements of $\mathbb{Z}_m^*$ instead of $[n]$.) For an element $x = \zeta^j \in K$ in the power basis of K, all the embeddings of x have magnitude 1, and hence $\|x\|_2^{can} = \sqrt{n}$ and $\|x\|_\infty^{can} = 1$. Also considering the coefficient embedding $\|x\|_2 = 1$.

5.3 Ideals in the Ring of Integers

The ring of integers $\mathcal{O}_K$, of a number field K of degree n, is a free $\mathbb{Z}$-module (see [Wes99, p 39, Theorem 2.22]) of rank n, i.e., the set of all $\mathbb{Z}$-linear combinations of some *integral basis* $\{b_1, \dots, b_n\} \subset \mathcal{O}_K$. Such a set is called an *integral basis*, and it is also a $\mathbb{Q}$-basis for K. As usual, there are infinitely many such bases when $n > 1$.

Continuing with our example of the mth cyclotomic number field $K = \mathbb{Q}(\zeta_m)$ of degree $n = \varphi(m)$, the power basis $\{1, \zeta_m, \dots, \zeta_m^{n-1}\}$ of K also happens to be an integral basis of the *cyclotomic ring* $\mathcal{O}_K = \mathbb{Z}[\zeta_m]$. (In general, it is unusual for the power basis of a number field to generate the entire ring of integers.)

Definition 5.10 (Ideal)**.** *An* (integral) ideal $\mathcal{I} \subseteq \mathcal{O}_K$ *is a nontrivial (i.e., nonempty and nonzero*[3]*) additive subgroup that is closed under multiplication by* $\mathcal{O}_K$ *– that is,* $r \cdot g \in \mathcal{I}$ *for any* $r \in \mathcal{O}_K$ *and* $g \in \mathcal{I}$*. A* fractional ideal $\mathcal{I} \subset K$ *is a set such that* $d \cdot \mathcal{I}$ *is an integral ideal for some* $d \in \mathcal{O}_K$*. The* inverse $\mathcal{I}^{-1}$ *of an ideal* $\mathcal{I}$ *is the set* $\{a \in K : a \cdot \mathcal{I} \subseteq \mathcal{O}_K\}$*.*

An ideal $\mathcal{I}$ in $\mathcal{O}_K$ is finitely generated as the set of all K-linear combinations of some *generators* $g_1, g_2, \dots \in \mathcal{O}_K$, denoted $\mathcal{I} = \langle g_1, g_2, \dots \rangle$. (In fact, it is know that two generators [Ste04, Proposition 9.1.7] always suffice.)

Definition 5.11. *An ideal* $\mathcal{I}$ *is* principal *if* $\mathcal{I} = \langle g \rangle$ *for* $g \in \mathcal{O}_K$ *– that is, if one generator suffices.*

More useful to us is the fact [Oss08, Proposition 1.6.1] that an ideal (integral or fractional) is also a free $\mathbb{Z}$-module of rank n, i.e., it is generated as the set of all $\mathbb{Z}$-linear combinations of some basis $\{b_1, \dots, b_n\} \subset \mathcal{O}_K$.

Definition 5.12. *Let* $\mathcal{I}, \mathcal{J}$ *be ideal of a ring* R*. Their* sum *is the ideal*

$$\mathcal{I} + \mathcal{J} = \{a + b : a \in \mathcal{I}, b \in \mathcal{J}\}$$

and their product $\mathcal{I}\mathcal{J}$ *is ideal generated by all products of elements in* $\mathcal{I}$ *with elements in* $\mathcal{J}$*, or*

$$\mathcal{I}\mathcal{J} = \langle a \cdot b : a \in \mathcal{I}, b \in \mathcal{J} \rangle.$$

Two ideals $\mathcal{I}, \mathcal{J} \subseteq \mathcal{O}_K$ are said to be *coprime* (or *relatively prime*) if $\mathcal{I} + \mathcal{J} = \mathcal{O}_K$.

[3] *Some texts also define the trivial set* $\{0\}$ *as an ideal, but in this work it is more convenient to exclude it.*

5.4 Prime Ideals - Unique Factorization and Distributions

In this section we will define the notion of prime ideals and recall some of their properties. A prime ideal shares many important properties of a prime number in $\mathbb{Z}$.

Definition 5.13. *An ideal $\mathfrak{p} \subsetneq \mathcal{O}_K$ is* prime *if whenever $a, b \in \mathcal{O}_K$ and $ab \in \mathfrak{p}$ then either $a \in \mathfrak{p}$ or $b \in \mathfrak{p}$.*

Unique Factorization. As per unique-prime-factorization theorem, we have that every integer greater than 1 is either prime itself or is the product of prime numbers. Similar in any ring of integers $\mathcal{O}_K$ of the number field K has unique factorization of ideals into prime ideals.

Proposition 5.14 (Unique Factorization of Ideals [Ste04, Theorem 6.1.9])**.** *Suppose $\mathcal{I}$ is an integral ideal of $\mathcal{O}_K$. Then $\mathcal{I}$ can be written as a product*

$$\mathcal{I} = \mathfrak{p}_1 \dots \mathfrak{p}_n$$

of prime ideals of $\mathcal{O}_K$, and this representation is unique up to order.

Ideal Norm and some of its properties. Now we will define the norm of an ideal and mention some of the properties about the norms of prime ideals.

Definition 5.15. *If $\mathcal{I}$ is an ideal of a ring of integers $\mathcal{O}_K$, we define the* norm *of $\mathcal{I}$ to be*

$$\mathsf{N}(\mathcal{I}) = |\mathcal{O}_K/\mathcal{I}|$$

where $|\mathcal{O}_K/\mathcal{I}|$ dentes the size of the quotient ring $\mathcal{O}_K/\mathcal{I}$.

It is know that [Wes99, p 60, Lemma 2.2] $\mathsf{N}(\mathcal{I}\mathcal{J}) = \mathsf{N}(\mathcal{I})\mathsf{N}(\mathcal{J})$.

In $\mathcal{O}_K$, an [Ste04, Proposition 6.1.4] ideal $\mathfrak{p}$ is prime if and only if it is *maximal*, i.e., if the only proper superideal of $\mathfrak{p}$ is $\mathcal{O}_K$ itself, which implies that the quotient ring $\mathcal{O}_K/\mathfrak{p}$ is a finite field of order $\mathsf{N}(\mathfrak{p})$.

Proposition 5.16 ([Oss08, Corollary 1.6.9])**.** *For a in a ring of integers $\mathcal{O}_K$, let $\mathfrak{p} = \langle a \rangle$ be the principal ideal generated by a, then we have that $\mathsf{N}(\mathcal{I}) = |\mathsf{N}(a)|$.*

Suppose $\mathfrak{p}$ is an ideal of a ring of integers $\mathcal{O}_K$, and $\mathsf{N}(\mathfrak{p}) = p$ for some prime integer $p \in \mathbb{Z}$. Then we have that [Oss08, Lemma 1.6.7] $\mathfrak{p}$ is prime in $\mathcal{O}_K$. Note that, many prime ideals do not have prime norms. In fact [Oss08, Lemma 4.6.1] if $\mathfrak{p}$ is a prime ideal in a ring of integers $\mathcal{O}_K$, then $\mathsf{N}(\mathfrak{p}) = p^n$ for some prime $p \in \mathbb{Z}$ and $n \in \mathbb{N}$.

Distribution of Prime ideals. The distribution of prime ideals in number fields is quite analogous to the distribution of primes in the integers. Just as the prime number theorem states that the number of primes less than x is approximately $x/\ln x$, we have Landau's prime ideal theorem.

Theorem 5.17 (Landau's prime number theorem [BS96, Theorem 8.7.2])**.** *Let K be an algebraic number field of degree n. Let $\pi_K(x)$ denote the number of prime ideals whose norm is $\leq x$. Let $\xi(x) = (\ln x)^{3/5}(\ln\ln x)^{-1/5}$. There is a $c > 0$ (depending on K) such that*

$$\pi_K(x) = Li(x) + O(xe^{-c\xi(x)}) \sim \frac{x}{\ln x},$$

where $Li(x) = \int_2^t \frac{dt}{\ln t}$.

Furthermore the prime ideals in the above theorem are dominated by the ideals of norm a prime number. Assuming the Generalized Riemann Hypothesis (GRH) [BS96, Conjecture 8.7.3], a stronger statement [BS96, Theorem 8.7.4] can be made but the above mentioned unconditional statement suffices for our purposes.

In our constructions we will need results on the distribution of prime ideals that are also principal. From prime number theorem for arithmetic progressions, we know that the number of primes less that or equal to x and congruent to $a \mod n$ (where a and n are co-prime), is $x/(\phi(n)\ln x)$. Similarly one of the consequences of Chebotarëv's density[4] theorem (see for example [Ste10, Proof of Lemma 4]) is that the among all the prime ideals in a number field K, $\frac{1}{h}$ of them are principal, where h is the class number of K.

We refer the reader to [Lan90, p 77] for a general analytic formula for computing the class number of any number field K. The class number[5] of the n-th cyclotomic field K, factors as h^+ times h^-, where h^+ is the class number of the maximal real subfield of K. The Brauer-Siegel theorem (see [Was97, Theorem 4.20]) implies that $\log(h^-)$ grows roughly as $\frac{1}{4}\phi(n)\log n$ as $n \to \infty$. However, h^+ tends to be rather small. For n a power of 2, it is conjectured that h^+ =1. This is true for $n = 2^k$ with $k \leq 7$, and also for $k = 8$ if we assume GRH. This provides for theoretical evidence that principal prime ideals exist. However since the class number is already exponential this does not suffice for our purposes.

Nevertheless restricting the Landau's prime number theorem to principal ideals we can heuristically expect that with noticeable probability a random principal ideal will have a prime norm.

Conjecture 5.18. *Let K be the n-th cyclotomic field for n a power of 2. For every $\sigma = \mathsf{poly}(n)$ there is a constant $c > 1$ such that for sufficiently large n we have that*

$$\Pr_{f \leftarrow D_{Z^n,\sigma}}[\mathsf{N}(f) \geq 2^{O(n)} \textit{ and is prime}] \geq \frac{1}{n^c}.$$

[4] Just like Landau's prime number theorem is a generalization of the prime number theorem, we have the Chebotarëv's density theorem [BS96, Theorem 8.7.9] with generalizes the prime number theorem for arithmetic progressions [BS96, Theorem 8.4.2] to number fields. Chebotarëv's density theorem is a very technical result building on field theory and we do not delve into stating it formally. We refer the reader to [SL96] for a very down to earth introduction to Chebotarëv's Density Theorem.

[5] We would like to thank Alice Silverberg and Lawrence Washington for pointing [SW13] these facts about class number of cyclotomic fields to us.

Smart and Vercauteren [SV10] and Boneh and Freeman [BF11a] follow a similar heuristic in their applications. Experimental results supporting this heuristic have been provided by Smart and Vercauteren [SV10].

5.5 Ideal Lattices

Recall that a number field $K = Q(\zeta)$ is isomorphic to $\mathbb{Q}[x]/f(x)$ where $f(x)$ is the minimal polynomial of ζ. Also recall that any ideal $\mathcal{I}$ of $\mathcal{O}_K$ is a free $\mathbb{Z}$-module, i.e., it is generated as the set of all $\mathbb{Z}$-linear combinations of some basis $B = \{b_1, \ldots, b_n\} \subset \mathcal{O}_K$. Therefore under the coefficient embedding τ, the ideal $\mathcal{I}$ of $\mathcal{O}_K$ yields a rank-n lattice $\tau(\mathcal{I})$ having basis $\{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$, where each $\boldsymbol{b}_i = \tau(b_i)$. Obviously, addition is done component-wise in the coefficients, and multiplication is polynomial multiplication modulo the polynomial $f(x)$. We call $\mathcal{I}$ an *ideal lattice* to stress its dual interpretation as both an ideal and a lattice. When visualizing it as a lattice we speak of, e.g., the minimum distance $\lambda_1(\mathcal{I})$ of an ideal, etc.

As pointed out earlier the mth cyclotomic ring with $n = \varphi(m)$ happens to be exactly $\mathbb{Z}[\zeta_m]$ which corresponds to the lattice $\mathbb{Z}^n$.

Proposition 5.19 ([LPR12, p 22])**.** *For any ideal $\mathcal{I}$ of the mth cyclotomic ring (with $n = \varphi(m)$) we have $\lambda_n(\mathcal{I}) = \lambda_1(\mathcal{I})$.*

We will sketch the argument here. Consider the mth cyclotomic field such that $n = \varphi(m)$. Observe that multiplying a shortest nonzero element $\boldsymbol{v} \in \mathcal{I}$ by $1, \zeta, \ldots, \zeta^{n-1}$ gives n linearly independent elements of the same length. This allows us to conclude the above proposition.

Invertibility of ring elements. Let R denote the $2n^{th}$ cyclotomic ring and let R_q denote R/qR for a prime q. We note that R_q is also a ring and not all elements in it are invertible. Let $R_q^\times$ denote the set of elements in R_q that are invertible. We next provide a lemma of Stehlé and Steinfeld that points out that a (large enough) random element is R_q is also in $R_q^\times$ with large probability.

Lemma 5.20 ([SS11, Lemma 4.1])**.** *Let $n \geq 8$ be a power of 2 such that $x^n + 1$ splits into n linear factors modulo $q \geq 5$. Let $\sigma \geq \sqrt{n \ln(2n(1 + 1/\delta))/\pi} \cdot q^{1/n}$, for an arbitrary $\delta \in (0, 1/2)$. Then*

$$\Pr_{f \leftarrow D_{Z^n,\sigma}}[f \mod q \notin R_q^\times] \leq n(1/q + 2\delta).$$

CHAPTER 6

The New Encoding Schemes

We will first describe our system for the "symmetric setting" (i.e. corresponding to Definition 3.2 in Section 3.2.) Later in Section 6.3 we explain how to handle the general case (Definition A.3 in Appendix A). There we will also consider other extensions. In this chapter we focus on functionality, leaving much of the discussion on security considerations to Chapter 7.

An instance of our basic construction is parameterized by the security parameter λ and the required multi-linearity level $\kappa \leq$poly(λ). Based on these parameters, we choose the $2n$th cyclotomic ring $R = \mathbb{Z}[x]/(x^n + 1)$ where n is a power of 2 (n is set large enough to ensure security), a modulus q that defines $R_q = R/qR$ (with q large enough to support functionality), and another parameter m (chosen so that we can apply Theorem 4.8). The specific constraints that these parameters must satisfy are discussed in Section 6.2, an approximate setting to keep in mind is $n = \tilde{O}(\kappa\lambda^2)$, $q = 2^{\kappa\lambda}$ and $m = O(n^2)$.

6.1 The Basic Graded Encoding Scheme

We start by giving some intuition behind our scheme. An instance of our scheme relative to the parameters above encodes elements of a quotient ring $QR = R/\mathcal{I}$, where $\mathcal{I}$ is a principal prime ideal $\mathcal{I} = \langle \mathbf{g} \rangle \subset R$, generated by a "short" vector $\mathbf{g}$. Namely, the "ring elements" that are encoded in our scheme are cosets of the form $\boldsymbol{e} + \mathcal{I}$ for some vector $\boldsymbol{e}$. The short generator $\mathbf{g}$ itself is kept secret, and no "good" description of $\mathcal{I}$ is made public in our scheme. In addition, our system depends on another secret element $\mathbf{z}$, which is chosen at random in R_q (and hence is not short).

A level-zero ("plaintext") encoding of a coset $\boldsymbol{e} + \mathcal{I} \in R/\mathcal{I}$ is just a short vector in that coset (which must exist, since the generator $\mathbf{g}$ is short and therefore the basic cell of $\mathcal{I}$ is quite small). For higher-level encodings, a level-i encoding of the same coset is a vector of the form $\boldsymbol{c}/\mathbf{z}^i \in R_q$ with $\boldsymbol{c} \in \boldsymbol{e} + \mathcal{I}$ short. Specifically, for $i \in \{0, 1, \ldots, \kappa\}$ the set of all level-i encodings is $S_i = \{\boldsymbol{c}/\mathbf{z}^i \in R_q : \|\boldsymbol{c}\| < q^{1/8}\}$, and the set of level-$i$ encodings of the "plaintext element" $\boldsymbol{e} + \mathcal{I}$ is $S_i^{(\boldsymbol{e}+\mathcal{I})} = \{\boldsymbol{c}/\mathbf{z}^i \in R_q : \boldsymbol{c} \in \boldsymbol{e} + \mathcal{I},\ \|\boldsymbol{c}\| < q^{1/8}\ \}$. Throughout the construction we use the size of the numerator as the "noise level" in the encoding. Namely, with each level-i encoding $\boldsymbol{c}/\mathbf{z}^i$ we produce also an upper bound on $\|\boldsymbol{c}\|$.

Instance generation: $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^\kappa)$. Our instance-generation procedure chooses at random the ideal-generator $\mathbf{g}$ and denominator $\mathbf{z}$, as well as several other vectors that are used in the other procedures and are described later in the section. The denominator $\mathbf{z}$ is chosen uniformly at random in R_q, and hence is not "small" with overwhelming probability. Using Lemma 5.20 we have that $\mathbf{z}$ is invertible in R_q with overwhelming probability.

We simply draw $\mathbf{g}$ from a discrete Gaussian over $\mathbb{Z}^n$, say $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$ with $\sigma = \sqrt{\lambda n}$ repeatedly till we have that:

(i) $\|\mathbf{g}\| \le \sigma\sqrt{n}$ and $\mathbf{g}$ is invertible in R_q.

(ii) $\|\mathbf{g}^{-1}\| \le n^{c+1.5}$ (in K) for an appropriate constant c. (Recall that we denote $K = \mathbb{Q}[x]/(x^n+1)$. The reason that we need $\mathbf{g}^{-1} \in K$ to be short is explained when we describe the zero-testing procedure.)

(iii) $\mathsf{N}(\mathbf{g})$ is a prime $\ge 2^{O(n)}$.

From Lemma 6.1 we can conclude that the above described rejection sampling procedure succeeds in polynomially many trials. Condition (iii) from above, Proposition 5.16 and the discussion there after imply that $\mathcal{I} = \langle \mathbf{g} \rangle$ is a principal prime ideal.

Once we have $\mathbf{g}, \mathbf{z}$, we choose and publish some other elements in R_q that will be used for the various procedures below. Specifically we have $m+1$ elements $\mathbf{x}_1, \ldots, \mathbf{x}_m, \mathbf{y}$ that are used for encoding, and an element $\mathbf{p}_{zt}$ that is used as a zero-testing parameter. These elements are described later. Finally we also choose a random seed s for a strong randomness extractor. The instance-generation procedure outputs $\mathsf{params} = (n, q, \mathbf{y}, \{\mathbf{x}_i\}_i, s)$ and $\mathbf{p}_{zt}$.

Lemma 6.1. *If $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$, then assuming Conjecture 5.18 there exists a constant c such that* (i), (ii) *and* (iii) *from above are simultaneously satisfied with a noticeable probability.*

Proof. We will proceed by obtaining bounds on probabilities that each of the above conditions (i), (ii) and (iii) individually holds. Subsequently the lemma follows by a union bound argument.

(i) It follows directly from Lemma 4.5 and Lemma 5.20 that condition (i) is satisfied with overwhelming probability.

(ii) Now we argue that with good probability $\mathbf{g}^{-1}$ in the field of fractions K is also rather short. We will argue this by looking at $\mathbf{g}$ in terms of the canonical embedding. As pointed in Section 5.2, the canonical embedding representation can be obtained by multiplying the coefficient representation with the matrix V_{2n}. And this transformation for a power of 2 cyclotomic corresponds to just a rigid rotation and a scaling (thus the spherical Gaussian distribution is not affected by the transformation). Therefore we have that sampling $\mathbf{g}$ from $D_{\mathbb{Z}^n,\sigma}$ and considering the canonical embedding is the same as sampling directly the canonical representation for an appropriately scaled gaussian parameter σ', which in our case is at least $\omega(1)$. This implies that roughly with probability $1-o(1/n^{c+1})$, evaluating $\mathbf{g}$ at any complex $2n$'th root of unity $\zeta \in \mathbb{C}$ yields $\mathbf{g}(\zeta)$ which is greater than $1/n^{c+1}$.

Next by taking a union bound, with probability $1-o(1/n^c)$ we have $\mathbf{g}^{-1}(\zeta) = 1/\mathbf{g}(\zeta) < n^{c+1}$ for all the primitive $2n$'th roots of unity ζ, which means that $\|\mathbf{g}^{-1}\|_\infty^{can} < n^{c+1}$. This implies an upper bound of $\|\mathbf{g}^{-1}\|_\infty < n^{c+1}$ as well (because for every $\boldsymbol{a} \in K$ we have that $\|\boldsymbol{a}\|_\infty \leq \|\boldsymbol{a}\|_\infty^{can}$; see for example [DPSZ11, Theorem 7 and Discussion on p. 39] for a detailed proof). Hence a bound of $\|\mathbf{g}^{-1}\| < n^{c+1.5}$.

(iii) Conjecture 5.18 implies that there exists a constant c such that condition (iii) is satisfied with probability at least $\frac{1}{n^c}$.

Putting the above bounds together and taking a union bound implies the claimed lemma. □

Sampling level-zero encodings: $\boldsymbol{d} \leftarrow \mathsf{samp}(\mathsf{params})$. To sample a level-zero encoding of a random coset, we just draw a random short element in R, $\boldsymbol{d} \leftarrow D_{\mathbb{Z}^n,\sigma'}$, where $\sigma' = \sigma n\sqrt{\lambda}$ (for σ that was used to sample $\mathbf{g}$). In Lemma 6.2 we argue that the sampled value $\boldsymbol{d}$ corresponds to a random coset of $\mathcal{I}$. Finally note that by Lemma 4.6 the size of this level-zero encoding is bounded by $\sigma'\sqrt{n}$ (and we use this as our noise-bound for this encoding).

Lemma 6.2. *Let $\mathcal{I} = \langle \mathbf{g} \rangle$ and $\sigma' \geq \sqrt{\lambda n}\|\mathbf{g}\|$, then we have that the distribution $\boldsymbol{d} \mod \mathcal{I}$ where $\boldsymbol{d} \leftarrow D_{\mathbb{Z}^n,\sigma'}$ is close to uniform over $\mathbb{Z}^n \mod \mathcal{I}$, up to negligible distance.*

Proof. We can safely assume that $\lambda_1(\mathcal{I}) \leq \|\mathbf{g}\|$. Next according to Proposition 5.19 we have that $\lambda_n(\mathcal{I}) = \lambda_1(\mathcal{I})$. This along with Lemma 4.2 allows us to conclude that with overwhelming probability

$$\begin{aligned}\eta_{2^{-\lambda}}(\mathcal{I}) &\leq \sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \|\mathbf{g}\| \\ &\leq \sqrt{\frac{\ln(2n(1+1/\epsilon))}{\pi}} \cdot \|\mathbf{g}\| \\ &\leq \sqrt{\lambda n}\|\mathbf{g}\|\end{aligned}$$

Finally since we have that $\sigma' \geq \eta_{2^{-\lambda}}(\mathcal{I})$, therefore by Corollary 4.4 we can conclude that the induced distribution over the cosets of $\mathcal{I}$ is close to uniform, up to a negligible distance. □

Encodings at higher levels: $\boldsymbol{u}_i \leftarrow \mathsf{enc}(\mathsf{params}, i, \boldsymbol{d})$. To allow encoding of cosets at higher levels, we publish as part of our instance-generation a level-one encoding of $1+\mathcal{I}$, namely an element $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ where $\mathbf{a} \in 1+\mathcal{I}$ is short. A simplistic method of doing that is drawing $\mathbf{a} \leftarrow D_{1+\mathcal{I},\sigma''}$, for some parameter σ'', then computing $\mathbf{y}$ from $\mathbf{a}$. (Later we describe a somewhat more involved procedure, which we believe is more secure, see details in Section 7.4.) Given a level-zero encoding $\boldsymbol{d}$ as above, we can multiply it by $\mathbf{y}$ over R_q to get $\boldsymbol{u}_1 := [\mathbf{y}\boldsymbol{d}]_q$. (We use the notation $[\cdot]_q$ to denote operations in R_q.) Note that $\boldsymbol{u}_1 = [\boldsymbol{d}\mathbf{a}/\mathbf{z}]_q$, where $\boldsymbol{d}\mathbf{a} \in \boldsymbol{d} + \mathcal{I}$ as needed. Note that the size of the numerator $\boldsymbol{d}\mathbf{a}$ of $\boldsymbol{u}_1$ can be bounded by $\gamma_{\mathsf{Mult}}\|\boldsymbol{d}\| \cdot \|\mathbf{a}\|$ (recall that γ_{Mult} can be bounded by $\sqrt{n}$ using Lemma 5.9) and we use this as our noise-bound for this encoding. More generally we can generate a level-i encoding as $\boldsymbol{u}_i := [\boldsymbol{d}\mathbf{y}^i]_q = [\boldsymbol{d}\mathbf{a}^i/\mathbf{z}^i]_q$. The numerator $\boldsymbol{d}\mathbf{a}^i$ is obviously in $\boldsymbol{d}+\mathcal{I}$, and its size can again be bounded (using Lemma 5.9) by $\gamma_{\mathsf{Mult}}^{i/2}\|\boldsymbol{d}\| \cdot \|\mathbf{a}\|^i$.

The above encoding is insufficient, however, since from $\boldsymbol{u}_1$ and $\mathbf{y}$ it is easy to get back $\boldsymbol{d}$ by simple division in R_q. We therefore include in the public parameters also the "randomizers" $\mathbf{x}_i$, these are just random encodings of zero, namely $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$ where the $\mathbf{b}_i$'s are short elements in $\mathcal{I}$. A simplistic procedure for choosing these randomizers would be to draw these elements as $\mathbf{b}_i \leftarrow D_{\mathcal{I},\sigma'''}$ (where σ''' will be set later so that we can use Theorem 4.8) and publish $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$. (Later we describe a somewhat more involved procedure, which we believe is more secure, see details in Section 7.4.) Below we denote by $\mathbf{X}$ the matrix with the vectors $\mathbf{x}_i$ as rows, namely $\mathbf{X} = (\mathbf{x}_1|\ldots|\mathbf{x}_m)^T$. We also use $\mathbf{B}$ to denote the matrix with the numerators $\mathbf{b}_i$ as rows, i.e., $\mathbf{B} = (\mathbf{b}_1|\ldots|\mathbf{b}_m)^T$.

We use the $\mathbf{x}_i$'s to randomize level-one encodings: Given $\boldsymbol{u}' = [\boldsymbol{c}'/\mathbf{z}]_q$ with noise-bound $\|\boldsymbol{c}'\| < \gamma$, we draw an m-vector of integer coefficients $\boldsymbol{r} \leftarrow D_{\mathbb{Z}^m,\sigma^*}$ for large enough σ^* (e.g. $\sigma^* = 2^\lambda\gamma$), and output

$$\boldsymbol{u} \;:=\; [\boldsymbol{u}' + \mathbf{X}\boldsymbol{r}]_q \;=\; \left[\boldsymbol{u}' + \sum_{i=1}^{m} r_i\mathbf{x}_i\right]_q \quad \left(=\; \left[\frac{\boldsymbol{c}' + \sum_i r_i\mathbf{b}_i}{\mathbf{z}}\right]_q\right).$$

We write $\mathbf{B}\boldsymbol{r}$ as a shorthand for $\sum_i r_i\mathbf{b}_i$ and similarly $\mathbf{X}\boldsymbol{r}$ as a shorthand for $\sum_i r_i\mathbf{x}_i$.

Since all the $\mathbf{b}_i$'s are in the ideal $\mathcal{I}$, then clearly $\boldsymbol{c}' + \sum_i r_i\mathbf{b}_i$ is in the same coset of $\mathcal{I}$ as $\boldsymbol{c}'$ itself. Moreover since (using Lemma 4.9) $\|\mathbf{b}_i\| < \mathrm{poly}(n,m)$ therefore we have that $\|\mathbf{B}\boldsymbol{r}\| < \sigma^*\mathrm{poly}(m,n)$. If indeed $\|\boldsymbol{c}'\| < \gamma$, then we can conclude that $\|\boldsymbol{c}' + \mathbf{B}\boldsymbol{r}\| < \gamma + \sigma^*\mathrm{poly}(m,n)$ (and we use this as our noise-bound for this encoding.)

We also claim that the distribution of $\boldsymbol{u}$ is nearly independent of original $\boldsymbol{u}'$ (except of course its coset). To see why, note that if the $\mathbf{b}_i$'s are chosen from a wide enough spherical distribution (specifying a constraint on σ''') then we can use Theorem 4.8 to conclude that $\mathbf{B}\boldsymbol{r}$ is close to a wide ellipsoid Gaussian. With our choice of σ^* the "width" of that distribution is much larger than the original $\boldsymbol{c}'$, hence the distribution of $\boldsymbol{c}' + \mathbf{B}\boldsymbol{r}$ is nearly independent of $\boldsymbol{c}'$, except in the coset that it belongs to. In particular for this to work we will need σ^* to be super-polynomially larger than the noise bound of $\boldsymbol{c}'$.

Adding and multiplying encodings. It is easy to see that the encoding as above is additively homomorphic, in the sense that adding encodings yields an encoding of the sum.

This follows since if we have many short $\boldsymbol{c}_j$'s then their sum is still short, $\|\sum_j \boldsymbol{c}_j\| \ll q$, and therefore the sum $\boldsymbol{c} = \sum_j \boldsymbol{c}_j = [\sum_j \boldsymbol{c}_j]_q \in R_q$ belong to the coset $\sum_j(\boldsymbol{c}_j + \mathcal{I})$. Hence, if we denote $\boldsymbol{u}_j = \boldsymbol{c}_j/\mathbf{z} \in R_q$ then each $\boldsymbol{u}_j$ is an encoding of the coset $\boldsymbol{c}_j + \mathcal{I}$, and the sum $[\sum_j \boldsymbol{u}_j]_q$ is of the form $\boldsymbol{c}/\mathbf{z}$ where $\boldsymbol{c}$ is still a short element in the sum of the cosets.

Moreover, since $\mathcal{I}$ is an ideal then multiplying upto κ encodings can be interpreted as an encoding of the product, by raising the denominator to the appropriate power. Namely, for $\boldsymbol{u}_j = \boldsymbol{c}_j/\mathbf{z} \in R_q$ as above, we have

$$\boldsymbol{u} = \left[\prod_{j=1}^{\kappa} \boldsymbol{u}_j\right]_q = \left[\frac{\prod_j \boldsymbol{c}_j}{\mathbf{z}^\kappa}\right]_q.$$

As long as the $\boldsymbol{c}_j$'s are small enough to begin with, we still have $\|\prod_j \boldsymbol{c}_j\| \ll q$, which means that $[\prod_j \boldsymbol{c}_j]_q = \prod_j \boldsymbol{c}_j$ (where the product $\prod_j \boldsymbol{c}_j$ is computed in R), hence $[\prod_j \boldsymbol{c}_j]_q$ belongs to the product coset $\prod_j(\boldsymbol{c}_j + \mathcal{I})$.

Thus, if each $\boldsymbol{u}_j$ is a level-1 encoding of the coset $\boldsymbol{c}_j + \mathcal{I}$ with short-enough numerator, then their product is a level-κ encoding of the product coset. We note that just like level-1 encoding, level-κ encoding (and in fact any of the intermediate level encoding) also offers additive homomorphism.

Zero testing: $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}_\kappa) \overset{?}{=} 0/1$. Since the encoding is additively homomorphic, we can test equality between encodings by subtracting them and comparing to zero. To enable zero-testing, we generate the zero-testing parameter as follows: We draw a "somewhat small" ring element $\boldsymbol{h} \leftarrow D_{\mathbb{Z}^n,\sqrt{q}}$, such that $\boldsymbol{h} \notin \mathcal{I}$ and set the zero-testing parameter as $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$. (Later we describe a somewhat more involved procedure, which we believe is more secure, see details in Section 7.4.) To test if a level-κ encoding $\boldsymbol{u} = [\boldsymbol{c}/\mathbf{z}^\kappa]_q$ is an encoding of zero, we just multiply it in R_q by $\mathbf{p}_{zt}$ and check whether the resulting element $\boldsymbol{w} = [\mathbf{p}_{zt} \cdot \boldsymbol{u}]_q$ is short (e.g., shorter than $q^{3/4}$). Namely, we use the test

$$\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}) = \begin{cases} 1 & \text{if } \|[\mathbf{p}_{zt}\boldsymbol{u}]_q\|_\infty < q^{3/4} \\ 0 & \text{otherwise} \end{cases} \tag{6.1}$$

In Lemma 6.3 we will argue that encodings of zero (such that the numerator is less than $q^{1/8}$) always pass the zero test. Next in Lemma 6.5 we argue that encodings of non-zero cosets pass the zero test only with a negligible probability.

Lemma 6.3. *For any $\boldsymbol{u} = [\boldsymbol{c}/\mathbf{z}^\kappa]_q$ such that $\|\boldsymbol{c}\| < q^{1/8}$ and $\boldsymbol{c} \in \mathcal{I} = \langle \mathbf{g} \rangle$, such that $\|\mathbf{g}^{-1}\| < \frac{q^{1/8}}{n^{3/2}}$ (in K) we have that $\|[\mathbf{p}_{zt}\boldsymbol{u}]_q\|_\infty < q^{3/4}$ where $\boldsymbol{h} \leftarrow D_{\mathbb{Z}^n,\sqrt{q}}$, and $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$.*

Proof. To see why this works, we note that

$$\boldsymbol{w} = \mathbf{p}_{zt} \cdot \boldsymbol{u} = \frac{\boldsymbol{h}\mathbf{z}^\kappa}{\mathbf{g}} \cdot \frac{\boldsymbol{c}}{\mathbf{z}^\kappa} = \boldsymbol{h} \cdot \boldsymbol{c}/\mathbf{g} \quad \text{(all the operations in } R_q\text{)}.$$

If $\boldsymbol{u}$ is an encoding of zero then $\boldsymbol{c}$ is a short vector in $\mathcal{I}$ (containing elements $\mathbf{g}\boldsymbol{r}$ for $\boldsymbol{r} \in R$), which means that it is divisible by $\mathbf{g}$ in R. Hence the element $\boldsymbol{c}/\mathbf{g} \in R$ is the same as the element $\boldsymbol{c} \cdot \mathbf{g}^{-1} \in K$. Next we have that $\boldsymbol{c} \cdot \mathbf{g}^{-1}$ is at most $\|\boldsymbol{c}\| \cdot \|\mathbf{g}^{-1}\| \cdot \gamma_{\mathsf{Mult}}$ (recall that using Lemma 5.9 γ_{Mult} can be bounded $\sqrt{n}$). Next we have that $\|\boldsymbol{w}\| \leq \|\boldsymbol{h}\| \cdot \|\boldsymbol{c}\| \cdot \|\mathbf{g}^{-1}\| \cdot \gamma_{\mathsf{Mult}}^2$, which for our choice of parameter is $q^{1/2} \cdot \sqrt{n} \cdot q^{1/8} \cdot \|\mathbf{g}^{-1}\| \cdot n < q^{3/4}$ (Note that by Lemma 4.5 we have that $\|\boldsymbol{h}\| \leq q^{1/2} \cdot \sqrt{n}$ with overwhelming probability). This immediately also gives an upper bound on the ℓ_∞ norm of $\boldsymbol{w}$. □

If $\boldsymbol{u}$ is an encoding of a non-zero coset, then $\boldsymbol{c}$ is a short vector in some coset of $\mathcal{I}$. In this case we have $\boldsymbol{w} = [\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$, where $\boldsymbol{c}, \mathbf{g}$ are small (and $\boldsymbol{h}$ is "somewhat small"). Intuitively, since $[\boldsymbol{h}/\mathbf{g}]_q$ is large with high probability then for a "random enough" $\boldsymbol{c}$ we expect the size of $\boldsymbol{w}$ to be large. More formally, we argue below (Lemma 6.4) that when choosing a uniformly random coset of $\mathcal{I} = \langle \mathbf{g} \rangle$, there are *no short elements* $\mathbf{c}$ in that coset such that $[\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$ is small. This will allow up to conclude Lemma 6.5.

Lemma 6.4. *Let $\boldsymbol{w} = [\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$ and suppose $\|\mathbf{g} \cdot \boldsymbol{w}\|$ and $\|\boldsymbol{c} \cdot \boldsymbol{h}\|$ are each at most $q/2$. Suppose $\langle \mathbf{g} \rangle$ is a prime ideal. Then, either $\mathbf{c}$ or $\boldsymbol{h}$ is in the ideal $\langle \mathbf{g} \rangle$.*

Proof. Since $\mathbf{g} \cdot \boldsymbol{w} = \boldsymbol{c} \cdot \boldsymbol{h} \bmod q$, and since $\|\mathbf{g} \cdot \boldsymbol{w}\|$ and $\|\boldsymbol{c} \cdot \boldsymbol{h}\|$ are each at most $q/2$, we have $\mathbf{g} \cdot \boldsymbol{w} = \boldsymbol{c} \cdot \boldsymbol{h}$ exactly. We also have an equality of ideals $\langle \mathbf{g} \rangle \cdot \langle \boldsymbol{w} \rangle = \langle \boldsymbol{c} \rangle \cdot \langle \boldsymbol{h} \rangle$, and, since $\langle \mathbf{g} \rangle$ is a prime ideal and our cyclotomic ring is a unique factorization domain (see Proposition 5.14), we have that $\langle \mathbf{g} \rangle$ divides either $\langle \boldsymbol{c} \rangle$ or $\langle \boldsymbol{h} \rangle$ (or both). The result follows. □

Lemma 6.5. *Let $q = n^{\omega(1)}$, and $\langle \mathbf{g} \rangle$ be a prime ideal such that $\|\mathbf{g}\| = \mathsf{poly}(n)$. Sample $\boldsymbol{h} \leftarrow D_{\mathbb{Z}^n, \sqrt{q}}$ such that $\boldsymbol{h} \notin \langle \mathbf{g} \rangle$. Then, there is no $\epsilon > 0$ and $\boldsymbol{c} \notin \mathcal{I}$ such that $\|\boldsymbol{c}\| < q^{1/8}$ and $\|[\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q\| < q^{1-\epsilon}$.*

Proof. We are give than $\|c\| < q^{1/8}$ and have $\|h\| < \sqrt{q \cdot n}$ (with overwhelming probability using Lemma 4.5). Hence, using Lemma 5.9 we have that $\|\boldsymbol{c} \cdot \boldsymbol{h}\| < q^{1/8+1/2} \cdot n < q/2$. Also for the sake of contradiction assume that that $\boldsymbol{w} = [\boldsymbol{c} \cdot \boldsymbol{h}/\mathbf{g}]_q$ is such that $\|\boldsymbol{w}\| < q^{1-\epsilon}$. Then again we have that $\|\boldsymbol{w} \cdot \mathbf{g}\| < q^{1-\epsilon} \cdot \|\mathbf{g}\|\sqrt{n} < q/2$ as $\|\mathbf{g}\| = \mathsf{poly}(n)$ and $q = n^{\omega(1)}$. Now using Lemma 6.4, we have that either $\boldsymbol{c}$ or $\boldsymbol{h}$ is in the ideal $\langle \mathbf{g} \rangle$, which is a contradiction. □

Extraction: $s \leftarrow \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u_\kappa)$. To extract a "canonical" and "random" representation of a coset from an encoding $\boldsymbol{u} = [\mathbf{c}/\mathbf{z}^\kappa]_q$, we just multiply by the zero-testing parameter $\mathbf{p}_{zt}$, collect the $(\log q)/4 - \lambda$ most-significant bits of each of the n coefficients of the result, and apply a strong randomness extractor to the collected bits (using the seed from the public parameters). Namely

$$\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{u}) = \textsc{Extract}_s(\mathsf{msbs}([\boldsymbol{u} \cdot \mathbf{p}_{zt}]_q)) \quad \text{(}\mathsf{msbs}\text{ of coefficient representation).}$$

This works because for any two encodings $\boldsymbol{u}, \boldsymbol{u}'$ of the same coset we have

$$\|\mathbf{p}_{zt}\boldsymbol{u} - \mathbf{p}_{zt}\boldsymbol{u}'\|_\infty = \|\mathbf{p}_{zt}(\boldsymbol{u} - \boldsymbol{u}')\|_\infty < q^{3/4},$$

so we expect $\mathbf{p}_{zt}\boldsymbol{u}$, $\mathbf{p}_{zt}\boldsymbol{u}'$ to agree on their $(\log q)/4 - \lambda$ most significant bits. (There is a negligible (in λ) chance that $\boldsymbol{u}$ and $\boldsymbol{u}'$ are such that $\mathbf{p}_{zt}\boldsymbol{u}$ and $\mathbf{p}_{zt}\boldsymbol{u}'$ are on opposite sides of a boundary, such that they have different MSBs.) On the other hand, by Lemma 6.5, we know that we cannot have $\|\mathbf{p}_{zt}(\boldsymbol{u}-\boldsymbol{u}')\| < q^{1-\epsilon}$ when $\boldsymbol{u}-\boldsymbol{u}'$ encodes something nonzero, and therefore (since $\lambda \ll \log q/4$) the values $\mathbf{p}_{zt}\boldsymbol{u}$ and $\mathbf{p}_{zt}\boldsymbol{u}'$ cannot agree on their $(\log q)/4 - \lambda$ MSBs.

This means, however, that no two points in the basic cell of $\mathcal{I}$ agree on their collected bits when multiplied by $\mathbf{p}_{zt}$, so the collected bits from an encoding of a random coset have min-entropy at least $\log |R/\mathcal{I}|$. We can therefore use a strong randomness extractor to extract a nearly uniform bit-string of length (say) $\lfloor \log |R/\mathcal{I}| \rfloor - \lambda$.

6.2 Setting the parameters

In this section we provide the parameters for the basic setting that should be set so that all the constraints required by the scheme are met. A overview is presented in Table 6.2.

Parameter	Constraints	Value Set
σ	By Lemma 6.1, $\|\mathbf{g}\| \le \sigma\sqrt{n}$, $\|\mathbf{g}^{-1}\| \le n^{c+1.5}$.	$\sqrt{n\lambda}$
σ'	By Lemma 6.2, $\sigma' \ge \sqrt{n\lambda} \cdot \|\mathbf{g}\|$.	$\lambda n^{3/2}$
σ^*	Super-polynomially larger than γ the size of the numerator of encoding being randomized. By Theorem 4.8, $\sigma^* > \mathsf{poly}(n, m)$	$2^\lambda \gamma$
q	Multiplication of κ encoding should have small numerator. By Lemma 6.5, $q > n^{\omega(1)}$. By Lemma 6.3, $\|\mathbf{g}^{-1}\| < \frac{q^{1/8}}{n^{3/2}}$.	$q \ge 2^{8\kappa\lambda} n^{O(\kappa)}$
m	Constrained by Theorem 4.8.	$O(n^2)$

Table 6.1: Parameters for our graded encoding scheme.

- The basic Gaussian parameter σ that we use to draw the ideal generator, $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$, needs to be set to satisfy $\sigma \ge \eta_{2^{-\lambda}}(\mathbb{Z}^n)$, which means that we have $\sigma = \sqrt{\lambda n}$. Then as argued in Lemma 6.1 we have that the size of $\mathbf{g}$ is bounded with overwhelming probability by $\|\mathbf{g}\| \le \sigma\sqrt{n} = n\sqrt{\lambda}$.

- Once we have the ideal lattice $\mathcal{I} = \langle \mathbf{g} \rangle$, the Gaussian parameter σ' by Lemma 6.2 we should have $\sigma' \ge \|\mathbf{g}\|\sqrt{\lambda n}$. Given the bound from above bound on the size of $\mathbf{g}$, it is sufficient to set $\sigma' = \lambda n^{3/2}$, which means that the size of level-zero elements is bounded with overwhelming probability by λn^2.

- Recall that σ'' and σ''' are the the size of the numerators of $\mathbf{y}$ and the $\mathbf{x}_i$. Theorem 4.8 requires that σ''' be larger that $\eta_{2^{-\lambda}}(\mathbb{Z}^n)$. In Section 7.4 we show an alternate (more secure) procedure for generation of $\mathbf{y}$ and the $\mathbf{x}_i$'s and the that the size of the numerators in $\mathbf{y}$ and the $\mathbf{x}_i$'s generated by that procedure will be bounded by $\sigma\mathsf{poly}(n)$ with high probability.

- The Gaussian parameter σ^* that we use to draw the coefficient vector $\boldsymbol{r}$ during re-randomization of newly generated level-1 encodings, must be large enough so that: (1) The resulting distribution on $\sum r_i\mathbf{x}_i$ is to close to a wide ellipsoid Gaussian encodings of zero. Thus Theorem 4.8 requires that $\sigma^* > \mathsf{poly}(n, m, \lambda)$. (2) The resulting distribution on $\sum r_i\mathbf{x}_i$ is such that it "drowns" the numerator $\mathbf{a}\boldsymbol{d}$ of the initial encoding $\mathbf{a}\boldsymbol{d}/\mathbf{z}$ and setting $\sigma^* = 2^\lambda$ is suffices for this purpose. For this value of σ^*, a re-randomized level-one encoding is of the form $[\boldsymbol{c}/\mathbf{z}]_q$ with the size of $\boldsymbol{c}$ is bounded by $\|\boldsymbol{c}\| \leq 2^\lambda \cdot \text{poly}(n, m)$.

- A level-κ encoding is obtained by multiplying κ level-one encodings (which will always be re-randomized). Hence it is of the form $[\boldsymbol{c}/\mathbf{z}^\kappa]_q$ with $\boldsymbol{c}$ of size bounded with high probability by $\|\boldsymbol{c}\| \leq (2^\lambda \cdot \text{poly}(n))^\kappa = 2^{\kappa\lambda} \cdot n^{O(\kappa)}$. To use Lemma 6.5 for level-κ encodings, we need $\|\boldsymbol{c}\| \leq q^{1/8}$, so it is sufficient to set $q \geq 2^{8\kappa\lambda} \cdot n^{O(\kappa)}$. With this choice the constraints from Lemma 6.5 ($q > n^{\omega(1)}$) and Lemma 6.3 ($\|\mathbf{g}^{-1}\| < \frac{q^{1/8}}{n^{3/2}}$) are easily satisfied.

- Finally, we need m to be sufficiently large so that we can use Theorem 4.8, which we can do here by setting $m = O(n^2)$.

- Finally, in order to get λ-level security against lattice attacks, we roughly need to set the dimension n large enough so that $q < 2^{n/\lambda}$, which means that $n > \tilde{O}(\kappa\lambda^2)$.

6.3 Extensions and Variants

Some applications of multi-linear maps require various modifications to the basic encoding scheme from above, such as "assymetric maps" that have difference source groups. We briefly describe some of these variants below.

Avoiding prime ideals. Note that in certain application it may not be essential for the ideal $\mathcal{I}$ to be a prime. For example, for the application (as explained in Chapter 10) of one-round N-party key-exchange it suffices to have a principal ideal $\mathcal{I}$ such that its norm has large prime factors.

Another re-randomization approach. Recall that the re-randomization approach as described the in the basic variant of the scheme involved publishing encodings of zero which can then be added to the encoded term to re-randomize it. A different approach is to re-randomize $\mathbf{y}$ first, by setting $\mathbf{y}' := \mathbf{y} + \mathbf{X}\boldsymbol{r}$ and then encode via the re-randomized encoding of 1, namely as $\boldsymbol{u}_1 := [\mathbf{y}'\boldsymbol{d}]_q$. This does not have the information-theoretic same-distribution guarantee as provided by the basic variant of the scheme (since the distributions $[\mathbf{y}'\boldsymbol{d}]_q$ and $[\mathbf{y}'\boldsymbol{d}']_q$ may differ, even if $\boldsymbol{d}, \boldsymbol{d}'$ are both short and in the same coset). But on the plus side, it is more convenient to use this re-randomization method for encoding at high levels $i > 1$: After computing the randomized $\mathbf{y}'$, we can use it by setting $\boldsymbol{u}_i := [\boldsymbol{d}(\mathbf{y}')^i]_q$.

Extending re-randomization. Note that in the basic variant of the scheme we used the matrix $\mathbf{X}$ to randomize level-one encodings. Using similar pubic parameter $\mathbf{X}_i$ now consisting of encoding of zero at the ith level, we can generalize the re-randomization procedure to work at any level $i \leq \kappa$. In particular we abstract this procedure as $\mathsf{reRand}(\mathbf{y}, i, \boldsymbol{u}')$: Given $\boldsymbol{u}' = [\boldsymbol{c}'/\mathbf{z}^i]_q$ with noise-bound $\|\boldsymbol{c}'\| < \gamma$, we draw an m-vector of integer coefficients $\boldsymbol{r} \leftarrow D_{\mathbb{Z}^m,\sigma^*}$ for large enough σ^* (e.g. $\sigma^* = 2^\lambda\gamma$), and output $\boldsymbol{u} := [\boldsymbol{u}' + \mathbf{X}_i\boldsymbol{r}]_q$ as a re-randomized version of $\boldsymbol{u}$. Using the same argument as in the basic variant of the scheme we can conclude that the distribution generated in this way will be independent of $\boldsymbol{c}'$, except in the coset that it belongs to.

Note that for some applications (e.g. [GGH+13c]) it might be useful to use the re-randomization operation multiple times. Here we consider the case in which ℓ re-randomizations (for some constant ℓ) are needed. Furthermore in between these re-randomization steps we might have some (say, some constant) addition and multiplication operations on the intermediate encodings. One way to achieve this would be to use $\sigma^* = 2^{\lambda^j}$ when performing the j^{th} re-randomization (for any j). In other words sample $\boldsymbol{r}$ from $D_{\mathbb{Z}^m,\sigma^*}$ where $\sigma^* = 2^{\lambda^j}$ and use it to re-randomize the encoding that has been obtained after $j-1$ previous re-randomizations. Furthermore observe that the addition and multiplication of encodings increases noise by a small factor which will be wiped clean with re-randomizations. In this setting where at most ℓ re-randomizations are needed we will need $q > 2^{8\kappa\lambda^\ell n^{O(\kappa)}}$. Finally, in order to get λ-level security against lattice attacks, we will need to set the dimension n such that $n > \tilde{O}(\kappa\lambda^{1+\ell})$.

Asymmetric encoding. Now we will describe our construction for general graded encodings (Definition A.3 in Appendix A).

In this variant we still choose just one ideal generator $\mathbf{g}$, but several different denominators $\mathbf{z}_j \overset{r}{\leftarrow} R_q$, $j = 1, \ldots, \tau$. Then, a vector of the form $\boldsymbol{c}/\mathbf{z}_j \in R_q$ with $\boldsymbol{c}$ short is a level-one encoding of the coset $\boldsymbol{c}+\mathcal{I}$ relative to the "j'th dimension". In this case we use vectors rather than integers to represent the different levels, where for an index $\boldsymbol{w} = \langle w_1, \ldots, w_\tau \rangle \in \mathbb{N}^\tau$ and a coset $\boldsymbol{c}' + \mathcal{I}$, the encodings of $\boldsymbol{c}' + \mathcal{I}$ relative to the index $\boldsymbol{w}$ are

$$S_{\boldsymbol{w}}^{(\boldsymbol{c}'+\mathcal{I})} = \left\{ \boldsymbol{c}/\boldsymbol{z}^* \ : \ \boldsymbol{c} \in \boldsymbol{c}' + \mathcal{I},\ \|\mathbf{c}\| < q^{1/8},\ \boldsymbol{z}^* = \prod_{i=1}^{\tau} \mathbf{z}_i^{w_i} \right\}.$$

To enable encoding in this asymmetric variant, we provide the public parameters $\mathbf{y}_j = [\mathbf{a}_j/\mathbf{z}_j]_q$ and $\{\mathbf{x}_{i,j} = [\mathbf{b}_{i,j}/\mathbf{z}_j]_q\}_i$ for all $j = 1, 2, \ldots, \kappa$, with short $\mathbf{a}_i \in 1 + \mathcal{I}$ and $\mathbf{b}_{i,j} \in \mathcal{I}$. To enable zero-test relative to index $\langle v_1, \ldots, v_\tau \rangle \in \mathbb{N}^\tau$ we provide the zero-test parameter $\mathbf{p}_{zt} = (\boldsymbol{h} \cdot \prod_{i=1}^{\tau} \mathbf{z}_i^{v_i})/\mathbf{g} \in R_q$. The parameters for this variant will have to be set in order to provide functionality up to $\sum_i v_i$ levels. In particular, we will need $q > 2^{8\kappa\lambda^{\sum_i v_i} n^{O(\kappa)}}$ and $n > \tilde{O}(\kappa\lambda^{1+\sum_i v_i})$.

Providing zero-test security. In applications that require resilience of the zero test even against invalid encodings, we augment the zero-test parameter by publishing many elements $\mathbf{p}_{\mathrm{zt},i} = [\boldsymbol{h}_i\mathbf{z}^\kappa/\mathbf{g}]_q$ for several different $\boldsymbol{h}_i$'s. As part of our new zero-test we require that a level-κ encoding must pass the zero-test relative to *all* the parameters $\mathbf{p}_{\mathrm{zt},i}$.

Consider a purported encoding $\boldsymbol{u} = \boldsymbol{c}/\mathbf{z}^\kappa$ where in this case we do not assume necessarily that $\|\boldsymbol{c}\| < q^{1/8}$ (as would be true for a valid encoding). Applying multiple zero-testers, we obtain

$$\mathbf{p}_{\text{zt},1}\boldsymbol{u} = \boldsymbol{h}_i\boldsymbol{c}/\mathbf{g}, \quad \ldots, \quad \mathbf{p}_{\text{zt},t}\boldsymbol{u} = \boldsymbol{h}_t\boldsymbol{c}/\mathbf{g} \ .$$

This t-dimensional vector lies in a lattice L generated by the vector $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_t)$ modulo q, Note that since $\|\boldsymbol{h}_i\| \ll q$ for all i, the vector $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_t)$ is quite short modulo q. Moreover, by making t large enough (but still polynomial), we can ensure that *all* of the vectors in L whose lengths are much less than q are *unreduced* (small) multiples of $(\boldsymbol{h}_1, \ldots, \boldsymbol{h}_t)$. Therefore, if the encoding passes the multiple zero-test, $\boldsymbol{c}/\mathbf{g}$ must be small, and therefore $\boldsymbol{u}$ has the form of an encoding of zero.

Avoiding Principal Ideals. In light of the fact that some of the attacks in Chapter 9 rely on the fact that $\mathcal{I}$ is a principal ideal, it makes sense to seek a scheme that can use also "generic" (non-principal) ideals according to a nice canonical distribution. Unfortunately, we do not know how to do this, since we do not know how to generate a general ideal $\mathcal{I}$ according to a nice distribution together with short vectors (e.g., within $\mathsf{poly}(n)$ of the first minima) in each of $\mathcal{I}$ and $\mathcal{I}^{-1}$.

We note that we can at least adapt the zero-test to general ideals, should the other problems be resolved. We can replace the single zero-test parameter $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$ by n parameters, $\mathbf{p}_{\text{zt},i} = [\boldsymbol{h}_i\mathbf{z}^\kappa \cdot \boldsymbol{f}_i]_q$, where the vectors $\boldsymbol{f}_i$ are "in spirit" just a small basis of the fractional ideal $\mathcal{I}^{-1}$ (but they are mapped to R_q via $\frac{1}{\boldsymbol{x}} \in K \mapsto \boldsymbol{x}^{-1} \in R_q$). We note that a similar approach also addresses the (small) possibility that $\|\mathbf{g}^{-1}\|$ is not small. Since $\mathbf{g}^{-1} \subset R$, we can reduce $\mathbf{g}^{-1}$ modulo the integral basis of R to obtain short elements of $\mathcal{I}^{-1}$, and hence zero-testers that are sufficiently small.

CHAPTER 7

Security of Our Constructions

The security of our graded encoding systems relies on new, perhaps unconventional assumptions, and at present it seems unlikely that they can be reduced to more established assumptions, such as learning-with-errors (LWE) [Reg05], or even the NTRU hardness assumption [HPS98]. Given that the construction of multilinear maps has been a central open problem now for over a decade, we feel that exploring unconventional assumptions for this purpose is well worth the effort, as long as this exploration is informed by extensive cryptanalysis.

Simplistic Attacks. We begin our cryptanalysis with a "sanity check," arguing that simplistic attacks that only compute rational functions in the system parameters cannot recover any "interesting quantities", and in particular cannot break our DDH analog. In particular, we consider "simplistic" generic attacks that operate on the encodings of params and the problem instance using only simple operations – add, subtract, multiply, divide. That is, we model [Kal85a, Kal85b] attackers as *arithmetic straight-line programs* (ASLPs). This model is analogous[Sho97b] to the generic group model, which is often used as a "sanity check" in the analysis of group-based cryptosystems. As an example in our case, an ASLP can generate the element $\mathbf{p}_{zt}\mathbf{x}_i^{\kappa}$, which equals $\boldsymbol{h}\mathbf{g}^{\kappa-1}\mathbf{b}_i'^{\kappa}$ where $\mathbf{b}_i' = \mathbf{b}_i/\mathbf{g}$. We want to check that an ASLP cannot generate anything "dangerous."

We prove that an ASLP cannot solve GCDH. We do this by defining a weight function w for rational functions, such that everything in the GCDH instance has weight zero, but a GCDH solution has weight 1. The weight function behaves much like polynomial degree. For example, the term $[\mathbf{a}/\mathbf{z}]_q$ in params has weight 0, since we set $w(\mathbf{a}) = 1 = w(\mathbf{z})$. As another example, $w(\mathbf{p}_{zt}) = w(\boldsymbol{h}) + \kappa \cdot w(\mathbf{z}) - w(\mathbf{g})$, which equals 0, since we set $w(\mathbf{g}) = 1$ and $w(\mathbf{p}_{zt}) = 1 - \kappa$. To vastly oversimplify the remainder of our analysis, we show that, given terms of weight 0 (as in the GCDH instance), an ASLP attacker can only produce more terms of weight 0, and thus not a GCDH solution. (See Lemma 7.5 for a more accurate statement.)

Non-generic attacks. More realistically, we consider (non-generic) averaging, algebraic and lattice attacks. To make this investigation broadly accessible, in this chapter we will start by presenting the different attack scenarios that we need to be worried about. More specifically, we identify seemingly useful quantities that can be computed from the public parameters, and other quantities that if we could compute them then we could break the scheme. We describe averaging and lattice-reduction attacks that can perhaps be useful in recovering some of these "interesting targets," and propose countermeasures to render these attacks less dangerous. While describing the attacks themselves we do not delve into the number theoretic details which are deferred to Chapter 9, where they are studied extensively. Many of these attacks arose in the cryptanalysis of NTRU signature schemes [HKL+00, HPS01, HHGP+03], but a couple of them are new (and will be of broader interest).

Undoubtedly there is a lot of meat here for cryptanalysts. But the bottom line is that we have extended the best known attacks (see Chapter 9) and still not found an attack that is threatening to our constructions.

7.1 Our Hardness Assumption

In our constructions, the attacker sees the public parameters $\mathsf{params} = (\mathbf{y}, \{\mathbf{x}_i\}_{i=1}^m)$, where $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ is a level-1 encoding of $1 + \mathcal{I}$ and each $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$ is a level-1 encoding of $0 + \mathcal{I}$. Recall (from Table 6.2) that $\mathcal{I} = \langle \mathbf{g} \rangle$ where $\|\mathbf{g}\| = \mathrm{poly}(n) = q^{o(1)}$, and a level-$i$ encoding of a coset $\alpha + \mathcal{I}$ is an element of the form $\boldsymbol{u} = [\boldsymbol{c}/\mathbf{z}^i]_q$ where $\boldsymbol{c} \in \alpha + \mathcal{I}$ is short, typically $\|\boldsymbol{c}\| = q^{o(1)}$ (and always $\|\boldsymbol{c}\| < q^{1/8}$). In addition the attacker also sees a zero-testing parameter at level κ of the form $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$ with $\|\boldsymbol{h}\| = q^{1/2+o(1)}$.

Expressing the abstract GDDH assumption from Chapter 3 in terms of our specific construction, we get the following computational assumptions (below we state both the search and the decision versions). Consider the following process, on parameters $\lambda, n, q, \kappa, \sigma = \mathrm{poly}(n), \sigma^* = \sigma \cdot 2^\lambda$ (as described in Chapter 6):

```
$(\mathbf{y}, \{\mathbf{x}_i\}_i, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^n, 1^\kappa)$
For $i = 0, \ldots, \kappa$
    Choose $\boldsymbol{e}_i \leftarrow D_{\mathbb{Z}^n,\sigma}$ and $\boldsymbol{f}_i \leftarrow D_{\mathbb{Z}^n,\sigma}$    // $\boldsymbol{e}_i, \boldsymbol{f}_i$ in random $\eta_i + \mathcal{I}, \phi_i + \mathcal{I}$
    Set $\boldsymbol{u}_i = \left[\boldsymbol{e}_i\mathbf{y} + \sum_j r_{ij}\mathbf{x}_j\right]_q$ where $r_{ij} \leftarrow D_{Z,\sigma^*}$ // encode only the $\eta_i$'s
Set $\boldsymbol{u}^* = \left[\prod_{i=1}^{\kappa} \boldsymbol{u}_i\right]_q$    // level-$\kappa$ encoding
Set $\boldsymbol{v} = [\boldsymbol{e}_0 \cdot \boldsymbol{u}^*]_q$    // encoding of the right product
Set $\boldsymbol{v}' = [\boldsymbol{f}_0 \cdot \boldsymbol{u}^*]_q$    // encoding of a random product
```

Definition 7.1 (GCDH/GDDH)**.** *The (graded) CDH problem (GCDH) is, on input* $((\mathbf{y}, \{\mathbf{x}_i\}_i, \mathbf{p}_{zt}), \boldsymbol{u}_0, \ldots, \boldsymbol{u}_\kappa)$ *to output a level-κ encoding of* $\prod_i \boldsymbol{e}_i + \mathcal{I}$*, specifically* $\boldsymbol{w} \in R_q$ *such that* $\|[\mathbf{p}_{zt}(\boldsymbol{v} -$

$\boldsymbol{w})]_q\| < q^{3/4}$. [1] *The graded DDH problem (GDDH) is to distinguish between* $\boldsymbol{v}$ *and* $\boldsymbol{v}'$, *or more formally between the distributions*

$$\mathcal{D}_{\mathrm{GDDH}} = \{(\mathbf{y}, \{\mathbf{x}_i\}_i, \mathbf{p}_{zt}), \boldsymbol{u}_0, \ldots, \boldsymbol{u}_\kappa, \boldsymbol{v}\} \quad \textit{and} \quad \mathcal{D}_{\mathrm{RAND}} = \{(\mathbf{y}, \{\mathbf{x}_i\}_i, \mathbf{p}_{zt}), \boldsymbol{u}_0, \ldots, \boldsymbol{u}_\kappa, \boldsymbol{v}'\}.$$

7.2 Simplistic Models of Attacks

We begin our cryptanalysis effort by considering "simplistic" generic attacks. Roughly, these are attacks in which we just take the terms the public parameters, add, subtract, multiply, and divide them, and hope to get something useful out of it. In other words, we consider *arithmetic straight-line programs* (ASLP) [Kal85a, Kal85b] over the ring R_q as our model of attack.

We argue that such simplistic attacks are inherently incapable of solving GCDH. To that end we consider the different terms from the public parameters as formal variables, and show that all of the rational functions that the attacker can derive have a special form. Then we argue that any term of this form that expresses a solution to GCDH must refer to a polynomial of large size and cannot serve as a correct solution.

Before presenting this analysis, we remark that a slightly less simplistic attack model is the *black-box field* (BBF) model of Boneh and Lipton [BL96]. In that model, the attacker can still compute terms that are rational functions in the given parameters, but now it can also test whether two terms are equal (and in our case perhaps also see the results of applying the zero test on two terms). Although we do not have any bounds on the security of our scheme in this model, we note that Boneh and Lipton's generic BBF algorithm for solving discrete log does not extend to our setting to solve our "discrete log" problem. The reason is that their algorithm requires black-box exponentiations of high (exponential) degree, whereas our encodings only permit the evaluation of polynomially-bounded degree, after which the "noise" in our encodings overwhelms the signal.

7.2.1 Hardness of GCDH in the Arithmetic Straight-Line Program Model

Our ASLP analysis resembles potential-function analysis to some extent. We assign some *weight* to terms from the public parameters and the GCDH instance that the attacker gets as input (and think of this weight as our "potential"). We then characterize the weight of the terms that the attacker can compute using an ASLP on these input terms, and argue that terms of this weight are not useful for solving GCDH.

Notation. First, we establish some terminology. Recall that a rational function is a ratio of two (multivariate) polynomials, and that the set of rational functions in some variables is closed under addition, subtraction, multiplication and division. We denote the rational functions over the set of variables V over a ring R by $\mathcal{R}_R(V)$.

[1] *This formulation allows the adversary to output even an* invalid encoding, *as long as it passes the equality check.*

Definition 7.2 (Weight of Variables and Rational Functions)**.** *Consider a set of variables $V = \{x_1, \ldots, x_t\}$ over some ring R, and a* weight function *on these variables $w : V \to \mathbb{Z}$. This weight function is inductively extended rational functions in these variables over R, $w^* : \mathcal{R}_R(V) \to \mathbb{Z}$ as follows:*

- *For any constant $c \in R$, $w^*(c) = 0$, and for any variable $x \in V$ $w^*(x) = w(x)$;*
- *$\forall a \in \mathcal{R}_R(V)$, $w^*(-a) = w^*(a)$ and if $a \not\equiv 0$ then $w^*(1/a) = -w^*(a)$;*
- *$\forall a, b \in \mathcal{R}_R(V)$, s.t. $a + b$ is not equivalent to any simpler function, $w^*(a + b) = \max\{w^*(a), w^*(b)\}$.*
- *$\forall a, b \in \mathcal{R}_R(V)$, s.t. ab is not equivalent to any simpler function, $w^*(ab) = w^*(a) + w^*(b)$.*

It can be shown that this extension w^* is well defined over the ring of integers in any number field. One example of such a weight function is the degree of the polynomial in the variables in V, when $w(x)$ is set to 1 for each $x \in V$. Below we identify w^* with w and denote both by $w(\cdot)$.

Definition 7.3 (Homogeneous Weight-Balanced Rational Function for weight function $w(\cdot)$)**.** *We say that a rational function $r(x_1, \ldots, x_t) = p(x_1, \ldots, x_t)/q(x_1, \ldots, x_t)$ is* homogeneous *for weight function $w(\cdot)$ if p and q are such that each one of their monomials has the same weight. We say that r is* homogeneous weight-balanced *for weight function $w(\cdot)$ if it is homogeneous and has weight zero.*

We use the following easy fact:

Fact 7.4. *Let $r_1(x_1, \ldots, x_t)$ and $r_2(x_1, \ldots, x_t)$ be homogeneous balanced rational functions for weight function $w(\cdot)$. Then $-r_1$, $1/r_1$, $r_1 + r_2$ and $r_1 \cdot r_2$ are all homogeneous balanced rational functions for weight function $w(\cdot)$.*

Intuition. Using the above definitions, our basic strategy will be to treat the relevant elements in our scheme as formal variables and assign a *weight* and a *size* to them. Weights will be assigned such that all the terms that the adversary sees are homogenous weight-balanced rational functions. Fact 7.4 then implies that the terms that an ASLP attacker can produce must also be homogenous weight-balanced rational function. On the other hand the assigned size value lower-bounds the expected size of that element in the actual scheme. The main lemma in our analysis asserts that any element obtained as weight-balanced rational function, which is equivalent to $\prod_{i=0}^{\kappa} \mathbf{e}_i/\mathbf{z}^{\kappa} \pmod{\mathcal{I}}$, must have numerator of size more than $\sqrt{q}$. This means that when multiplied by the zero-testing parameter we get reduction modulo q, hence such term will not pass the equality test.

Size of terms. Below we use the following rules for the evolution of the size: If a, b are an elements of size $\mathsf{sz}(a), \mathsf{sz}(b)$, respectively, then we have $\mathsf{sz}(-a) = \mathsf{sz}(a)$, $\mathsf{sz}(1/a) = q$, $\mathsf{sz}(a + b) = \mathsf{sz}(a) + \mathsf{sz}(b)$ and $\mathsf{sz}(ab) = \mathsf{sz}(a) \cdot \mathsf{sz}(b)$. (The convention of $\mathsf{sz}(1/a) = q$ captures the intuition that the inverse of a small R_q element has size roughly q.)

Weight and size of elements in our scheme. Recall that a GCDH attacker gets as input the terms $\mathbf{a}/\mathbf{z}, \{\mathbf{b}_i/\mathbf{z}\}_{i=1}^m, \boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}$, and $\{\boldsymbol{e}_j/\mathbf{z}\}_{j=0}^\kappa$ (all in R_q), where we have $\mathcal{I} = \langle \mathbf{g} \rangle$, $\mathbf{b}_i \in \mathcal{I}$ for all i and $\mathbf{a} \in 1 + \mathcal{I}$.

To ensure that all the terms that the attacker gets are homogenous weight-balanced rational functions, we set $w(\mathbf{z}) = w(\mathbf{g}) = w(\mathbf{a}) = 1$ and also $w(\mathbf{b}_i) = 1$ for all i and $w(\boldsymbol{e}_j) = 1$ for all j. Finally, to make the zero-test parameter weight-balanced we set $w(\boldsymbol{h}) = 1 - \kappa$. We note that $\boldsymbol{h}$ is the only element that has negative weight. (If we wish to consider the decomposition $\mathbf{b}_i = \boldsymbol{r}_i\mathbf{g}$, then $w(\boldsymbol{r}_i) = 0$, and similarly if we decompose $\mathbf{a} = \boldsymbol{r}\mathbf{g} + 1$ then $w(\boldsymbol{r}) = 0$.)

For our analysis below, it is sufficient to assign size c for some constant $c > 0$ to all the "small" elements, size just over $\sqrt{q}$ to the mid-size element $\boldsymbol{h}$, and size q to the random element $\mathbf{z}$. Namely we have $\mathsf{sz}(\mathbf{z}) = q$, $\mathsf{sz}(\mathbf{g}) = \mathsf{sz}(\mathbf{a}) = c$, $\mathsf{sz}(\mathbf{b}_i) = c$ for all i, $\mathsf{sz}(\boldsymbol{e}_j) = c$ for all j and $\mathsf{sz}(\boldsymbol{h}) = \sqrt{q}$.

Lemma 7.5. *Consider the GCDH instance* $\Gamma = (\mathbf{a}/\mathbf{z}, \{\mathbf{b}_i/\mathbf{z}\}_{i=1}^m, \boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}, \{\boldsymbol{e}_j/\mathbf{z}\}_{j=0}^\kappa)$ *with weights and sizes as above. Assume that q is a prime. Let $\mathcal{A}$ be an arithmetic straight-line program. If* $\mathcal{A}(\Gamma) = \boldsymbol{c}/\mathbf{z}^k$ *such that* $[\boldsymbol{c}]_q \equiv \prod_{j=0}^\kappa \boldsymbol{e}_j \pmod{\mathcal{I}}$ *then* $\mathsf{sz}([\boldsymbol{c}]_q) > \sqrt{q}$.

Proof. By Fact 7.4 and the weights of elements in Γ, $\mathcal{A}$ can produce only homogenous weight-balanced rational functions of the variables. Since $w(\mathbf{z}) = 1$, this implies $w(\mathbf{c})$ is κ. Going forward, the intuition is since $\prod_{j=0}^\kappa \boldsymbol{e}_j$ has weight $\kappa + 1$, the only way to get $\mathbf{c}$ to have the correct weight is to make it divisible by $\boldsymbol{h}$, since it is the only variable with negative weight. But this makes the size of $\mathbf{c}$ at least $\sqrt{q}$.

Formally we prove below that any homogeneous balanced rational function $\boldsymbol{d}$ that satisfies $\boldsymbol{d} \equiv \boldsymbol{c} \pmod{q}$ and $\boldsymbol{d} \equiv \prod_{j=0}^\kappa \boldsymbol{e}_j \pmod{\mathcal{I}}$ much have size at least $\sqrt{q}$, so in particular this must hold for $[\boldsymbol{c}]_q$.

Since $\boldsymbol{c}$ and $\boldsymbol{d}$ are homogeneous and $\boldsymbol{d} \equiv \boldsymbol{c} \pmod{q}$, there exist two homogeneous rational functions $\boldsymbol{s}, \boldsymbol{s'}$ such that $\boldsymbol{c} = \boldsymbol{s}\boldsymbol{d} + \boldsymbol{s'}$ with $\boldsymbol{s} \equiv 1 \pmod{q}$ and $\boldsymbol{s'} \equiv 0 \pmod{q}$. Since $\boldsymbol{c}$ is homogeneous therefore we have

$$w(\boldsymbol{c}) = w(\boldsymbol{s}) + w(\boldsymbol{d}) = w(\boldsymbol{s'}).$$

Similarly since $\boldsymbol{d} \equiv \prod_{j=0}^\kappa \boldsymbol{e}_j \pmod{\mathcal{I}}$ then we must have $\boldsymbol{d} = \boldsymbol{r} \prod_{j=0}^\kappa \boldsymbol{e}_j + \boldsymbol{r'}$ for homogeneous rational functions $\boldsymbol{r}, \boldsymbol{r'}$ that satisfy $\boldsymbol{r} \equiv 1 \pmod{\mathcal{I}}$ and $\boldsymbol{r'} \equiv 0 \pmod{\mathcal{I}}$, and again we have

$$w(\boldsymbol{d}) = w(\boldsymbol{r}) + \kappa + 1.$$

Putting the two weight equations together, we thus have $w(\boldsymbol{c}) = w(\boldsymbol{s}) + w(\boldsymbol{r}) + \kappa + 1$. At the same time, by Fact 7.4 we know that $\mathcal{A}$ can only produce weight-balanced rational terms, so $w(\boldsymbol{c}/\mathbf{z}^\kappa) = 0$. Therefore $w(\boldsymbol{c}) = w(\mathbf{z}^\kappa) = \kappa$, which implies that $w(\boldsymbol{s}) + w(\boldsymbol{r}) = -1$. This implies that either $w(\boldsymbol{s}) < 0$ or $w(\boldsymbol{r}) < 1$.

Considering the size of $\boldsymbol{d}$, we first note that if $\boldsymbol{d} = \boldsymbol{p}/\boldsymbol{p'}$ for a nontrivial denominator $\boldsymbol{p'}$ then $\mathsf{sz}(\boldsymbol{d}) \geq q$ and there is nothing more to prove. Below we therefore assume that the denominator $\boldsymbol{p}'$ is trivial, i.e. $\boldsymbol{d}$ is a simple polynomial. Since $\boldsymbol{d} = \boldsymbol{r}\prod_{j=0}^{\kappa}\boldsymbol{e}_j + \boldsymbol{r}'$, then also $\boldsymbol{r}'$ is a simple polynomial and the only terms that we can have in the denominator of $\boldsymbol{r}$ are the $\boldsymbol{e}_j$'s. But we know that $\boldsymbol{r} \equiv 1 \pmod{\mathcal{I}}$ so the same $\boldsymbol{e}_j$'s must be in its numerator, making $\boldsymbol{r}$ too a simple polynomial. We conclude that $\boldsymbol{r}, \boldsymbol{r}'$ must both be simple polynomials, and $\mathsf{sz}(\boldsymbol{d}) = \mathsf{sz}(\boldsymbol{r}) \cdot \mathsf{sz}(\prod_j \boldsymbol{e}_j) + \mathsf{sz}(\boldsymbol{r}')$.

Returning to the weight, we now have two cases to analyze: either $w(\boldsymbol{s}) < 0$ or $w(\boldsymbol{r}) \leq 0$.

- If $w(\boldsymbol{r}) \leq 0$, then since the only variable with non-positive weight in our scheme is $\boldsymbol{h}$, it must be that $\boldsymbol{h}$ divides $\boldsymbol{r}$. Hence we get $\mathsf{sz}(\boldsymbol{c}) \geq \mathsf{sz}(\boldsymbol{d}) \geq \mathsf{sz}(\boldsymbol{r}) \geq \mathsf{sz}(\boldsymbol{h}) \geq \sqrt{q}$.
- Considering the other case $w(\boldsymbol{s}) < 0$, we note $\boldsymbol{s} \equiv 1 \pmod{q}$ but none of the terms in our system are equivalent to 1 modulo q. The only way to get a homogeneous rational function $\boldsymbol{s} \equiv 1 \pmod{q}$ is if $w(\mathbf{s})$ is divisible by $q-1$. Since the weight of $\boldsymbol{s}$ is negative and divisible by $q-1$, then in particular we have $w(\boldsymbol{s}) \leq -q+1$. Therefore, $w(\boldsymbol{r}) \geq q-2$. For Γ, weights, and sizes as defined above, clearly $\mathsf{sz}(\boldsymbol{r})$, and hence $\mathsf{sz}(\boldsymbol{d})$, exceeds $\sqrt{q}$.

□

7.3 Cryptanalysis Beyond the Generic Models

Below we attempt "real cryptanalysis" of our scheme, using state of the art tools in algebraic cryptanalysis and lattice reduction. Throughout this section we consider in particular the GDDH assumption, hence we assume that the attacker is given the following inputs, all relative to the random element $\mathbf{z} \in R_q$ and the ideal $\mathcal{I} = \langle \mathbf{g} \rangle \subset R$, with $\|\mathbf{g}\| \approx \sigma\sqrt{n}$.

- $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$, a level-one encoding of 1, namely $\mathbf{a} \in 1 + \mathcal{I}$ and $\|\mathbf{a}\| \geq \sigma\sqrt{n}$.
- $\mathbf{x}_i = [\mathbf{b}_i/\mathbf{z}]_q$, m randomizing terms s.t. $\forall i,\ \mathbf{b}_i \in \mathcal{I}$ and $\|\mathbf{b}_i\| \geq \sigma\sqrt{n}$. Below it will be convenient to denote $\mathbf{b}_i = \mathbf{b}'_i \cdot \mathbf{g}$, where $\mathbf{b}'_i$ is of size similar to $\mathbf{b}_i$.
- $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^k/\mathbf{g}]_q$ the zero-test parameter with $\|\boldsymbol{h}\| \approx \sqrt{qn}$;
- $\boldsymbol{u}_j = [\boldsymbol{e}_j/\mathbf{z}]_q$, $\kappa+1$ level-one encodings of random elements where $\forall j,\ \|\boldsymbol{e}_j\| \approx 2^{\lambda}\sigma\sqrt{n}$;
- $\boldsymbol{w} = [\boldsymbol{c}/\mathbf{z}^k]_q$, the "challenge element" with allegedly $\|\boldsymbol{c}\| < q^{1/8}$ and $\boldsymbol{c} \equiv \prod_{j=0}^{\kappa}\boldsymbol{e}_j \pmod{\mathcal{I}}$.

Our parameter setting is $n = \tilde{O}(\kappa\lambda^2)$ and $q \approx 2^{n/\lambda}$. In the analysis below we consider as a "real break" any method that has a heuristically significant chance of distinguishing the challenge $\boldsymbol{w}$ from a level-κ encoding of a random element different from $\prod_j \boldsymbol{e}_j$.

7.3.1 Easily computable quantities

Using only algebraic transformations (with no need for lattice reduction), it is easy to compute from the given parameters also the following quantities:

- Taking different κ-products including some number $r \geq 1$ of the $\mathbf{x}_i$'s, some number $s \geq 0$ of the $\boldsymbol{u}_j$'s and some power of $\mathbf{y}$, and multiplying these products by the zero-test parameter $\mathbf{p}_{zt}$, we get many different elements of the form

$$\begin{aligned}\boldsymbol{v} &= \left[\left(\prod_{k=1}^{r} \mathbf{x}_{i_k}\right) \cdot \left(\prod_{k=1}^{s} \boldsymbol{u}_{j_k}\right) \cdot \mathbf{y}^{\kappa-r-s} \cdot \mathbf{p}_{zt}\right]_q \\ &= \left(\prod_{k=1}^{r} \mathbf{b}'_{i_k}\right) \cdot \mathbf{g}^{r-1} \cdot \left(\prod_{k=1}^{s} \boldsymbol{e}_{j_k}\right) \cdot \mathbf{a}^{\kappa-r-s} \cdot \boldsymbol{h} \qquad (7.1)\end{aligned}$$

  Importantly, the right-hand-side in Equation (7.1) is *not reduced modulo* q, because it is a product of the mid-size $\boldsymbol{h}$ by exactly κ short elements, hence its size is smaller than q.

- All the $\boldsymbol{v}$'s of the form of Equation (7.1) have a common factor $\boldsymbol{h}$, but if we choose the other elements at random then with high probability they will have no other common factors. Hence after seeing enough of them we can expect to get a basis for the principal ideal lattice $\langle \boldsymbol{h} \rangle$.

  A similar argument implies that we can also compute bases for the principal ideals $\langle \boldsymbol{h} \cdot \boldsymbol{e}_j \rangle$ for every $j \in \{0, 1, \ldots, \kappa\}$ and also bases for $\langle \boldsymbol{h} \cdot \mathbf{g} \rangle$ and $\langle \boldsymbol{h} \cdot \mathbf{a} \rangle$.

- Given a basis for $\langle \boldsymbol{h} \rangle$, we can get a basis for the fractional principal ideal $\langle 1/\boldsymbol{h} \rangle$ (where $1/\boldsymbol{h}$ is the inverse of $\boldsymbol{h}$ in the number field K).

- Using the bases for $\langle \boldsymbol{h} \cdot \mathbf{g} \rangle$ and $\langle 1/\boldsymbol{h} \rangle$, we can compute a basis for our principal ideal $\mathcal{I} = \langle \mathbf{g} \rangle$. Similarly we can also compute a basis for $\langle \mathbf{a} \rangle$ and bases for all the principal ideals $\langle \boldsymbol{e}_j \rangle$.

The above tells us that we cannot expect to hide the ideal $\mathcal{I}$ itself, or the ideals generated by any of the other important elements in our scheme. It may still be hard, however, to find the short generators for these ideals, or any short elements in them. Indeed this difficulty is the sole reason for the conjectured security of our schemes.

7.3.2 Using averaging attacks

Averaging attacks are described in Sections 9.1 through 9.4, roughly speaking they allow us, after seeing many elements of the form $\boldsymbol{r}_i \cdot \boldsymbol{a}$ for the same $\boldsymbol{a}$ but many different "random" $\boldsymbol{r}_i$'s (e.g., that are independent of $\boldsymbol{a}$), to get a good approximation of $\boldsymbol{a}$ (or some related quantities from which we can derive $\boldsymbol{a}$).

In our case, if we use simplistic Gaussian distributions to choose all our public parameters, then we expect to be able to apply these tools with elements from Equation (7.1), in order to get approximations for $\boldsymbol{h}$ or $\boldsymbol{h} \cdot \mathbf{g}^r$ for various r's. The tools from the literature do not quite work "right out of the box" because the terms that we want to recover are not very short. Specifically they have size more than $\sqrt{q}$, so techniques from the literature may need to average super-polynomial (or even exponential) number of samples to get useful approximations.

In Section 9.5, however, we describe a new method that can recover elements such as $\boldsymbol{h}$ or $\boldsymbol{h} \cdot \mathbf{g}^r$ from approximations that are not very accurate. The level of accuracy needed to apply Theorem 9.11 still requires super-polynomial number of samples, but only just: It is heuristically enough to use only $n^{O(\log\log n)}$ samples. Indeed this potential attack is the reason for the slightly involved method of choosing the randomizers in Section 6.1, which is based on the countermeasures discussed in Section 7.4 below.

Another potential problem in using these attacks is that our public parameters only include a small number of terms, whereas averaging attacks typically need a much larger number of samples. However, the attacker can get many more samples by taking sums and products of terms from the public parameters, and it seems likely that such samples will be "independent enough" to serve in the averaging attacks.

Below we show how recovering (small multiples of) the terms $\mathbf{g}$ or $1/\boldsymbol{h}$, can be used to break our scheme, and also a plausible method of using a small multiple of $\boldsymbol{h} \cdot \mathbf{g}^r$ for a large value of r. We remark that for the cases of having a small multiple of $\mathbf{g}$ or $1/\boldsymbol{h}$ we can show a real working attack, but for the case of having a small multiple of $\boldsymbol{h} \cdot \mathbf{g}^r$ we only have a "somewhat plausible approach" that does not seem to lead to a real attack.

7.3.3 Cryptanalysis with extra help

A short element in $\langle \mathbf{g} \rangle$. We begin by showing that knowing any short element in the ideal $\mathcal{I} = \langle \mathbf{g} \rangle$ would enable the attacker to break our scheme. Any short element in $\mathcal{I}$ has the form $\boldsymbol{d} \cdot \boldsymbol{g}$ for a short $\boldsymbol{d}$ (because $\mathbf{g}^{-1} \in K$ is short). We begin the attack by multiplying in R_q the short $\boldsymbol{d} \cdot \boldsymbol{g}$ by the zero-test parameter $\mathbf{p}_{zt}$, thus getting the modified zero-test parameter $\mathbf{p}'_{zt} = [\boldsymbol{d} \cdot \boldsymbol{h} \cdot \mathbf{z}^\kappa]_q$. Then we multiply the modified zero-test parameter by both the "challenge element" $\boldsymbol{w}$ and by the product of κ of the random encodings $\boldsymbol{u}_j$.

In the case where $\boldsymbol{w}$ is indeed an encoding of the right product, we would have $\boldsymbol{w} = (\boldsymbol{c}\mathbf{g} + \prod_{j=0}^{\kappa} \boldsymbol{e}_i)/\mathbf{z}^\kappa$ for some not-too-big $\boldsymbol{c}$ (i.e., $\|\boldsymbol{c}\| < q^{1/8}$). Hence in this case we would get the two elements

$$\boldsymbol{v}_1 \;:=\; [\mathbf{p}'_{zt} \cdot \boldsymbol{w}]_q \;=\; \boldsymbol{d} \cdot \boldsymbol{h} \cdot \left(\boldsymbol{c} \cdot \mathbf{g} + \prod_{j=0}^{\kappa} \boldsymbol{e}_j \right) \quad \text{and} \quad \boldsymbol{v}_2 \;:=\; \left[\mathbf{p}'_{zt} \cdot \prod_{j=1}^{\kappa} \boldsymbol{u}_j \right]_q \;=\; \boldsymbol{d} \cdot \boldsymbol{h} \cdot \prod_{j=1}^{\kappa} \boldsymbol{e}_j.$$

Our next goal is to "divide $\boldsymbol{v}_1$ by $\boldsymbol{v}_2$ modulo $\mathcal{I}$" in order to isolate the element $\boldsymbol{e}_0$. For that purpose, we use our knowledge of a basis of $\mathcal{I}$ and compute the Hermite normal form (HNF) of that lattice. Recall that the HNF basis has the form of a upper-triangular matrix, and with good probability the first entry on the main diagonal is the norm of $\mathcal{I}$ (denoted $\mathsf{N}(\mathcal{I})$) and all the other entries are 1. Below we assume that this is indeed the case.

We can reduce both $\boldsymbol{v}_1$ and $\boldsymbol{v}_2$ modulo the HNF basis of $\mathcal{I}$, and if the basis has the above special form then we get two integers $\nu_1 = [\boldsymbol{v}_1]_{\mathrm{HNF}(\mathcal{I})} \in \mathbb{Z}$ and $\nu_1 = [\boldsymbol{v}_1]_{\mathrm{HNF}(\mathcal{I})} \in \mathbb{Z}$. Clearly we have

$$\nu_1 \equiv \boldsymbol{v}_1 \equiv \boldsymbol{dh} \prod_{j=0}^{\kappa} \boldsymbol{e}_j \pmod{\mathcal{I}}, \quad \text{and} \quad \nu_2 \equiv \boldsymbol{v}_2 \equiv \boldsymbol{dh} \prod_{j=1}^{\kappa} \boldsymbol{e}_j \pmod{\mathcal{I}}$$

Assuming that ν_2 is co-prime to $\mathsf{N}(\mathcal{I})$, we can now compute over the integers $\eta = \nu_1 \cdot \nu_2^{-1} \bmod \mathsf{N}(\mathcal{I})$. Observing that we always have $\mathsf{N}(\mathcal{I}) \in \mathcal{I}$, we therefore get (for some $\tau \in \mathbb{Z}$)

$$\eta \cdot \nu_2 = \nu_1 + \tau \cdot \mathsf{N}(\mathcal{I}) \equiv \nu_1 \pmod{\mathcal{I}}.$$

At the same time we also have

$$\boldsymbol{e}_0 \cdot \nu_2 \equiv \boldsymbol{e}_0 \cdot \boldsymbol{v}_2 \equiv \boldsymbol{v}_1 \equiv \nu_1 \pmod{\mathcal{I}}.$$

Since ν_2 is co-prime with $\mathsf{N}(\mathcal{I})$ then it is also co-prime with the ideal generator $\mathbf{g}$, and hence the two equalities above imply that $\eta \equiv \boldsymbol{e}_0 \pmod{\mathcal{I}}$.

Finally, we can reduce η modulo the rotation basis of $\boldsymbol{d} \cdot \mathbf{g}$, which is a basis consisting of only short vectors (because $\boldsymbol{d} \cdot \mathbf{g}$ itself is short). This yields a short element $\boldsymbol{e}_0' = \eta + \boldsymbol{t} \cdot \boldsymbol{d}\mathbf{g} \equiv \eta \equiv \boldsymbol{e}_0 \pmod{\mathcal{I}}$. We observe that the short $\boldsymbol{e}_0'$ is functionally equivalent to the coset $\boldsymbol{e}_0$ which was encoded in $\boldsymbol{u}_0$. (At least, it is functionally equivalent when $\boldsymbol{d} \cdot \mathbf{g}$ is short enough; if it is not short enough, the attack may fail.)

In particular we can use it to verify that the challenge element is indeed an encoding of the right product: we just multiply $\boldsymbol{u}_0' = \boldsymbol{e}_0' \cdot \mathbf{y}$ to get a level-one encoding, then check that $\boldsymbol{u}_0 - \boldsymbol{u}_0'$ is a level-one encoding of zero. (Or course this test will fail in the random case, since the element that we recover will be in the coset of $\boldsymbol{f}_0$ not in the coset of $\boldsymbol{e}_0$.)

A small multiple of $1/\boldsymbol{h}$. Recall that we can compute from the public parameters a basis for the fractional ideal $\langle 1/\boldsymbol{h} \rangle$. If we could find a "somewhat short" element in that lattice, namely an element $\boldsymbol{v} = \boldsymbol{d}/\boldsymbol{h}$ with $\|\boldsymbol{d}\| \ll \sqrt{q}$, then we can mount the following simple attack:

Multiplying the zero-test parameter by $\boldsymbol{v}$, we get the "higher-quality" zero-test parameter $\mathbf{p}_{zt}' = [\mathbf{p}_{zt} \cdot \boldsymbol{v}]_q = [\boldsymbol{d}\mathbf{z}^\kappa/\mathbf{g}]$. Once we have this higher-quality parameter, we can square it and multiply by one of the randomizers to get

$$\mathbf{p}_{zt}'' = [(\mathbf{p}_{zt}')^2 \mathbf{x}_0]_q = [\boldsymbol{d}^2 \mathbf{z}^{2\kappa}/\mathbf{g}^2 \cdot \mathbf{b}_0' \mathbf{g}]_q = [\boldsymbol{d}^2 \mathbf{b}_0' \mathbf{z}^{2\kappa}/\mathbf{g}]_q.$$

If $\|\boldsymbol{d}\|$ is sufficiently short so that $\|\boldsymbol{d}^2 \mathbf{b}_0'\| \ll q$, then we can use $\mathbf{p}_{zt}''$ as a zero-test parameter at level 2κ. In particular we can distinguish whether the challenge element is an encoding of the right product or a random product by computing the level-$(\kappa+1)$ encoding of the product $\prod_{j=0}^{\kappa} \boldsymbol{u}_j$, mapping $\boldsymbol{w}$ to level $\kappa + 1$ by multiplying with $\mathbf{y}$, then use the level-2κ zero-test parameter $\mathbf{p}_{zt}''$ to check if these two elements are indeed in the same coset.

A small multiple of $\boldsymbol{h}\mathbf{g}^r$. If we could compute an element $\boldsymbol{h}\mathbf{g}^r$ (for a large value of r) or a not-too-big multiple of it, say $\boldsymbol{v} = \boldsymbol{dh}\mathbf{g}^r$ such that $\|\boldsymbol{v}\| \ll q$, then the following line of attack becomes "somewhat plausible," though it does not seem to lead to a real attack.

Extracting the r'th root of $\boldsymbol{v}$ we get $\boldsymbol{v}' = \sqrt[r]{\boldsymbol{dh}} \cdot \mathbf{g}$. We note that when $\boldsymbol{dh}$ is "random and independent of $\mathbf{g}^r$", then $\sqrt[r]{\boldsymbol{dh}}$ (over the number-field K) tends to a (known) constant as r increases. [2] We can therefore hope that for a large enough value of r the fractional element $\sqrt[r]{\boldsymbol{v}}$ will provide a good enough approximation of $\mathbf{g}$, and then we could perhaps use an algorithm such as the one from Section 9.5 to recover $\mathbf{g}$ exactly.

It seems, however, that this line of attack as described does not work in our case. The reason is that we cannot hope to get approximations of $\boldsymbol{h}\mathbf{g}^r$ for $r \geq \kappa - 1$, and our dimension n is always much larger than κ, so this method inherently cannot produce good enough approximations. Still perhaps it can be used in conjunction with other tools.

7.4 Some Countermeasures

As explained above, the most potent attacks that we found against our scheme make use of averaging attacks, using samples that we get by multiplying the zero-test parameter by products of κ other elements from the public parameters. We note that for the purpose of defending against averaging attacks we can ignore the GDDH instance, since it can be generated by the attacker itself just from the public parameters. (At least as long as the averaging part does not use the challenge element $\boldsymbol{w}$.)

Fortunately, Gentry, Peikert and Vaikuntanathan (GPV) [GPV08] have already given us an approach to defeat this sort of averaging attacks. One of the key conceptual insights of [GPV08] is that using *any* good basis $\boldsymbol{B}$ of a lattice Λ (e.g., a lattice where $\|\boldsymbol{B}\|$ is less than some bound β) can generate samples from the lattice according to a *canonical* Gaussian distribution (with deviation tightly related to β). Thus, the sampled lattice points do not reveal anything about the sampler's *particular* basis $\boldsymbol{B}$ aside from an upper bound on $\|\boldsymbol{B}\|$. We will use a similar approach, where we derive all the elements in the public parameters from a small set of elements, using a GPV-type procedure.

Specifically, we give out (potentially many) encodings of 0 $\{\mathbf{x}'_i = \mathbf{b}'_i \cdot \mathbf{g}/\mathbf{z}\}$. Let us ignore, for the moment, the fact that these encodings live in R_q, and instead pretend that we present them to the attacker as elements $\mathbf{b}'_i\mathbf{g}/\mathbf{z}$ in the overlying cyclotomic field. (Of course, we are giving the attacker an additional advantage here.) Then, all of the encodings are in the fractional principal ideal lattice $\mathcal{J} = \langle \mathbf{g}/\mathbf{z} \rangle$. If we simply chose the $\mathbf{b}'_i$ values randomly and independently, it is conceivable that an averaging/transcript attack could recover $\mathbf{g}/\mathbf{z}$. However, we instead follow [GPV08] by generating the encodings $\{\mathbf{b}_i\}$ according to a Gaussian distribution over the fractional ideal lattice, using an efficient discrete Gaussian sampler [GPV08, Pei10, DN12a]. By the same argument as [GPV08], such encodings (presented in characteristic zero) reveal nothing in particular about the term $\mathbf{g}/\mathbf{z}$ that is being used to

[2] An easy example: If $\mathcal{U} \in_R [0, B]$ then $\Pr[\mathcal{U} > \frac{9}{10}B] = 0.1$. However if $\mathcal{U} \in_R [0, B^{100}]$ then $\Pr[\sqrt[100]{\mathcal{U}} > \frac{9}{10}B] \approx 1$.

generate the encodings. More formally we have:

As argued in Lemma 6.1 note that when choosing $\mathbf{g} \leftarrow D_{\mathbb{Z}^n,\sigma}$ we get $\|\mathbf{g}^{-1}\| < n^{c+1.5}$ (in K) with a noticeable probability and we re-choose $\mathbf{g}$ until this condition is met. Similarly, one can show that with probability noticeable probability over the choice of $\mathbf{z}$ we have $\|\mathbf{z}^{-1}\| < n^2/q$ (in K), so in our instance generation we re-choose $\mathbf{z}$ until this condition is met. When this condition is met, then we have $\|\mathbf{g}/\mathbf{z}\| < \sigma n^3/q$ (using Lemmas 5.9 and 6.1). Additionally since we have $\|\tilde{\boldsymbol{B}}\| \geq \|\boldsymbol{B}\|$, therefore we can use the GPV procedure (Theorem 4.7) to sample elements from $\mathcal{J}$ according to the Gaussian distribution $\mathbf{x}'_i \leftarrow D_{\mathcal{J},s}$ with parameter $s = \sigma n^{3.5}/q$ (say).

We note that the elements that we draw are of the form $\mathbf{x}'_i = \mathbf{b}'_i \cdot \mathbf{g}/\mathbf{z}$ for some (integral) $\mathbf{b}'_i \in R$. Moreover we can bound the size of the $\mathbf{b}'_i$'s by $\|\mathbf{b}'_i\| \leq n\|\mathbf{x}'_i\| \cdot \|\mathbf{z}\| \cdot \|1/\mathbf{g}\| < n(\sigma n^4/q) \cdot q\sqrt{n} \cdot n^{c+1.5} = n^{c+7}\sigma$.

Next we map these elements to R_q by setting $\mathbf{x}_i = [\mathbf{b}'_i\mathbf{g}/\mathbf{z}]_q$. Denoting the numerator by $\mathbf{b}_i = \mathbf{b}'_i\mathbf{g}$, we can bound its size by $\|\mathbf{b}_i\| = \sqrt{n}\|\mathbf{b}'_i\| \cdot \|\mathbf{g}\| < n^{c+7.5}\sigma \cdot \sigma\sqrt{n} = \sigma^2 n^{c+8}$. Sampled this way, we know that the randomizers $\mathbf{x}_i$ do not provide any more power to the attacker beyond the ability to sample elements from $\mathcal{J}$ according to $D_{\mathcal{J},s}$. [3] Finally, we note that the public parameter $\boldsymbol{y}$ corresponding to an encoding of 1 can also be sampled in a similar manner.

We set $\boldsymbol{h}$ in a similar way. Again, we use [GPV08] to prevent the attacker analyzing the zero-tester $\boldsymbol{h} \cdot \boldsymbol{z}^\kappa/\boldsymbol{g}$ geometrically to extract useful information about $\boldsymbol{h}$, or the other terms, individually. Roughly, once $\boldsymbol{g}$ and $\mathbf{z}$ are chosen, one chooses $\boldsymbol{h}$ according to an ellipsoid Gaussian of the same "shape" as $\boldsymbol{g}/\boldsymbol{z}^\kappa$, so that the distribution of the zero-tester is a spherical Gaussian.

An alternative heuristic countermeasure. Although we prefer to use the GPV-type approach above, we note for completeness that another plausible line of defense against averaging attacks is to actually decrease the number of elements made public, perhaps as few as only two. Namely we can publish only two elements $\mathbf{x}_1 = [\mathbf{b}'_1\mathbf{g}/\mathbf{z}]_q$ and $\mathbf{x}_2 = [\mathbf{b}'_2\mathbf{g}/\mathbf{z}]_q$, perhaps chosen according to the procedure above conditioned on $\mathbf{b}'_1, \mathbf{b}'_2$ being co-prime. To re-randomize a level-one encoding $\boldsymbol{u}$, we can then choose two small elements $\boldsymbol{a}_1, \boldsymbol{a}_2$ and set $\boldsymbol{u}' = \boldsymbol{u} + \boldsymbol{a}_1 \cdot \mathbf{x}_1 + \boldsymbol{a}_2 \cdot \mathbf{x}_2$. One drawback of this method is that we can no longer use Theorem 4.8 to argue that the output distribution of reRand is nearly independent of its input, instead we need to use yet another computational assumption (and a rather awkward one at that). Another drawback is that it is not at all clear that the attacker cannot just take many terms of the form $\boldsymbol{a}_1 \cdot \mathbf{x}_1 + \boldsymbol{a}_2 \cdot \mathbf{x}_2$ (for many random pairs $(\boldsymbol{a}_1, \boldsymbol{a}_2)$) to use for the samples of the averaging attacks.

[3] We expect it be even slightly less powerful, since these samples are mapped into R_q before the attacker sees them.

7.5 Easiness of other problems

In light of the apparent hardness of our CDH/DDH analog, we could optimistically hope to get also the analog of other hardness assumptions in bilinear maps, such as decision-linear, subgroup membership, etc. Unfortunately, these problems turn out to be easy in our setting, at least with the simple encoding methods.

To see why, observe that publishing level-1 encodings of 0 and 1 enables some "weak discrete log" computation at any level strictly smaller than κ. Specifically, consider one particular encoding of zero $\mathbf{x}_j = [\mathbf{b}_j/\mathbf{z}]_q$ (where $\mathbf{b}_j = \boldsymbol{c}_j\mathbf{g}$ for some $\boldsymbol{c}_j$), which is given in the public parameters together with an encoding of one $\mathbf{y} = [\mathbf{a}/\mathbf{z}]_q$ and the zero-testing parameter $\mathbf{p}_{zt} = [\boldsymbol{h}\mathbf{z}^\kappa/\mathbf{g}]_q$. Given a level-$i$ encoding with $1 \le i \lneqq \kappa$, $\boldsymbol{u} = [\boldsymbol{d}/\mathbf{z}^i]_q$, we can multiply it by $\mathbf{x}_j$, $\mathbf{p}_{zt}$, and some power of $\mathbf{y}$ to get

$$\begin{aligned}\boldsymbol{f} &= [\boldsymbol{u}\cdot\mathbf{x}_j\cdot\mathbf{p}_{zt}\cdot\mathbf{y}^{\kappa-i-1}]_q = \left[\frac{\boldsymbol{d}}{\mathbf{z}^i}\cdot\frac{\boldsymbol{c}_j\cdot\mathbf{g}}{\mathbf{z}}\cdot\frac{\boldsymbol{h}\mathbf{z}^\kappa}{\mathbf{g}}\cdot\frac{\mathbf{a}^{\kappa-i-1}}{\mathbf{z}^{\kappa-i-1}}\right]_q \\ &= \underbrace{\boldsymbol{d}\cdot\boldsymbol{c}_j\cdot\boldsymbol{h}\cdot\mathbf{a}^{\kappa-i-1}}_{\ll q} = \boldsymbol{d}\cdot\underbrace{\boldsymbol{c}_j\cdot\boldsymbol{h}}_{\Delta_j} \pmod{\mathcal{I}}.\end{aligned}$$

We stress that the right-hand-side of the equality above is *not reduced modulo* q. This means that from a level-i encoding $\boldsymbol{u}$ of an element $\boldsymbol{d}+\mathcal{I}$, we can get a "plaintext version" of $\boldsymbol{d}\cdot\Delta_j$ from some fixed Δ_j (that depends only on the public parameters but not on $\boldsymbol{u}$). This "plaintext version" is not small enough to be a valid level-zero encoding (because Δ_j is roughly the size of $\boldsymbol{h}$, so in particular $\Delta_j > \sqrt{q}$). Nonetheless, we can still use it in attacks.

For starters, we can apply the above procedure to many of the level-one encodings of zero from the public parameters, thereby getting many elements in the ideal $\mathcal{I}$ itself. This by itself still does not yield a basis of $\mathcal{I}$ (since all these elements have the extra factor of $\boldsymbol{h}$), but as shown in Section 7.3.1 we can remove this extra factor and nonetheless compute a basis for $\mathcal{I}$. This is not a small basis of course, but it tells us that we cannot hope to hide the plaintext space $R/\mathcal{I}$ itself.

Next, consider the subgroup membership setting, where we have $\mathbf{g} = \mathbf{g}_1\cdot\mathbf{g}_2$, we are given a level-1 encoding $\boldsymbol{u} = [\boldsymbol{d}/\mathbf{z}]_q$ and need to decide if $\boldsymbol{d} \in \langle\mathbf{g}_1\rangle$. Using the procedure above we can get $\boldsymbol{f} = \boldsymbol{d}\cdot\Delta_j$, which belongs to the ideal $\langle\mathbf{g}_1\rangle$ if $\boldsymbol{d}$ does. Taking the GCD of the ideals $\langle\boldsymbol{f}\rangle$ and $\mathcal{I}$ will then give us the factor $\langle\mathbf{g}_1\rangle$ with high probability. It follows that the subgroup membership problem is easy for the encoding method above.

Finally, consider getting a matrix of elements $\boldsymbol{A} = (\boldsymbol{a}_{i,j})_{i,j}$, all encoded at some level $i \lneqq \kappa$. Using the method above we can get a "plaintext version" of $\Delta_j\cdot\boldsymbol{M}$, which has the same rank as $\boldsymbol{A}$. Since the decision linear problem is essentially a matrix rank problem, this means that this problem too is easy for this encoding method.

At this point it is worth stressing again that these attacks do not seem to apply to the GDDH problem, specifically because in that problem we need to make a decision about a level-κ encoding, and the "weak discrete log" procedure from above only applies to encoding at levels strictly below κ.

Alternatives. The attacks above make it clear that providing encodings of zero in the public parameters (in conjunction with the zero-testing parameter) gives significant power to the adversary. One interesting method to counter these attacks is to use a different randomization tool that can be applied even when we do not have these encodings of zero in the public parameters. For more details on this, we refer the reader to the subsequent work on functional encryption [GGH+13b] where such tools have been developed.

CHAPTER 8

Preliminaries III: Computation in a Number Field

In this chapted we will recall notions that will be useful in understanding the cryptanalysis survey presented in the next chapter.

The *group of units* $\mathcal{U}_K$ associated to a number field K is the group of elements of $\mathcal{O}_K$ that have an inverse in $\mathcal{O}_K$. An element $a \in \mathcal{O}_K$ is a unit if and only if $\mathsf{N}(a) = \pm 1$. The unit group may contain *torsion units* (roots of unity) and *nontorsion units*. By the Dirichlet Unit Theorem, the group of nontorsion units is finitely generated and has *rank* (where rank refers to maximal number of multiplicatively independent elements) is exactly equal to $s_1 + s_2 - 1$.

Let $\sigma : K \to \mathbb{R}^{s_1} \times \mathbb{C}^{2s_2}$ be the canonical embedding defined in Section 5.2. Then the *logarithmic embedding* $\lambda : \mathcal{U}_K \to \mathbb{R}^{s_1+s_2}$ is a homomorphism from a multiplicative group to an additive group given by $\lambda(a) = (\ln|\sigma_1(a)|, \ldots, \ln|\sigma_{s_1+s_2}(a)|)$. The kernel of λ consists of the torsion units in K. For every unit $u \in \mathcal{U}_K$, since $\mathsf{N}(u) = \pm 1$, we have $\sum_{i\in[s_1]} \ln|\sigma_i(u)| + 2\sum_{i\in[s_2]} \ln|\sigma_{s_1+i}(u)| = 0$. This implies that units have rank only $s_1 + s_2 - 1$.

Returning to our example of the mth cyclotomic number field $K = \mathbb{Q}(\zeta_m)$ has a maximal *real subfield* $K^+ = \mathbb{Q}(\zeta_m + \zeta_m^{-1})$, and thus all elements in K^+, are real numbers. It has index 2 in K; its degree is $n/2$. The ring of integers [Was82, Proposition 2.16] $\mathcal{O}_{K^+}$ of K^+ is simply $\mathbb{Z}[\zeta_m + \zeta_m^{-1}]$. The embeddings σ_1, σ_{-1} both fix every element in K^+, and the relative norm $\mathsf{N}_{K/K^+}(a)$ of $a \in K$ is $\sigma_1(a) \cdot \sigma_{-1}(a) = a \cdot \overline{a}$.

The group of units $\mathcal{U}_K$ in the cyclotomic number field $K = \mathbb{Q}(\zeta_m)$ has rank $s_2 - 1 = n/2 - 1$. Since the signature of the real subfield K^+ is $(n/2, 0)$, the rank of the real units $\mathcal{U}_{K^+} = \mathcal{U}_K \cap \mathcal{O}_{K^+}$ is also $n/2 - 1$. For m a prime power, $\mathcal{U}_K$ is generated by ζ_m and $\mathcal{U}_{K^+}$. For m a prime power, an explicit set of generators of $\mathcal{U}_K$ is $\{\pm\zeta_m, (1-\zeta_m^k)/(1-\zeta_m) : k \in \mathbb{Z}_m^*\}$. To see that $\epsilon = (1-\zeta_m^k)/(1-\zeta_m)$ is a unit, observe that $\epsilon = 1 + \zeta_m + \ldots + \zeta_m^{k-1} \in \mathcal{O}_K$ and $\mathsf{N}_{K/Q}(\epsilon) = \prod_{\ell\in\mathbb{Z}_m^*}(1-\zeta_m^\ell)/\prod_{\ell\in\mathbb{Z}_m^*}(1-\zeta_m^\ell) = 1$. Ramachandra [Ram67] explicitly described a full-rank set of independent units for the case that m not a prime power.

In the coefficient embedding, where $a \in \mathcal{O}_K$ is viewed as a polynomial $a(x) \in \mathbb{Z}[x]/\Phi_m(x)$, we have an extension of Fermat's Little Theorem: $a(x)^Q = a(x^Q) \bmod Q$ for any prime Q. When $Q = 1 \bmod m$, this becomes $a^Q = a \bmod Q$.

8.1 Some Computational Aspects of Number Fields and Ideal Lattices

An element $v \in K$ can be represented in its canonical embedding conveniently in terms of the integral basis for $\mathcal{O}_K$. Given $v \in K$ represented in its canonical embedding, it is efficient to convert it to its coefficient embedding, or vice versa – via linear transformations corresponding to multipoint interpolation and evaluation. "Efficient" means in time polynomial in n, $\log \Delta_K$, and the bit-length of v. (Here, Δ_K is the *discriminant* of K. For the important case of the m-th cyclotomic field of degree $n = \phi(m)$, we have $\Delta_K \leq n^n$.) Given $v_1, v_2 \in K$, represented in either their canonical or their coefficient embeddings, it is efficient to compute $v_1 + v_2$, $v_1 \cdot v_2$, and v_1/v_2. To handle denominators, the inverse $1/v_2$ can be represented as $v_2'/\mathsf{N}(v_2)$ where $v_2' \in \mathcal{O}_K$.

Like all lattices, an ideal lattice has a canonical basis called its *Hermite Normal Form* (HNF). The HNF basis of a lattices is unique and can be computed efficiently from any other basis of the lattice. The HNF basis has nice efficiency properties – in particular, it can be expressed in at most $O(n \log d)$ bits, where d is the absolute value of the determinant of a basis of the lattice [Mic01]. It also has nice security properties, in the sense that it reveals no information that cannot be derived in polynomial time from any other basis [Mic01]. For ideal lattices in the canonical embedding, the HNF basis is an integer lattice representing a linear transformation of the integral basis of $\mathcal{O}_K$. The determinant of the HNF basis equals the norm of the ideal. Given HNF bases of ideals $\mathcal{I}_1, \mathcal{I}_2$, one can efficiently compute an HNF basis for the ideals $\mathcal{I}_1 + \mathcal{I}_2$, $\mathcal{I}_1 \cdot \mathcal{I}_2$, $\mathcal{I}_1/\mathcal{I}_2$. Various other natural operations on ideals and bases are also efficient. An example: one can efficiently reduce an element $v \in K$ modulo a basis B – that is, find the element $w \in K$ with $v - w \in \mathcal{I}$ and $w \in \mathcal{P}(B)$, where $\mathcal{P}(B)$ is the parallelepiped associated to B.

8.2 Computational Hardness Assumptions over Number Fields

Hard problems involving ideal lattices often have both algebraic and geometric aspects.

Geometrically, we can specialize standard lattice problems – such as the shortest vector problem (SVP), shortest independent vector problem (SIVP), closest vector problem (SVP), the bounded distance decoding problem (BDDP), etc. – to ideal lattices. The celebrated LLL algorithm [LLL82] finds somewhat short vectors in (general) lattices:

Fact 8.1. *Let $\boldsymbol{B} = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_n}\}$ be a basis of a lattice Λ. Given $\boldsymbol{B}$, the LLL algorithm outputs a vector $\boldsymbol{v} \in L$ satisfying $\|\boldsymbol{v}\|_2 \leq 2^{n/2} \cdot \det(\Lambda)^{1/n}$. The algorithm runs in time polynomial in the size of its input.*

Schnorr and others have described other lattice reduction algorithms with a variety of tradeoffs; for example, [Sch87] proves the following:

Fact 8.2. *Let $\boldsymbol{B} = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_n}\}$ be a basis of a lattice Λ. Given B and integer k, Schnorr's algorithm [Sch87] outputs a vector $\boldsymbol{v} \in \Lambda$ satisfying $\|\boldsymbol{v}\|_2 \leq k^{O(n/k)} \cdot \det(\Lambda)^{1/n}$ in time $k^{O(k)}$.*

The asymptotics of lattice reduction algorithms are still similar to [Sch87], and thus attacks on ideal lattices using purely geometric tools are limited.

Algebraically, we can consider problems such as the factorization of ideals, the structure of the class group and unit group, etc. Subexponential classical algorithms are known for factoring ideals, computing the class group and unit group, and computing a generator of a principal ideal (the Principal Ideal Generator Problem (PIGP)). Polynomial-time quantum algorithms are known for the latter three problems when the degree of the field is constant [Hal05, SV05].

Factoring ideals reduces to factoring integers, hence is subexponential-time classically [LLMP90] and polynomial-time quantumly [Sho97a]. In particular, for any monogenic ring $R = \mathbb{Z}[x]/(f(x))$ such as $\mathcal{O}_K$ for a cyclotomic field K, there is an efficient algorithm to find all of the prime ideals in R with norms that are a power of a prime p. The algorithm resorts to the following theorem.

Theorem 8.3 (Kummer-Dedekind, from [Ste08])**.** *Suppose $f(x) = \prod_i g_i(x)^{e_i} \bmod p$ for prime integer p. The prime ideals $\mathfrak{p}_i$ in $\mathbb{Z}[x]/(f(x))$ whose norms are powers of p are precisely $\mathfrak{p}_i = (p, g_i(x))$.*

There are polynomial time algorithms for factoring polynomials in $\mathbb{Z}_p[x]$ – e.g., by Kaltofen and Shoup [KS98]. Therefore, at least for monogenic rings, factoring an ideal with norm N efficiently reduces to factoring the integer N.

Peikert and Rosen [PR07] provided a reduction of an average-case lattice problem to the worst-case hardness of ideal lattice problem, where the lossiness of the reduction was only logarithmic over fields of small root discriminant. Gentry [Gen10] showed that ideal lattice problems are efficiently *self*-reducible (in some sense) in the quantum setting. This worst-case/average-case reduction exploited, among other things, efficient factorization of ideals via Kummer-Dedekind. Lyubashevsky, Peikert and Regev [LPR10] defined a decision problem called "ring learning with errors" (RLWE) and showed that an attacker that can solve RLWE on average can be used to solve ideal lattice problems, such as SIVP, in the worst case. (Earlier, Regev [Reg05] found an analogous worst-case/average-case connection between the learning with errors (LWE) problem and problems over general lattices.) They relied heavily on the algebraic structure of ideal lattice problems – in particular, on underlying ring automorphisms – to construct their search-to-decision reduction.

CHAPTER 9

Survey of Lattice Cryptanalysis

Here we provide a survey of relevant cryptanalysis techniques from the literature, and also provide two new attacks that we developed in the course of this work. More specifically we consider:

- **Averaging Attacks:** Averaging attacks – described in Sections 9.1 through 9.4 – allow us, after seeing many elements of the form $\boldsymbol{r}_i \cdot \boldsymbol{a}$ for the same $\boldsymbol{a}$ but many different "random" $\boldsymbol{r}_i$'s, to get a good approximation of $\boldsymbol{a}$ (or some related quantities from which we can derive $\boldsymbol{a}$). We will describe the attack itself in Sections 9.1 and 9.2 and consider extensions in Sections 9.3 and 9.4. In particular:

  - In Section 9.1 we present a known attack [HKL$^+$00, GS02] that given a set $S = \{\boldsymbol{v} \cdot \mathbf{y}_i\}$, where $\boldsymbol{v}, \mathbf{y}_1, \mathbf{y}_2, \ldots$ are ring elements, uses "averaging" to recover $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$, where $\overline{\boldsymbol{v}} = \boldsymbol{v}(x^{-1})$ is the conjugate of $\boldsymbol{v}$. These attacks have recently been significantly generalized to lattices with symmetry [Len13].
  - Next in Section 9.2 we present the Gentry-Szydlo [GS02] algorithm that recovers $\boldsymbol{v}$ from $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and a basis of the ideal $\langle \boldsymbol{v} \rangle$.
  - In Sections 9.3 and 9.4 we consider extensions of averaging attacks [NR09, DN12b].

  In our case, one might attempt to mount such an averaging attack on the (possibly many) encodings of 0 $\{\mathbf{x}_i = \mathbf{b}'_i\mathbf{g}/\mathbf{z}\}$ that we provide in params. For example, the attacker can derive the values $\{[\mathbf{p}_{zt}\mathbf{x}_i^{\kappa}]_q = \boldsymbol{h}\mathbf{g}^{\kappa-1} \cdot \mathbf{b}_i'^{\kappa}\}$ as described in Section 7.3.1. Conceivably, depending on the particular distributions of the parameters, the attacker could use averaging to remove the $\mathbf{b}'_i$'s and recover $\boldsymbol{h}\mathbf{g}^{\kappa-1}$.

  We have a couple of defenses against this averaging attack. First, for our constructions it seems that $\boldsymbol{h}\mathbf{g}^{\kappa-1}$ (and other terms that could conceivably be obtained through averaging as explained in Section 7.3.1) do not seem to be useful to the attacker (see

Section 7.3.3). Second, as described in Section 7.4, we choose our params according to distributions designed to make averaging attacks useless. More precisely, we adapt an observation of Gentry, Peikert and Vaikuntanathan [GPV08] in the context of lattice-based signatures – namely, that we can use a "good" lattice basis to generate a transcript of lattice points according to a *canonical* distribution that reveals nothing about the *particular* good basis that we are using (aside from the fact that it is "good"). We generate our params according to such canonical distributions.

- **Closest principal ideal generator problem:** In Section 9.5 we provide a polynomial-time algorithm that solves the *closest principal ideal generator* problem in certain cases. Specifically, it can recover a generator of a principal ideal $\mathcal{I} = \langle \mathbf{g} \rangle$ from a basis of $\mathcal{I}$ and an ϵ-approximation of the generator $\mathbf{g}$, for small enough ϵ – namely, $\epsilon \leq n^{-\Omega(\log \log n)}$. This helps make the averaging attacks described above more robust.

  We review Coppersmith-type attacks [Cop96b, Cop96a] and their relation to our setting in Section 9.6.

- **Dimension-Halving Attack:** In Section 9.7 we describe a "dimension-halving attack" on principal ideal lattices, demonstrating that one needs to double the dimension of principal ideal lattices (compared to general ideal lattices) to preserve security.

9.1 Averaging Attacks

In the so-called "averaging attack," the attacker is given a set $S = \{\boldsymbol{v} \cdot \mathbf{y}_i\}$, where $\boldsymbol{v}, \mathbf{y}_1, \mathbf{y}_2, \ldots$ are ring elements, and its goal is to use "averaging" to recover $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$, where $\overline{\boldsymbol{v}} = \boldsymbol{v}(x^{-1})$ is the conjugate of $\boldsymbol{v}$. It was used by Kaliski (in connection with patent [HKL$^+$00]) and Gentry and Szydlo [GS02] in attacks against NTRU signature schemes [HKL$^+$00, HPS01]. We review the averaging attack here. Along the way, we update the attack so that it works within the ring of integers of any cyclotomic field. (Previously, the attack focused on the ring $\mathbb{Z}[x]/(x^m - 1)$, as used by NTRU signature schemes.)

Now we will describe how the averaging attack works. The distributions of $\boldsymbol{v}$ and the $\mathbf{y}_i$'s may vary, but let us suppose for concreteness that the challenger samples $\boldsymbol{v}'$ and $\{\mathbf{y}_i'\}$ according to Gaussian distributions $\boldsymbol{v}' \leftarrow D_{\mathbb{Z}^m,\sigma}$ and $\mathbf{y}_i' \leftarrow D_{\mathbb{Z}^m,\sigma'}$, interprets these as coefficient vectors of polynomials in $\mathbb{Z}[x]/(x^m - 1)$, and finally sets $\boldsymbol{v} \leftarrow \boldsymbol{v}' \bmod \Phi_m(x)$ and $\mathbf{y}_i \leftarrow \mathbf{y}_i' \bmod \Phi_m(x)$.

Now, consider the average:

$$\mathbf{A}_r \quad = \quad (1/r) \sum_{i=1}^{r} (\boldsymbol{v} \cdot \mathbf{y}_i) \cdot \overline{(\boldsymbol{v} \cdot \mathbf{y}_i)} \quad = \quad (\boldsymbol{v} \cdot \overline{\boldsymbol{v}}) \cdot \left((1/r) \sum_{i=1}^{r} \mathbf{y}_i \cdot \overline{\mathbf{y}_i} \right).$$

Under the canonical embedding, we have:

$$\sigma(\mathbf{A}_r) = \sigma(\boldsymbol{v} \cdot \overline{\boldsymbol{v}}) \cdot \sigma(\mathbf{Y}_r), \quad \text{where} \quad \mathbf{Y}_r = \left((1/r) \sum_{i=1}^{r} \mathbf{y}_i \cdot \overline{\mathbf{y}_i} \right).$$

Toward understanding $\sigma(\mathbf{Y}_r)$, first consider a single vector $\sigma(\mathbf{y}_i \cdot \overline{\mathbf{y}_i})$ in the summation. Recall that, since we are working in a cyclotomic field, the embeddings are all complex and come in conjugate pairs (σ_j, σ_{-j}), where σ_j for $j \in \mathbb{Z}_m^*$ denotes the embedding $\sigma_j(\zeta_m) = \zeta_m^j$. Moreover, for any $\boldsymbol{a}$ in the cyclotomic field, the values $\sigma_j(\boldsymbol{a})$ and $\sigma_{-j}(\boldsymbol{a})$ are conjugate complex numbers, and therefore $\sigma_j(\boldsymbol{a}) \cdot \sigma_{-j}(\boldsymbol{a})$ is a non-negative real number. Now, notice that $\sigma_j(\boldsymbol{a}) \cdot \sigma_{-j}(\boldsymbol{a}) = \sigma_j(\boldsymbol{a}) \cdot \sigma_j(\overline{\boldsymbol{a}}) = \sigma_j(\boldsymbol{a} \cdot \overline{\boldsymbol{a}})$. This means that each vector $\sigma(\mathbf{y}_i \cdot \overline{\mathbf{y}_i})$ in the summation consists entirely of non-negative real numbers!

It is clear that, for any j, the average $\sigma_j(\mathbf{Y}_r) = 1/r \sum_{i=1}^{r} \sigma_j(\mathbf{y}_i \cdot \overline{\mathbf{y}_i})$ converges toward some positive number (rather than tending toward 0). Moreover, by symmetry, it converges to the *same* positive number for all j. Therefore, $\mathbf{A}_r$ converges to $s \cdot \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ for some known positive real scalar s.

The imprecision of the average decreases with $1/\sqrt{r}$. If the coefficients of $\boldsymbol{v}$ are only polynomial in size, then the averaging attack needs only a polynomial number of samples to obtain all of the coefficients of $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ to within less than $1/2$, whereupon the attacker can round to obtain $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ exactly.

As we describe in Section 9.5, in fact even if the coefficients of $\boldsymbol{v}$ are large, an ϵ-approximation of $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$, together with a basis of the ideal $\langle \boldsymbol{v} \cdot \overline{\boldsymbol{v}} \rangle$, is sufficient to recover $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ exactly when ϵ is some inverse-quasi-polynomial function of m. (Note that it is easy to generate a basis of the ideal $\langle \boldsymbol{v} \cdot \overline{\boldsymbol{v}} \rangle$ from a basis of the ideal $\langle \boldsymbol{v} \rangle$, and that the latter (as mentioned previously) can likely be generated from S.)

If the averaging attack is successful and we recover $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$, then we can then use an algorithm by Gentry and Szydlo [GS02] that takes $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and a basis of the ideal $\langle \boldsymbol{v} \rangle$, and outputs the actual element $\boldsymbol{v}$ in polynomial time. This attack is described in the next section.

9.2 Gentry-Szydlo: Recovering $\boldsymbol{v}$ from $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and $\langle \boldsymbol{v} \rangle$

In this section, we describe an algorithm by Gentry and Szydlo [GS02] (the GS algorithm) that recovers $\boldsymbol{v}$ from $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and a basis of the ideal $\langle \boldsymbol{v} \rangle$. The algorithm runs in polynomial time. Gentry and Szydlo used this algorithm in combination with the averaging attack above to break an NTRU signature scheme. They used a set of samples $S = \{\boldsymbol{v} \cdot \mathbf{y}_i\}$ to approximate $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ with sufficient precision to compute it exactly via rounding, and then invoked (but did not implement) the GS algorithm to recover $\boldsymbol{v}$ (the secret signing key). In our setting, the idea would be to attack our params using a similar approach. The GS algorithm was originally designed to work in $\mathbb{Z}[x]/(x^p - 1)$ for prime p. Here, we adapt it to a more general setting over the ring of integers $\mathcal{O}_K$ of the m-th cyclotomic field K. For convenience, we use R to refer to $\mathcal{O}_K$, and R_P to denote $\mathbb{Z}_P[x]/\Phi_m(x)$.

We start by pointing some intuition. Recall that the value $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ is the relative norm of $\boldsymbol{v} \in K = \mathbb{Q}(\zeta_m)$ with respect to the subfield $K^+ = \mathbb{Q}(\zeta_m + \zeta_m^{-1})$ – i.e., $\boldsymbol{v} \cdot \overline{\boldsymbol{v}} = \mathsf{N}_{K/K^+}(\boldsymbol{v})$. The GS algorithm might be somewhat surprising, since we do not know how to recover $\boldsymbol{v}$ efficiently from the norm $\mathsf{N}_{K/\mathbb{Q}}(\boldsymbol{v})$ and a basis of $\langle \boldsymbol{v} \rangle$. Indeed, the value $\mathsf{N}_{K/\mathbb{Q}}(\boldsymbol{v})$ is superfluous, since it can be derived from the basis of $\langle \boldsymbol{v} \rangle$; therefore, finding $\boldsymbol{v}$ would solve the so-called Principal

Ideal Generator Problem, which seems infeasible.

One might also be surprised that $\mathsf{N}_{K/K^+}(\boldsymbol{v})$ and $\langle \boldsymbol{v} \rangle$ are enough to uniquely define $\boldsymbol{v}$, given that $\mathsf{N}_{K/\mathbb{Q}}(\boldsymbol{v})$ and $\langle \boldsymbol{v} \rangle$ only define $\boldsymbol{v}$ up to an infinite group of units. (See Chapter 8 for a discussion on units in cyclotomic number field.) Indeed, $\mathsf{N}_{K/K^+}(\boldsymbol{v})$ and $\langle \boldsymbol{v} \rangle$ are *not* enough to uniquely define $\boldsymbol{v}$ – in particular, if $\boldsymbol{v}' = \boldsymbol{v} \cdot \mathbf{u}$ for any *torsion unit* (root of unity) $\mathbf{u}$, we have $\mathsf{N}_{K/K^+}(\boldsymbol{v}') = \mathsf{N}_{K/K^+}(\boldsymbol{v})$ and $\langle \boldsymbol{v}' \rangle = \langle \boldsymbol{v} \rangle$. However, in attacks, it is typically sufficient to obtain $\boldsymbol{v}$ up to a small set of roots of unity. On the other hand, if $\mathbf{u}$ is not a torsion unit – e.g., if it is a nontrivial cyclotomic unit – then we will have $\mathsf{N}_{K/K^+}(\mathbf{u}) \neq 1$ and therefore $\mathsf{N}_{K/K^+}(\boldsymbol{v}') \neq \mathsf{N}_{K/K^+}(\boldsymbol{v})$. The reason we have $\mathsf{N}_{K/K^+}(\mathbf{u}) \neq 1$ for nontorsion units is that, up to multiplication by a torsion unit, all nontorsion units in K are already in the real subfield K^+ – i.e., $\mathbf{u} = \zeta_m^i \cdot \mathbf{u}'$ where $\mathbf{u}' \in K^+$ is a nontorsion unit. So, $\mathsf{N}_{K/K^+}(\mathbf{u}) = \mathbf{u} \cdot \overline{\mathbf{u}} = \mathbf{u}'^2 \neq 1$.

The essential strategy of the GS algorithm is to combine algebra (in particular, Fermat's Little Theorem) with lattice reduction (LLL). By an extension of Fermat's Little Theorem, for any prime $P = 1 \bmod m$, we have that $\boldsymbol{v}^P = \boldsymbol{v}$ over R_P. Unless $\boldsymbol{v}$ is a zero divisor in R_P (there are only $\mathsf{poly}(m, \log \mathsf{N}_{K/Q}(\boldsymbol{v}))$ primes P for which this can happen), we have $\boldsymbol{v}^{P-1} = 1$ over R_P. Now, suppose that we compute a LLL-reduced basis B of the ideal $\left\langle \boldsymbol{v}^{P-1} \right\rangle$; this we can do in time polynomial in m, P, and the bit-length of $\boldsymbol{v}$. The shortest element $\boldsymbol{w}$ in the reduced basis has the form $\boldsymbol{v}^{P-1} \cdot \boldsymbol{a}$ for some $\boldsymbol{a}$. If it happens that $\|\boldsymbol{a}\|_\infty < P/2$ – i.e., if $\boldsymbol{a}$'s coefficients all have magnitude less than $P/2$ – then we obtain $\boldsymbol{a} = [\boldsymbol{w}]_P$ exactly, and thus $\boldsymbol{v}^{P-1}$. From $\boldsymbol{v}^{P-1}$, we can compute $\boldsymbol{v}$ in time polynomial in m, P, and the bit-length of $\boldsymbol{v}$.

The actual algorithm is more complicated than this, since the essential strategy above leaves two important issues unresolved.

- Issue 1 (How to Guarantee that $\boldsymbol{a}$ is small): LLL guarantees that it will find $\boldsymbol{w} \in \left\langle \boldsymbol{v}^{P-1} \right\rangle$ of length at most $2^{(n-1)/2} \cdot \lambda_1(\left\langle \boldsymbol{v}^{P-1} \right\rangle)$. But this does not imply that $\boldsymbol{a} = \boldsymbol{w}/\boldsymbol{v}^{P-1}$ has length at most $2^{(n-1)/2}$. Indeed, $\left\langle \boldsymbol{v}^{P-1} \right\rangle$ does not even define $\boldsymbol{v}$ uniquely (due to the group of units). Since these units can have arbitrarily high Euclidean norm, $\boldsymbol{a}$ could be arbitrarily long.

- Issue 2 (LLL needs P to be exponential): Let us suppose that we could somehow use LLL to ensure that $\|\boldsymbol{a}\|_\infty \leq 2^{(n-1)/2}$. Then, we need P to be at least $2^{(n+1)/2}$ for the strategy to work. But then $\boldsymbol{v}^{P-1}$ is so long that it takes exponential time even to write it down.

The algorithm resolves these two issues with the following two tools:

- Tool 1 (Implicit Lattice Reduction): We apply LLL *implicitly* to the *multiplicands* of $\boldsymbol{v}^{P-1}$ to ensure that $\boldsymbol{a} = \boldsymbol{w}/\boldsymbol{v}^{P-1}$ has length at most $2^{(n-1)/2}$. The idea is that the relative norm $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ actually reveals a lot about the "geometry" of $\boldsymbol{v}$ (and hence of $\boldsymbol{v}^{P-1}$). We use the relative norm to "cancel" $\boldsymbol{v}^{P-1}$'s geometry so that LLL implicitly acts on the multiplicands.

- Tool 2 (Polynomial Chains): We use $P > 2^{(n+1)/2}$. However, we never compute on $\boldsymbol{v}^{P-1}$ directly. Instead, $\boldsymbol{v}^{P-1}$ and $\boldsymbol{w}$ are represented implicitly via a chain of polynomials that are computed using LLL. From this chain, we compute $\boldsymbol{a} = [\boldsymbol{w}]_P$ exactly. Next, we perform computations modulo a set of small primes $p_1, \ldots, p_t$ – specifically, we reduce $\boldsymbol{a}$ modulo the p_i's, and use the polynomial chain to compute $\boldsymbol{v}^{P-1}$ modulo the p_i's. We do the same thing for another large prime P' such that $\gcd(P-1, P'-1) = 2m$, and then use the Euclidean algorithm (in the exponent) to compute $\boldsymbol{v}^{2m}$ modulo the p_i's. We chose the p_i's so that $2\|\boldsymbol{v}^{2m}\|_\infty < \prod p_i$, so we obtain $\boldsymbol{v}^{2m}$ exactly, from which we can compute $\boldsymbol{v}$ efficiently.

Below, we discuss the GS algorithm in detail.

Implicit Lattice Reduction. We begin with implicit lattice reduction, as characterized by the following lemma.

Lemma 9.1 ([GS02])**.** *Let $\boldsymbol{v} \in R$. Given $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and the HNF basis B for the ideal lattice $\langle \boldsymbol{v} \rangle$, we can output an element $\boldsymbol{w} \in \langle \boldsymbol{v} \rangle$ such that $\boldsymbol{w} = \boldsymbol{v} \cdot \boldsymbol{a}$ and $\|\boldsymbol{a}\|_2^{can} \leq 2^{(n-1)/2} \cdot \sqrt{n}$ in time polynomial in m and the bit-length of $\boldsymbol{v}$.*

Proof. Consider how LLL works. LLL maintains a sequence of n basis vectors $(\boldsymbol{w_1}, \ldots, \boldsymbol{w_n})$. In general, when LLL is deciding whether to perform an operation – a size-reduction step or a swap step – the only information that LLL requires are all of the mutual dot products $\langle \boldsymbol{w_i}, \boldsymbol{w_j} \rangle_{i,j \in [n]}$. In short, LLL needs only the Gram matrix corresponding to its reduced-so-far lattice basis.

Now, consider LLL in our setting, as applied to ideal lattices under the canonical embedding (without trying to do LLL implicitly yet). At a given stage, LLL has a sequence of vectors $(\sigma(\boldsymbol{w}_1), \ldots, \sigma(\boldsymbol{w}_n))$ where the $\boldsymbol{w}_i$'s are in $\langle \boldsymbol{v} \rangle$. LLL (as before) considers only the mutual (Hermitian) inner products of the vectors in deciding whether to perform a step. These inner products are of the form $\langle \sigma(\boldsymbol{w}_i), \sigma(\boldsymbol{w}_j) \rangle = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(\boldsymbol{w}_i \overline{\boldsymbol{w}_j})$.

Now, to do LLL *implicitly* in the canonical embedding – i.e., to use LLL to reduce the multiplicands $\boldsymbol{a}_i = \boldsymbol{w}_i / \boldsymbol{v}$ – LLL needs the mutual Hermitian inner products for $i, j \in [n]$:

$$\langle \sigma(\boldsymbol{w}_i/\boldsymbol{v}), \sigma(\boldsymbol{w}_j/\boldsymbol{v}) \rangle = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(\boldsymbol{w}_i/\boldsymbol{v}) \overline{\sigma_k(\boldsymbol{w}_j/\boldsymbol{v})} = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(1/\boldsymbol{v}\overline{\boldsymbol{v}}) \sigma_k(\boldsymbol{w}_i \overline{\boldsymbol{w}_j}).$$

But all of the values $\sigma_k(1/\boldsymbol{v}\overline{\boldsymbol{v}})$ can be computed efficiently from $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ (and the implicit LLL algorithm actually possesses all of the vectors $\{\sigma(\boldsymbol{w}_i)\}$). Therefore, LLL has all of the information it needs to decide whether to perform a step. To actually perform a step implicitly – size-reduction or swapping – it simply applies the linear transformation dictated by the step to the vectors $\{\sigma(\boldsymbol{w}_i)\}$ that it has in its hand.

The bound $\|\boldsymbol{a}\|^{can} \leq 2^{(n-1)/2} \cdot \sqrt{n}$ follows from the guarantee of LLL and the fact $\|1\|^{can} = \sqrt{n}$ in the canonical embedding. □

Polynomial Chains. Next we talk about the second tool that we use, polynomial chains.

Lemma 9.2 (Theorem 1 in [GS02])**.** *Let $\boldsymbol{v}_0 \in R$. Let $k = \sum k_i 2^i$ with $k_i \in \{0,1\}$ be an integer with $r = \lfloor \log_2 k \rfloor$. Let P be a prime such that $\boldsymbol{v}_0$ is not a zero divisor in R_P. Then, given the input $\boldsymbol{v}_0 \cdot \overline{\boldsymbol{v}_0}$ and a basis B_0 of $\langle \boldsymbol{v}_0 \rangle$, we may compute, in time polynomial in r, m, and the bit-length of the input, the chains:*

$$\{\boldsymbol{v}_0^{k_{r-1}} \cdot \boldsymbol{v}_0^2 \cdot \overline{\boldsymbol{v}_1}, \ldots, \boldsymbol{v}_0^{k_0} \cdot \boldsymbol{v}_{r-1}^2 \cdot \overline{\boldsymbol{v}_r}\} \quad \text{and}$$
$$\{\boldsymbol{v}_0 \cdot \overline{\boldsymbol{v}_0}, \ldots, \boldsymbol{v}_{r-1} \cdot \overline{\boldsymbol{v}_{r-1}}\},$$

where for all $i > 0$, no $\boldsymbol{v}_i$ is a zero divisor in R_P, and $\|\boldsymbol{v}_i\|_2^{can} < 2^{(n-1)/2}\sqrt{n}$. Using these chains, we may compute $\boldsymbol{v}_0^k \cdot \overline{\boldsymbol{v}_r} \bmod P$ in polynomial time. If $k = P - 1 \geq 2^{(n+1)/2}\sqrt{n}\gamma_2$ with $P = 1 \bmod 2m$, we may compute $\overline{\boldsymbol{v}_r}$ exactly, and thereafter use the above chains to compute $\boldsymbol{v}_0^{P-1} \bmod Q$ in polynomial time for any prime Q such that $\overline{\boldsymbol{v}_r}$ is not a zero divisor in R_Q. (Here, γ_2 denotes the maximal value of $\frac{\|\boldsymbol{a}\|_\infty}{\|\boldsymbol{a}\|_2^{can}}$ for any $\boldsymbol{a}$ in the number field.)

Proof. (Sketch) Consider the first term of the first chain: $\boldsymbol{v}_0^{k_{r-1}} \cdot \boldsymbol{v}_0^2 \cdot \overline{\boldsymbol{v}_1}$. For convenience, let $c = k_{r-1} + 2$. Given $\boldsymbol{v}_0 \cdot \overline{\boldsymbol{v}_0}$ and a basis B_0 for $\langle \boldsymbol{v}_0 \rangle$, we efficiently compute $\boldsymbol{v}_0^c \cdot \overline{\boldsymbol{v}_0}^c$ and a basis B_0' for the ideal $\langle \boldsymbol{v}_0^c \rangle$. Then, using implicit lattice reduction (Lemma 9.1), we efficiently compute $\boldsymbol{w} = \boldsymbol{v}_0^c \cdot \boldsymbol{a}$ with $\|\boldsymbol{a}\|_2^{can} < 2^{(n-1)/2}\sqrt{n}$. We set $\boldsymbol{w}$ to be the first term of our chain and set $\boldsymbol{v}_1 \leftarrow \overline{\boldsymbol{a}}$. (Gentry and Szydlo provide techniques to handle the small possibility that $\boldsymbol{v}_1$ is a zero divisor in R_P.)

Now, we compute $\boldsymbol{v}_1 \cdot \overline{\boldsymbol{v}_1}$ as $\boldsymbol{w} \cdot \overline{\boldsymbol{w}} / (\boldsymbol{v}_0^c \cdot \overline{\boldsymbol{v}_0}^c)$. Also, we compute a basis B_1 of $\langle \boldsymbol{v}_1 \rangle$, as follows. Since B_0' generates $\langle \boldsymbol{v}_0^c \rangle$, the terms of the basis B_0' of $\langle \boldsymbol{v}_0^c \rangle$ have the form $\boldsymbol{b}_i = \boldsymbol{v}_0^c \cdot \boldsymbol{a}_i$, where $R = \langle \{\boldsymbol{a}_i\} \rangle$. Our basis B_1 of $\langle \boldsymbol{v}_1 \rangle$ consists of the terms $\boldsymbol{b}_i \cdot \overline{\boldsymbol{w}} / (\boldsymbol{v}_0^c \cdot \overline{\boldsymbol{v}_0}^c) = \boldsymbol{v}_1 \cdot \boldsymbol{a}_i$, which generates $\langle \boldsymbol{v}_1 \rangle$ since (again) $R = \langle \{\boldsymbol{a}_i\} \rangle$.

Now that we have $\boldsymbol{v}_1 \cdot \overline{\boldsymbol{v}_1}$ and a basis B_1 of $\langle \boldsymbol{v}_1 \rangle$, we continue the same process iteratively to compute all of the terms in the chains.

We compute $\boldsymbol{v}_0^k \cdot \overline{\boldsymbol{v}_r} \bmod P$ iteratively, as follows. For $s \leq r$, let $k^{(s)} \in [0, 2^{s+1} - 1]$ denote the $s+1$ MSBs of k. Suppose, inductively, that we have computed $\boldsymbol{v}_0^{k^{(s)}} \cdot \overline{\boldsymbol{v}_s} \bmod P$. (For $s = 1$, this term already exists in the polynomial chain.) Then, we compute

$$\boldsymbol{v}_0^{k^{(s+1)}} \cdot \overline{\boldsymbol{v}_{s+1}} = (\boldsymbol{v}_0^{k^{(s)}} \cdot \overline{\boldsymbol{v}_s})^2 \cdot (\boldsymbol{v}_0^{k_{r-s-1}} \cdot \boldsymbol{v}_s^2 \cdot \overline{\boldsymbol{v}_{s+1}}) / (\boldsymbol{v}_s \cdot \overline{\boldsymbol{v}_s})^2 \bmod P$$

where the latter two multiplicands on the right-hand-side come from the polynomial chains. (Notice that this iterative computation is rather similar to the repeated squaring approach to modular exponentiation.)

We compute $\overline{\boldsymbol{v}_r}$ exactly as $\boldsymbol{v}_0^{P-1} \cdot \overline{\boldsymbol{v}_r} \bmod P$. (This works since the coefficients of $\overline{\boldsymbol{v}_r}$ have magnitude at most $\|\boldsymbol{v}_i\|_2^{can} \cdot \gamma_2 \leq 2^{(n-1)/2}\sqrt{n}\gamma_2 < P/2$.) Thereafter, we clearly can compute $\boldsymbol{v}_0^{P-1}$ modulo any prime Q for which $\overline{\boldsymbol{v}_r}$ is not a zero divisor in R_Q. □

Remainders of the GS Algorithm. In the following lemma we show how to put things together.

Lemma 9.3 (Theorem 2 in [GS02])**.** *Let $\boldsymbol{v} \in R$. Then, given $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and a basis B of $\langle \boldsymbol{v} \rangle$, we may compute $\boldsymbol{v}^{2m}$ in time polynomial in m and the bit length of $\boldsymbol{v}$.*

Proof. We choose primes P and P' each large enough for Lemma 9.2, where $\gcd(P-1, P'-1) = 2m$ and $\boldsymbol{v}$ is not a zero divisor in either R_P or $R_{P'}$ (using Dirichlet's theorem on primes in arithmetic progression and the fact that $\boldsymbol{v}$ may be a zero divisor in R_Q for only a finite number of primes Q). By Lemma 9.2, we can compute chains that will allow us to compute $\boldsymbol{v}^{P-1} \bmod p_i$ and $\boldsymbol{v}^{P'-1} \bmod p_i$ in polynomial time for any prime p_i such that the values $\overline{\boldsymbol{v}_r}$ and $\overline{\boldsymbol{v}_r}'$ in the chains are not zero divisors in R_{p_i}. Choose a set of primes $p_1, \ldots, p_t$ that satisfy this condition and such that $2\|\boldsymbol{v}^{2m}\|_\infty < \prod p_i$. (We simply avoid the finite number of problematic primes.) Apply the Euclidean algorithm in the exponent to compute $\boldsymbol{v}^{2m}$ modulo each p_i, and ultimately $\boldsymbol{v}^{2m}$ exactly using the Chinese Remainder Theorem. □

Lemma 9.4 (Similar to [GS02])**.** *Let $\boldsymbol{v} \in R$. Let $\boldsymbol{w} = \boldsymbol{v}^r$ where $2m$ divides r. Then, given $\boldsymbol{w}$, we may output a list L of r values $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_r$ in time polynomial in r and the bit length of $\boldsymbol{w}$, such that L includes $\boldsymbol{v}$.*

Lemma 9.4 may seem trivial, and it certainly would be if r and m were relatively prime. In this case, one could simply pick a prime $Q > 2\|\boldsymbol{v}\|_\infty$ with $\gcd(r, Q-1) = 1$, set $s = r^{-1} \bmod m(Q-1)$, and compute $\boldsymbol{w}^s = \boldsymbol{v}^{rs} = \boldsymbol{v}^{1+km(Q-1)} = \boldsymbol{v}$ in R_Q (by Fermat's Little Theorem), which yields $\boldsymbol{v}$ exactly. Things become more complicated when $\gcd(r, m) \neq 1$.

Proof. First, we observe that $\boldsymbol{w}$ does not uniquely determine $\boldsymbol{v}$. Specifically, for any $\mathbf{e} = \pm x^i \in R$ (the $2m$ values that are plus or minus an m-th root of unity in R), we have that $\boldsymbol{v} \cdot \mathbf{e}$ is also in R and $\boldsymbol{w} = (\boldsymbol{v} \cdot \mathbf{e})^r$. However, we show that fixing $\boldsymbol{v}$'s value at any (complex) primitive m-th root of unity ζ_m also fixes $\boldsymbol{v}$'s value at the other primitive m-th roots of unity, after which we may obtain $\boldsymbol{v}$ via interpolation. Given $\boldsymbol{w}(\zeta_m) = \boldsymbol{v}(\zeta_m)^r$, there are only r possibilities for $\boldsymbol{v}(\zeta_m)$. By iterating the procedure below for each possibility of $\boldsymbol{v}(\zeta_m)$, the procedure will eventually use the "correct" value, and the correct value of $\boldsymbol{v}$ will be included in the output.

For any prime Q, by an extension of Fermat's Little Theorem, we have that $\boldsymbol{a}(x)^Q = \boldsymbol{a}(x^Q)$ in the ring R_Q. Let $Q = cr - b$ be a prime for some positive integers $b < r$ and c such that $\boldsymbol{w}$ is not a zero divisor in R_Q and $\gamma_\infty \cdot \|\boldsymbol{w}\|_\infty^{can} < Q/2$. (Where that γ_∞ denotes the maximal value of $\|\boldsymbol{a}\|_\infty / \|\boldsymbol{a}\|_\infty^{can}$ for $\boldsymbol{a} \in K$.) Given that m divides r, we compute that $(\boldsymbol{v}^r)^c = \boldsymbol{v}^Q \boldsymbol{v}^b = \boldsymbol{v}(x^Q)\boldsymbol{v}^b = \boldsymbol{v}(x^{-b})\boldsymbol{v}^b \bmod Q$. Since $\gamma_\infty^{can} \cdot \|\boldsymbol{v}(x^{-b})\boldsymbol{v}^b\|_\infty^{can} \leq \gamma_\infty \cdot \|\boldsymbol{w}\|_\infty < Q/2$, we efficiently recover the term $\mathbf{z}_b \leftarrow \boldsymbol{v}(x^{-b})\boldsymbol{v}^b$ exactly. This allows us to compute $\boldsymbol{v}(\zeta_m^{-b}) = \mathbf{z}_b(\zeta_m)/\boldsymbol{v}(\zeta_m)^b$. By choosing other Q's, we similarly compute $\mathbf{z}_b$ for each $b \in \mathbb{Z}_m^*$, thereby compute $\boldsymbol{v}(\zeta)$ for all complex primitive m-th roots of unity ζ, and thus recover $\boldsymbol{v}$. □

Theorem 9.5 ([GS02])**.** *Let $\boldsymbol{v} \in R$. Given $\boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and the HNF basis B for the ideal lattice $\langle \boldsymbol{v} \rangle$, we can compute $\boldsymbol{v}$ in time polynomial in m and the bit-length of $\boldsymbol{v}$.*

Proof. This follows from Lemmas 9.3 and 9.4. □

Some Extensions. Howgrave-Graham and Szydlo [HGS04] observed that one can use the GS algorithm to recover $\boldsymbol{v}$ from the relative norm $\mathsf{N}_{K/K^+} = \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ *without* a basis of $\langle \boldsymbol{v} \rangle$, as long as one has a factorization of $\mathsf{N}_{K/Q}(\boldsymbol{v} \cdot \overline{\boldsymbol{v}}) = \mathsf{N}_{K/Q}(\boldsymbol{v})^2$. The idea is that, from $\mathsf{N}_{K/K^+} = \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ and the factorization, one can use Kummer-Dedekind (Theorem 8.3) to generate a basis of some $\boldsymbol{v}'$ such that $\boldsymbol{v}' \cdot \overline{\boldsymbol{v}'} = \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$ ($\boldsymbol{v}$ may not be unique). If $\mathsf{N}_{K/Q}(\boldsymbol{v})$ is composite, one can compute its factorization using a classical sub-exponential factorization algorithm such as the number field sieve [LLMP90, LL93] or Shor's polynomial-time quantum algorithm [Sho97a].

Another way to view the GS and HS algorithms is the following. The averaging attack yields the *Gram matrix* (essentially the co-variance matrix) $\boldsymbol{B}_{priv}^T \cdot \boldsymbol{B}_{priv}$ associated to the secret lattice basis of the signer. In early NTRU signature schemes, this Gram matrix happened to have a very special form; it corresponded to the relative norm $\mathsf{N}_{K/K^+}(\boldsymbol{v}) = \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$. The GS and HS algorithms are able to *factor the Gram matrix* in this special case (using the auxiliary information $\langle \boldsymbol{v} \rangle$ in the case of the GS algorithm).

The NTRUSign signature scheme [HHGP$^+$03] was proposed shortly after the Gentry-Szydlo attack was announced. As noted in [GS02, HGS04], for NTRUSign, applying an averaging attack similar to that described in Section 9.1 still yields the Gram matrix $\boldsymbol{B}_{priv}^T \cdot \boldsymbol{B}_{priv}$ associated to the secret lattice basis of the signer. However, the Gram matrix in NTRUSign has a more complicated form than in previous NTRU signature schemes. In particular, it is a 2 × 2 block of ring elements:

$$\boldsymbol{B}_{priv}^T \cdot \boldsymbol{B}_{priv} = \begin{bmatrix} \boldsymbol{v} \cdot \overline{\boldsymbol{v}} + \mathbf{V} \cdot \overline{\mathbf{V}} & \boldsymbol{w} \cdot \overline{\boldsymbol{v}} + \mathbf{W} \cdot \overline{\mathbf{V}} \\ \boldsymbol{v} \cdot \overline{\boldsymbol{w}} + \mathbf{V} \cdot \overline{\mathbf{W}} & \boldsymbol{w} \cdot \overline{\boldsymbol{w}} + \mathbf{W} \cdot \overline{\mathbf{W}} \end{bmatrix}$$

where $\boldsymbol{v}$, $\boldsymbol{w}$, $\mathbf{V}$ and $\mathbf{W}$ are short elements that constitute the signer's private key. It remains an open problem to efficiently factor Gram matrices of this form (as well as general Gram matrices), even when given a basis (e.g., the HNF basis) of the lattice generated by $\boldsymbol{B}_{priv}$. Szydlo [Szy03] showed that the Gram matrix factorization problem can be reduced to an oracle that distinguishes whether two Gram matrices are associated to bases of the same lattice, but it is unknown how to instantiate this oracle efficiently in general.

The GS algorithm suggests an open problem about other relative norms: Is it possible to efficiently recover $\boldsymbol{v}$ from $\langle \boldsymbol{v} \rangle$ and the relative norm $\mathsf{N}_{K/L}(\boldsymbol{v})$ when L is some subfield of K other than the index-2 real subfield K^+? When $L = \mathbb{Q}$, this is just the Principal Ideal Generator problem, which seems infeasible in general, but perhaps the problem is feasible when the index $[K : L]$ is small or smooth. For example, suppose K is the m-th cyclotomic field for $m = 2^k$ and L is an index-4 subfield. In this case, can we efficiently recover $\boldsymbol{v}$ from $\langle \boldsymbol{v} \rangle$ and $\mathsf{N}_{K/L}(\boldsymbol{v})$? Can we, perhaps, first recover $\mathsf{N}_{K/K^+}(\boldsymbol{v})$ from $\langle \boldsymbol{v} \rangle$ and $\mathsf{N}_{K/L}(\boldsymbol{v})$, and then use the GS algorithm to recover $\boldsymbol{v}$? It seems doubtful, since the GS algorithm relies implicitly on the fact that $\langle \boldsymbol{v} \rangle$ and $\mathsf{N}_{K/K^+}(\boldsymbol{v})$ define $\boldsymbol{v}$ uniquely up to torsion units, due to the special relationship between the cyclotomic units and the subfield K^+.

We remark that it is interesting that, while the GS algorithm clearly relies on the structure of the cyclotomic unit group, this reliance is implicit; it would be worthwhile to make the connection more explicit.

9.3 Nguyen-Regev: A Gradient Descent Attack

Nguyen and Regev [NR09] described how to extend averaging and key recovery attacks to signature schemes based on *general lattices* – in particular, to lattices underlying the GGH [GGH97] and NTRUSign [HHGP+03] signature schemes (for suggested parameters). These attacks show that averaging a transcript of lattice-based signatures can be a devastating attack in general, and further recommend the approach taken by [GPV08] of ensuring that the distribution of signatures has some canonical distribution (e.g., a Gaussian distribution) that is essentially independent of the particular lattice basis that the signer is using.

Their attack is designed to "learn a parallelepiped". That is, given samples $\{\boldsymbol{B}_{priv} \cdot \mathbf{y}_{\boldsymbol{i}}\}$ where the $\mathbf{y}_i$'s are (discretely) uniform over a hypercube, their attack converges upon the shape of $\mathcal{P}(\boldsymbol{B}_{priv})$ and ultimately outputs the private basis $\boldsymbol{B}_{priv}$.

To understand the NR attack, it might help to understand why previous attacks failed to break GGH and NTRUSign. Previous attacks, were (in some sense) too modular. They divided the attack into two parts: 1) an averaging/covariance/second-moment attack which used samples $\{\boldsymbol{B}_{priv} \cdot \mathbf{y}_{\boldsymbol{i}}\}$ to recover the Gram matrix $\boldsymbol{B}_{priv}^T \cdot \boldsymbol{B}_{priv}$ associated to the secret lattice basis of the signer, and 2) a "factoring" attack that either factored the relative norm [GS02, HGS04] or otherwise tried to factor the Gram matrix [Szy03]. The second step, the factoring attack, sometimes used a lattice basis as auxiliary information (as in the GS algorithm). But, crucially, the second step *did not use the samples.* After using the samples to obtain the Gram matrix (and a lattice basis), previous attacks simply discarded the samples. In this case, key recovery reduces to the Gram matrix factorization problem (with a lattice basis), for which no general polynomial-time algorithm is known.

In contrast, the NR algorithm is (in some sense) less modular. They use the samples throughout the attack. In particular, they first show that the 4-th moment (also known as the *kurtosis*) of a transcript of signatures defines a global minimum related to the secret key. (Recall that, for a set of vectors $\boldsymbol{B} = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_n}\} \in GL_n(\mathbb{R})$, the k-th moment of the parallelepiped $\mathcal{P}(\boldsymbol{B})$ over a vector $\boldsymbol{w}$ is defined as $\mathrm{mom}_{B,k}(\boldsymbol{w}) = \mathrm{Exp}[\langle \boldsymbol{u}, \boldsymbol{w} \rangle^k]$ where $\boldsymbol{u}$ is chosen uniformly over $\mathcal{P}(\boldsymbol{B})$.) The group of $n \times n$ invertible matrices with real coefficients will be denoted by $GL_n(\mathbb{R})$ and $O_n(R)$ will denote the subgroup of orthogonal matrices.

Lemma 9.6 (Lemma 3 in [NR09])**.** *Let $\boldsymbol{B} = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_n}\} \in O_n(\mathbb{R})$. Then the global minimum of $\mathrm{mom}_{B,4}(\boldsymbol{w})$ over the unit sphere of $\mathbb{R}^n$ is $1/5$ and this minimum is obtained at $\pm \boldsymbol{b_1}, \ldots, \pm \boldsymbol{b_n}$. There are no other local minima.*

Then, they use *gradient descent* to find this global minimum approximately, using the samples at each stage of the descent to approximate the gradient function. This leads to the following theorem.

Theorem 9.7 (Theorem 4 in [NR09])**.** *For any $c_0 > 0$ there exists a $c_1 > 0$ such that given n^{c_1} samples uniformly distributed over some parallelepiped $\mathcal{P}(\boldsymbol{B})$, $\boldsymbol{B} = \{\boldsymbol{b_1}, \ldots, \boldsymbol{b_n}\} \in GL_n(\mathbb{R})$, the approximate gradient descent algorithm outputs with constant probability a vector $B \cdot \tilde{\mathbf{e}}$ where $\tilde{\mathbf{e}}$ is within ℓ_2 distance n^{-c_0} of some standard basis vector $\mathbf{e}_i$.*

Assuming the approximate solution output by the NR algorithm is "good enough" – that is, good enough to obtain B exactly via rounding – the NR attack succeeds. The secret bases in GGH and NTRUSign have small entries (polynomial in the security parameter), and so the NR attack succeeds asymptotically with only a polynomial number of signatures, and also performs quite well in practice for suggested parameters.

One issue that the NR attack leaves somewhat unresolved is: What happens when the approximate solution output by the NR algorithm is not "good enough" to use rounding to get the exact solution? Nguyen and Regev suggest using a CVP approximation algorithm, which they observe performs reasonably well in practice on suggested parameters, but which of course is not polynomial-time in general. This is a weakness also of the averaging attack described in Section 9.1. This weakness suggests an obvious way of fixing the schemes: choose the secret basis so that its entries are *super-polynomial* or even *sub-exponential* integers, so that averaging attacks cannot approximate the entries of the basis precisely enough to obtain them exactly via rounding. (Of course, this makes the cryptographic construction less practical, but still polynomial-time.)

In Section 9.5, we describe an attack that casts doubt on this fix, at least in the context of ideal lattices. We show that we can recover $\boldsymbol{v}$ from $\langle \boldsymbol{v} \rangle$ and a ϵ-approximation $\mathbf{u}$ of $\boldsymbol{v}$ when ϵ is inverse-quasi-polynomial, even when the coefficients of $\boldsymbol{v}$ are arbitrarily large.

9.4 Ducas-Nguyen: Gradient Descent over Zonotopes and Deformed Parallelepipeds

The Nguyen-Regev algorithm was designed to "learn a parallelepiped", Ducas and Nguyen [DN12b] showed how to extend the algorithm to learn more complicated shapes, including zonotopes and deformed parallelepipeds.

Recall that the parallelepiped associated to a basis $\boldsymbol{B} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n\}$ is the set $\mathcal{P}(\boldsymbol{B}) = \{\sum x_i \cdot \boldsymbol{b}_i : x_i \in [-1/2, 1/2)\}$. Under certain circumstances (see Section 9.3), Nguyen-Regev learns the parallelepiped $\mathcal{P}(\boldsymbol{B})$ from samples of the form $\{\boldsymbol{B} \cdot \boldsymbol{r}\}$, where $\boldsymbol{r} = (r_1, \ldots, r_n)$ is (discretely) uniform over an n-dimensional hypercube. This algorithm breaks certain signature schemes, such as the basic version of NTRUSign [HHGP$^+$03], where a transcript of signatures implicitly provides samples $\{\boldsymbol{B}_{priv} \cdot \boldsymbol{r}\}$ where $\boldsymbol{B}_{priv}$ is the signer's private basis. A zonotope is a generalization of a parallelepiped to a dependent set of vectors. Let $\boldsymbol{M} = \{\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m\}$ be a $n \times m$ matrix for $m > n$. The zonotope formed by $\boldsymbol{M}$ is the set $\mathcal{Z}(\boldsymbol{M}) = \{\sum x_i \cdot \boldsymbol{b}_i : x_i \in [-1/2, 1/2)\}$. Even though the vectors of $\boldsymbol{M}$ are dependent and the zonotope has a shape that is "closer to spherical" than a parallelepiped (the corners typically have more obtuse angles), Ducas and Nguyen show the Nguyen-Regev algorithm can be extended to this setting, when the samples have the form $\{\boldsymbol{M} \cdot \boldsymbol{r}\}$, where $\boldsymbol{r}$ is (discretely) uniform over an m-dimensional hypercube. Their new algorithm does not provably always work, but it works quite well in practice. They used their algorithm to break a version of NTRUSign with a "perturbations" countermeasure. In NTRUSign with perturbations, the signer uses perturbations to obscure its private basis, in such a way that a transcript of signatures induces the distribution of a zonotope rather than a parallelepiped.

Can the Nguyen-Regev and Ducas-Nguyen algorithms be extended even further? For example, suppose we have samples of the form $\{\boldsymbol{B} \cdot \boldsymbol{r}\}$ or $\{\boldsymbol{M} \cdot \boldsymbol{r}\}$, where $\boldsymbol{r}$ comes from a discrete Gaussian distribution. In these cases, assuming that the coordinates of $\boldsymbol{r}$ have moderate deviation, one can show [Pei10, AGHS12] that the samples also have a discrete Gaussian distribution over the lattice generated by $\boldsymbol{B}$ or $\boldsymbol{M}$, where the Gaussian is ellipsoidal according to the shape of $\boldsymbol{B}$ or $\boldsymbol{M}$. In the latter case, the ellipsoid get closer to a sphere as m gets larger relative to n (in the sense that the singular values of $\boldsymbol{M}$ get closer together). A discrete ellipsoidal Gaussian does not have any "corners" like a parallelepiped or zonotope, which are the local minima of the Nguyen-Regev and Ducas-Nguyen algorithms. This fact seems to prevent a direct application of Nguyen-Regev or Ducas-Nguyen. However, the shape of the ellipsoid still may provide some useful information.[1]

Interestingly, the re-randomization algorithm of our construction (see Section 6) involves adding a term of the form $(\boldsymbol{M} \cdot \boldsymbol{r})/\mathbf{z}$, where $\boldsymbol{r}$ has a spherical Gaussian distribution. Consequently, the numerator of this added term has an ellipsoidal Gaussian distribution, where the numerator's shape depends on the shape of $\boldsymbol{M}$. Note that as opposed to the case of signatures, re-randomization in our construction is not supposed to hide $\boldsymbol{M}$ (in fact we give out $\boldsymbol{M}/\mathbf{z}$ in the public parameters). Rather, the purpose of re-randomization in is just to "drown out" the initial value that is being randomized (while preserving its coset wrt the ideal $\mathcal{I}$).

9.5 A New Algorithm for the Closest Principal Ideal Generator Problem

As usual, let R be the ring of integers for the m-th cyclotomic field. Let $\boldsymbol{v} \in R$ and $\mathcal{I} = \langle \boldsymbol{v} \rangle$. Let $\mathbf{u}$ be a ϵ-approximation of $\boldsymbol{v}$ – i.e., $1/(1+\epsilon) \leq |\sigma_k(\boldsymbol{v})/\sigma_k(\mathbf{u})| \leq 1+\epsilon$ for all $k \in \mathbb{Z}_m^*$. How efficiently can we recover the principal ideal generator $\boldsymbol{v}$ from $\mathcal{I}$ and $\mathbf{u}$?

A cryptanalyst would hope that we can recover $\boldsymbol{v}$ whenever ϵ is bounded by some inverse-polynomial function, so that the averaging and Nguyen-Regev attacks become more devastating. Recall that the averaging and Nguyen-Regev attacks only output a $1/\mathsf{poly}$-approximate solution of $\boldsymbol{v}$ (or a related value) when given a polynomial number of samples; afterward, the attacks attempt to output an exact solution by rounding (or by solving approximate-CVP, but this is not efficient in general). Thus, the averaging and Nguyen-Regev attacks can easily be escaped by choosing $\boldsymbol{v}$ so that its coefficients are super-polynomial in size. However, a cryptanalyst could prevent this escape with an efficient algorithm to recover $\boldsymbol{v}$ from a $1/\mathsf{poly}$-approximation of $\boldsymbol{v}$, since this would break the scheme regardless of how large $\boldsymbol{v}$'s coefficients are.

Here, we show how to recover $\boldsymbol{v}$ in time polynomial in m and the bit-length of $\boldsymbol{v}$, assuming that ϵ is bounded by some inverse-quasi-polynomial function in m. This algorithm does

[1] For signature schemes, the signer can use the Gaussian samplers from [GPV08, Pei10] to get a perfectly spherical distribution, thus ensuring that the transcript of signatures "leaks no information at all."

not quite fulfill the cryptanalyst's dream, but it suggests a direction for future, possibly more devastating attacks. The algorithm that we describe here is a natural extension of the Gentry-Szydlo algorithm ([GS02], see Section 9.2). Whereas the GS algorithm uses the *exact* information about $\boldsymbol{v}$'s geometry provided by the relative norm $\mathsf{N}_{K/K^+}(\boldsymbol{v}) = \boldsymbol{v} \cdot \overline{\boldsymbol{v}}$, our algorithm here tries to make-do with the approximate information provided by $\mathbf{u}$.

The algorithm follows the algebraic strategy of the GS algorithm. In particular, it invokes Fermat's Little Theorem to assert that $\boldsymbol{v}^r = 1 \bmod P$ for prime P when $(P-1)$ and m divide r (as long as $\boldsymbol{v}$ is not a zero divisor in R_P). Next, it applies (implicit) lattice reduction to the lattice $\mathcal{I}^r$ to obtain a reduced element $\boldsymbol{w} = \boldsymbol{v}^r \cdot \boldsymbol{a}$. Finally, it tries to recover $\boldsymbol{a}$ (and hence $\boldsymbol{v}$) by using the fact that $\boldsymbol{a} = \boldsymbol{w} \bmod P$. The main differences between the GS algorithm and our algorithm are:

- We require r to be only quasi-polynomial (not exponential): The GS algorithm has exact information about $\boldsymbol{v}$'s geometry, which allows it to derive exact information about $\boldsymbol{v}^r$'s geometry even when r is exponential (though this information is represented implicitly in the polynomial chains). In contrast, we only have approximate information about $\boldsymbol{v}$'s geometry, and the accuracy of our information about $\boldsymbol{v}^r$'s geometry degrades exponentially with r. So, we cannot have r much bigger than $1/\epsilon$.

- We will work modulo the product of many primes: To compensate for the fact that r cannot be too large in our setting, we choose r so that $(p_i - 1)$ divides r for *many* primes p_i, and we work modulo $P = \prod p_i$. We heuristically estimate that we can achieve $P = 2^{\Omega(m)}$ when $r = 2^{O(\log m \log\log m)}$. (Similar to the GS algorithm, we need P to exceed the LLL approximation factor, and then some.)

Let us begin by considering how to set r and P. For some k to be determined, let $q_1, \ldots, q_k$ be the first k primes, and set $r_{k,m} = m \prod q_i$. Set $\mathcal{S}_{k,m}$ be the set of 2^k products of m with a subset product of $q_1, \ldots, q_k$. Set $\mathcal{T}_{k,m} = \{1 + s : s \in \mathcal{S}_{k,m}\}$, $\mathcal{P}_{k,m} = \{\text{prime } p \in T_{k,m}\}$, and $P_{k,m} = \prod_{p \in \mathcal{P}_{k,m}} p$. We claim that $(r_{k,m}, P_{k,m})$ will tend to be a good choice for (r, P). Certainly it is true that $r_{k,m}$ is divisible by $p_i - 1$ for the primes that divide P; the remaining issue is the size of $r_{k,m}$ and $P_{k,m}$.

First, consider the size of $r_{k,m}$. We have:

$$\ln r_{k,m} = \ln m + \sum_{i=1}^{k} \ln q_i = \ln m + q_k + o(k) = \ln m + k \ln k + o(k \ln k),$$

where the second and third equalities follow from extensions of the Prime Number Theorem (see Corollaries 8.2.7 and 8.2.8 in [BS96]). Assuming $k \ln k$ dominates m, we have $r_{k,m} = 2^{(1+o(1))k \ln k}$.

Now, consider the size of $P_{k,m}$. Clearly, many elements of $\mathcal{T}_{k,m}$ are not prime. For example, $1 + s$ cannot be prime unless s is divisible by 2 – i.e., unless 2 is part of the subset product that forms s. Similarly, if s is a subset product not divisible by 3, then $1 + s$ has (roughly) only a 1/2 (versus the usual 1/3) probability of not being divisible by 3. But, aside from such observations, we would heuristically expect that, by the Prime Number Theorem, an element $t \in T_{k,m}$ has a $\Omega(1/\ln t)$ chance of being prime. With this heuristic, we calculate:

$$P_{k,m} = \prod_{p \in \mathcal{P}_{k,m}} p = \prod_{t \in \mathcal{T}_{k,m}} t^{\Omega(1/\ln t)} = 2^{\Omega(|\mathcal{T}_{k,m}|)} = 2^{\Omega(2^k)} .$$

Assuming these heuristic estimates of $r_{k,m}$ and $P_{k,m}$ are true, then for any constant c_1, there is a constant c_2, such that setting $k = \lfloor \ln m \rfloor + c_2$ ensures that $P_{k,m}$ is at least $2^{c_1 \cdot m}$. With this value of k, we have $r_{k,m} = 2^{(1+o(1))\ln m \ln\ln m} = m^{(1+o(1))\ln 2 \ln\ln m}$. In other words, while $P_{k,m}$ is exponential in m, $r_{k,m}$ is only slightly quasi-polynomial in m. For convenience, we capture these observations in the following claim.

Claim 9.8. *Let $\rho_m(x)$ denote the smallest positive integer such that there exist distinct primes $\{p_i\}$ such that $\prod p_i \geq x$ and $\rho_m(x)$ is divisible by m and $(p_i - 1)$ for all i. Then, for $x = 2^{\Omega(m)}$, we have $\rho_m(x) = 2^{(1+o(1))\ln\ln x \ln\ln\ln x}$. For $x = 2^{\Theta(m)}$, we have $\rho_m(x) = m^{(1+o(1))\ln\ln m}$. The "proof" of the claim is constructive – that is, one can (heuristically) generate a value $r_{k,m}$ that meets these asymptotic bounds of $\rho_m(x)$ by setting $r_{k,m}$ to be the product of m with the first $c + \ln\ln x$ primes for some constant c.*

Next, we revisit Lemma 9.2, adapting implicit lattice reduction and the polynomial chains of the GS algorithm to our setting.

Lemma 9.9 (Adaptation of Lemma 9.2)**.** *Let $\boldsymbol{v}_0 \in R$ and let B_0 be the HNF basis B_0 for the ideal lattice $\mathcal{I}_0 = \langle \boldsymbol{v}_0 \rangle$. Let $\mathbf{u}_0$ be an ϵ-approximation of $\boldsymbol{v}_0$ – i.e., $1/(1+\epsilon) \leq |\sigma_k(\boldsymbol{v}_0)/\sigma_k(\mathbf{u}_0)| \leq 1+\epsilon$ for all $k \in \mathbb{Z}_m^*$. Let $k = \sum k_i 2^i$ with $k_i \in \{0,1\}$ be an integer with $r = \lfloor \log_2 k \rfloor$. Let P be an integer such that $\boldsymbol{v}_0$ is not a zero divisor in R_P. Then, given the input $(B_0, \mathbf{u}_0)$, we may compute, in time polynomial in r, m, and the bit-length of the input, the chains:*

$$\{\boldsymbol{v}_0^{k_{r-1}} \cdot \boldsymbol{v}_0^2/\boldsymbol{v}_1, \ldots, \boldsymbol{v}_0^{k_0} \cdot \boldsymbol{v}_{r-1}^2/\boldsymbol{v}_r\}$$

where for all $i > 0$, no $\boldsymbol{v}_i$ is a zero divisor in R_P, and $\|\boldsymbol{v}_i\|_2^{can} < 2^{(n-1)/2}\sqrt{n}(1+\epsilon)^{k^{(i)}}$, where $k^{(i)}$ is the integer formed by the $i+1$ most significant bits of k. Using these chains, we may compute $\boldsymbol{v}_0^k/\boldsymbol{v}_r \bmod P$ in polynomial time. If k and P are such that $\boldsymbol{v}_0^k = 1 \bmod P$ and $P > 2^{(n+1)/2}\sqrt{n}(1+\epsilon)^k \gamma_2$, we may compute $\boldsymbol{v}_r$ exactly, and thereafter use the above chains to compute $\boldsymbol{v}_0^k \bmod Q$ in polynomial time for any prime Q such that $\boldsymbol{v}_r$ is not a zero divisor in R_Q.

Proof. Consider the first term of the first chain: $\boldsymbol{v}_0^{k_{r-1}} \cdot \boldsymbol{v}_0^2/\boldsymbol{v}_1$. For convenience, let $c = 2k_r + k_{r-1}$. Given $(B_0, \mathbf{u}_0)$, we efficiently compute a basis B_0' for the ideal $\mathcal{I}_0' = \langle \mathbf{u}_0^c \rangle / \mathcal{I}^c$. Apply LLL to B_0'. Set $\mathbf{u}_1 \in \mathcal{I}_0'$ to be the element corresponding to the shortest vector in the reduced basis. Since $\mathcal{I}_0'$ is a principal (fractional) ideal, we have $\mathbf{u}_1 = (\mathbf{u}_0/\boldsymbol{v}_0)^c \boldsymbol{v}_1$ for some $\boldsymbol{v}_1 \in R$. (To handle the possibility that $\boldsymbol{v}_1$ is a zero divisor in R_P, use techniques by Gentry and Szydlo.) Since $\boldsymbol{v}_1 = \mathbf{u}_1 \cdot (\boldsymbol{v}_0/\mathbf{u}_0)^c$, we have that $\|\boldsymbol{v}_1\|_2^{can} \leq 2^{(n-1)/2} \cdot \sqrt{n} \cdot (1+\epsilon)^c$ by the guarantee of LLL and the fact $\|\boldsymbol{v}_0^c/\mathbf{u}_0^c\|_\infty^{can} \leq (1+\epsilon)^c$. Include the term $\mathbf{u}_0^c/\mathbf{u}_1 = \boldsymbol{v}_0^c/\boldsymbol{v}_1$ in the polynomial chain. Observe that $\mathbf{u}_1$ is a $(1+\epsilon)^c$ approximation of $\boldsymbol{v}_1$. Also, we can efficiently generate a basis B_1 of the ideal $\mathcal{I}_1 = \langle \boldsymbol{v}_1 \rangle = \langle \mathbf{u}_1 \rangle / \mathcal{I}_0'$.

The second term in the chain is supposed to be $\boldsymbol{v}_0^{k_{r-2}} \cdot \boldsymbol{v}_1^2 / \boldsymbol{v}_2$. Given $(B_0, B_1, \mathbf{u}_0, \mathbf{u}_1)$, we efficiently compute a basis B_1' for the ideal $\mathcal{I}_1' = \left\langle \mathbf{u}_0^{k_{r-2}} \mathbf{u}_1^2 \right\rangle / (\mathcal{I}_0^{k_{r-2}} \mathcal{I}_1^2)$. Apply LLL to B_1'. Set $\mathbf{u}_2 \in \mathcal{I}_1'$ to be the element corresponding to the shortest vector in the reduced basis. Since $\mathcal{I}_1'$ is a principal (fractional) ideal, we have $\mathbf{u}_2 = (\mathbf{u}_0/\boldsymbol{v}_0)^{k_{r-2}} (\mathbf{u}_1/\boldsymbol{v}_1)^2 \boldsymbol{v}_2$ for some $\boldsymbol{v}_2 \in R$. (To handle the possibility that $\boldsymbol{v}_2$ is a zero divisor in R_P, use techniques by Gentry and Szydlo.) Since $\boldsymbol{v}_2 = \mathbf{u}_2 \cdot (\boldsymbol{v}_0/\mathbf{u}_0)^{k_{r-2}} (\boldsymbol{v}_1/\mathbf{u}_1)^2$, we have that $\|\boldsymbol{v}_2\|_2^{can} \leq 2^{(n-1)/2} \cdot \sqrt{n} \cdot (1+\epsilon)^{4k_r + 2k_{r-1} + k_{r-2}}$ by the guarantee of LLL and the fact $\|(\boldsymbol{v}_0/\mathbf{u}_0)^{k_{r-2}} (\boldsymbol{v}_1/\mathbf{u}_1)^2\|_\infty^{can} \leq (1+\epsilon)^{4k_r + 2k_{r-1} + k_{r-2}}$. Include the term $\mathbf{u}_0^{k_{r-2}} \cdot \mathbf{u}_1^2 / \mathbf{u}_2 = \boldsymbol{v}_0^{k_{r-2}} \cdot \boldsymbol{v}_1^2 / \boldsymbol{v}_2$ in the polynomial chain. Observe that $\mathbf{u}_2$ is a $(1+\epsilon)^{4k_r + 2k_{r-1} + k_{r-2}}$ approximation of $\boldsymbol{v}_2$. Also, we can efficiently generate a basis B_2 of the ideal $\mathcal{I}_2 = \langle \boldsymbol{v}_2 \rangle = \langle \mathbf{u}_2 \rangle / \mathcal{I}_1'$. One continues in this fashion until all the terms in the polynomial chain are computed.

The rest of the proof proceeds similar to the proof of Lemma 9.2. □

Since in Lemma 9.2 k may be super-polynomial, we prefer not to compute $\boldsymbol{v}_0^k$ directly. Instead, as in Lemma 9.3, we may compute $\boldsymbol{v}_0^{2m}$ by computing $\boldsymbol{v}_0^{k_1}$ and $\boldsymbol{v}_0^{k_2}$ for which $\gcd(k_1, k_2) = 2m$, and then applying the Euclidean algorithm in the exponent.

Lemma 9.10. *Let $\boldsymbol{v} \in R$ and let B be the HNF basis for the ideal lattice $\mathcal{I} = \langle \boldsymbol{v} \rangle$. Let $\mathbf{u}$ be an ϵ-approximation of $\boldsymbol{v}$ – i.e., $1/(1+\epsilon) \leq |\sigma_k(\boldsymbol{v})/\sigma_k(\mathbf{u})| \leq 1+\epsilon$ for all $k \in \mathbb{Z}_m^*$. Then, given $\mathbf{u}$ and $\boldsymbol{B}$, we may compute $\boldsymbol{v}^{2m}$ in time polynomial in m and the bit length of $\boldsymbol{v}$.*

Proof. Similar to the proof of Lemma 9.3. □

Theorem 9.11. *Assuming Claim 9.8, there is an $\epsilon = m^{-(1+o(1)) \ln \ln m}$ such that, given the HNF basis for the ideal lattice $\mathcal{I} = \langle \boldsymbol{v} \rangle$ for some $\boldsymbol{v} \in R$ and an ϵ-approximation $\mathbf{u}$ of $\boldsymbol{v}$, we can compute $\boldsymbol{v}$ in time polynomial in m and the bit-length of $\boldsymbol{v}$.*

Proof. This follows from Lemmas 9.10 and 9.4 and Claim 9.8. □

We remark that this algorithm implies that the bounded distance decoding problem (BDDP) is *easy* for the Dirichlet unit lattice Λ for surprisingly low approximation factors. (Recall from Section 8 that the Dirichlet unit lattice is the lattice formed by the image of the units under the map $\lambda : K^* \to \mathbb{R}^{s_1+s_2}$ given by $\lambda(\boldsymbol{a}) = (\ln|\sigma_1(\boldsymbol{a})|, \ldots, \ln|\sigma_{s_1+s_2}(\boldsymbol{a})|)$.) Specifically, by the above algorithm, given an ϵ-approximation $\mathbf{u}$ of a unit $\boldsymbol{v}$, we can recover $\boldsymbol{v}$ exactly. So, in the Dirichlet unit lattice, taking logarithms, given a vector $\lambda(\mathbf{u})$ whose ℓ_∞ distance from Λ is at most $\ln(1+\epsilon) \approx \epsilon$, we can efficiently recover the vector in Λ-vector closest to $\lambda(\mathbf{u})$. Really, this corollary is not *so* surprising, since in the case of the m-th cyclotomic field for prime power m we already have in our hands a fairly short basis of Λ given by the basis $\{\lambda(\boldsymbol{b}_i) : \boldsymbol{b}_i = (1-\zeta_m^i)/(1-\zeta_m) : i \in \mathbb{Z}_m^*\}$, which gives more direct ways of achieving the same result. What is interesting is that, as with the GS algorithm, the algorithm above does not *explicitly* use the structure of the unit group, though of course it must be doing so implicitly; it would be interesting to make the connection more explicit.

9.6 Coppersmith Attacks

Coppersmith-type attacks [Cop96b, Cop96a] would seem to be ideally suited to ideal lattices, as these attacks elegantly combine algebra and geometry. Somewhat surprisingly, however, they have not yet resulted in attacks that are more effective than generic lattice reduction algorithms.

Cohn and Heninger [CH11] applied Coppersmith's method to solving the BDDP over ideal lattices. In the BDDP over ideal lattices, one is given a basis $\boldsymbol{B}$ of an ideal lattice $\mathcal{I} \subset \mathcal{O}_K$ and an element $\mathbf{u} \in \mathcal{O}_K$ that is very close to some $\boldsymbol{v} \in \mathcal{I}$; the task is to output $\boldsymbol{v}$. Following Coppersmith's method, and to oversimplify a bit, Cohn and Heninger let $\mathbf{x} = \mathbf{u} - \boldsymbol{v}$ be the small unknown offset, and generate numerous univariate polynomials that have $\mathbf{x}$ as a root modulo $\mathcal{I}^t$ for some large exponent t. For example, any polynomial of the form $\boldsymbol{a}^r \cdot (\mathbf{u} - X)^{t-r}$ with $\boldsymbol{a} \in \mathcal{I}$ evaluates at $\mathbf{x}$ to an element that is in $\mathcal{I}^t$, and therefore any linear combination of such polynomials does as well. These polynomials form a lattice, and they apply LLL to this lattice to find a polynomial $p(X)$ with (somewhat) small coefficients. They design the lattice so that $p(\mathbf{x})$ is small (by the smallness of p's coefficient vector and of $\|\mathbf{x}\|_\infty$), indeed smaller than any nonzero element in $\mathcal{I}^t$. Since $p(\mathbf{x}) = 0 \bmod \mathcal{I}^t$, they conclude that $p(\mathbf{x}) = 0$ exactly, whereupon they recover $\mathbf{x}$ with efficient characteristic-zero root finding techniques [Len83].

Coppersmith's method works well in many settings involving *integers* – e.g., finding small solutions of univariate equations [Cop96a], factoring when the MSBs of a factor are known [Cop96b], factoring numbers of the form $p^r q$ for large r [BDHG99], etc. The main obstacle to successfully applying this method to *ideals* appears to be that the Coppersmith lattices involved have too high dimension. The Coppersmith lattice used by Cohn and Heninger has $n \times n$ blocks where one would have only a single entry in the integer case. In short, the lattice dimension is multiplied by n versus the integer case, and consequently the lattice reduction step performs much worse.

We remark that the GS algorithm, as well as our algorithm for solving the closest principal ideal generator problem (see Section 9.5), have a strategy somewhat similar to Coppersmith's method. In particular, they use Coppersmith's strategy of using lattice reduction and smallness to convert a modular equation to an exact equation, and thereafter to extract roots in characteristic zero.

9.7 Dimension Halving in Principal Ideal Lattices

Dimension Halving when a generator is provided. Gentry [Gen01] observed that, given a generator $\boldsymbol{v}$ of a principal ideal $\mathcal{I}$ in the ring $\mathbb{Z}[x]/(x^m - 1)$, one can construct a sub-lattice of $\mathcal{I}$ of dimension only $\lfloor (m+1)/2 \rfloor$ that contains a vector of length $2 \cdot \lambda_1(\mathcal{I})$. Therefore, one can hope to find a short vector in $\mathcal{I}$ by reducing a lattice that has only *half* the usual dimension. We can update this observation to obtain the following results about principal ideals in the ring of integers $\mathcal{O}_K$ of the m-th cyclotomic field K.

Lemma 9.12. *Let $\boldsymbol{B}$ be a $\mathbb{Z}$-basis of a principal ideal $\mathcal{I} = \langle \boldsymbol{v} \rangle$ over the ring of integers $\mathcal{O}_K$ of the m-th cyclotomic field K. Let $n = \phi(m)$. Let Λ be the $n/2$-dimensional sub-lattice of $\mathcal{I}$ given by $\Lambda = \{\boldsymbol{v} \cdot \boldsymbol{r} : \boldsymbol{r} \in \mathcal{O}_{K^+}\}$, where $\mathcal{O}_{K^+}$ is the ring of integers of the index-2 real subfield $K^+ = \mathbb{Q}(\zeta_m + \zeta_m^{-1})$ of K. Then, $\lambda_1(\Lambda) \leq 2\lambda_1(\mathcal{I})$.*

Proof. Let $\mathbf{z} \in \mathcal{I}$ be such that $\|\mathbf{z}\|_2^{can} = \lambda_1(\mathcal{I})$ (in the canonical embedding). Since $\mathcal{I}$ is principal, $\mathbf{z} = \boldsymbol{v} \cdot \boldsymbol{a}$ for some $\boldsymbol{a} \in \mathcal{O}_K$. Let $\mathbf{z}' = \boldsymbol{v} \cdot \overline{\boldsymbol{a}}$, where $\overline{\boldsymbol{a}} = \boldsymbol{a}(x^{-1})$ is the conjugate of $\boldsymbol{a}$. Then

$$\|\mathbf{z}'\|^2 = \langle \sigma(\mathbf{z}'), \sigma(\overline{\mathbf{z}'}) \rangle = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(\mathbf{z}')\sigma_k(\overline{\mathbf{z}'}) = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(\boldsymbol{v})\sigma_k(\boldsymbol{a})\sigma_k(\overline{\boldsymbol{v}})\sigma_k(\overline{\boldsymbol{a}}) = \sum_{k \in \mathbb{Z}_m^*} \sigma_k(\mathbf{z})\sigma_k(\overline{\mathbf{z}}) = \|\mathbf{z}\|^2.$$

Thus, $\mathbf{z} + \mathbf{z}'$ is a $\mathcal{I}$-element with length at most $2\lambda_1(\mathcal{I})$, and it is contained in the sub-lattice Λ. □

Theorem 9.13. *Let $\boldsymbol{v}$ be a generator of a principal ideal $\mathcal{I}$ in the ring of integers $\mathcal{O}_K$ of the m-th cyclotomic field K. Given $\boldsymbol{v}$, we can efficiently construct a $n/2$-dimensional sub-lattice of $\mathcal{I}$ that contains some $\boldsymbol{w} \in \mathcal{I}$ of length at most $2\lambda_1(\mathcal{I})$.*

Proof. From $\boldsymbol{v}$, we can efficiently construct a lattice Λ that contains precisely all elements of the form $\boldsymbol{v} \cdot \boldsymbol{a}$ for $\boldsymbol{a} \in \mathcal{O}_{K^+}$. By Lemma 9.12, the lattice Λ has the desired properties. □

In fact, we can do slightly better. We can also consider the sub-lattice Λ^- that contains precisely all elements of the form $\boldsymbol{v} \cdot \boldsymbol{a}$ where $\boldsymbol{a}$ is in the $n/2$ dimensional lattice of elements that can be expressed as $\boldsymbol{b} - \overline{\boldsymbol{b}}$ for some $\boldsymbol{b} \in \mathcal{O}_K$. We can then show that either Λ or Λ^- has a $\mathcal{I}$-vector of length at most $\sqrt{2}\lambda_1(\mathcal{I})$.

Next, we extend this dimension-halving attack on principal ideal lattices to the setting where the attacker is *not* given a generator of the ideal (rather only a $\mathbb{Z}$-basis of the ideal).

Dimension Halving when a generator is not provided. Is approximate-SVP for principal ideal lattices *easier* than it is for general ideal lattices (over the ring of integers of the m-th cyclotomic number field)? For general ideal lattices, currently the best known algorithm for approximate-SVP involves applying a lattice reduction algorithm (e.g., LLL [LLL82] or BKZ [Sch87]) to a lattice of dimension $n = \phi(m)$. However, as we will see, the GS algorithm implies that, for principal ideal lattices, we only need to reduce lattices of dimension $n/2$. In short, the GS algorithm gives *much stronger attacks on principal ideal lattices* than we currently have on general ideal lattices (albeit still exponential time for small approximation factors).

Theorem 9.14. *Let $T(n, d, \gamma)$ denote the (worst-case) complexity of computing a γ-approximate shortest vector in the lattice $\mathcal{L}(\boldsymbol{B})$, where $\boldsymbol{B}$ is the HNF basis of an n-dimensional lattice of determinant at most d. Computing a γ-approximate shortest vector in the lattice $\mathcal{L}(\boldsymbol{B})$, where $\boldsymbol{B}$ is a HNF basis of a principal ideal lattice $\mathcal{I}$ of norm d in the ring of integers $\mathbb{Z}[x]/\Phi_m(x)$ of the m-th cyclotomic field, has worst-case complexity at most $\mathsf{poly}(m, \log d) + T(\phi(m)/2, d, \gamma/2)$.*

Proof. Let $\mathcal{I}_{\mathbf{u}} = \langle \mathbf{u} \rangle$ be the principal ideal lattice for which we want to solve approximate-SVP, presented as a $\mathbb{Z}$-basis of $\{\boldsymbol{b}_i\}_{i\in[n]}$ with $\boldsymbol{b}_i = \mathbf{u} \cdot \boldsymbol{a}_i$ and $\boldsymbol{a}_i \in \mathcal{O}_K$. Formally set $\boldsymbol{v} = \mathsf{N}_{K/\mathbb{Q}}(\mathbf{u}) \cdot (\mathbf{u}/\overline{\mathbf{u}})$ – that is $\boldsymbol{v}$ is essentially the fraction $\mathbf{u}/\overline{\mathbf{u}}$, except that we multiply by an appropriate integer to eliminate denominators and ensure $\boldsymbol{v} \in \mathcal{O}_K$. Observe that, from $\boldsymbol{B}$, we can compute both a basis of $\mathcal{I}_{\boldsymbol{v}} = \langle \boldsymbol{v} \rangle$ and also the term $\boldsymbol{v} \cdot \overline{\boldsymbol{v}} = \mathsf{N}_{K/\mathbb{Q}}(\mathbf{u})^2$. Use the GS algorithm to recover $\boldsymbol{v}$ (and hence $\mathbf{u}/\overline{\mathbf{u}}$) in polynomial time.

From $\mathbf{u}/\overline{\mathbf{u}}$ and $\boldsymbol{B}$, compute a $\mathbb{Z}$-basis $C = \{\mathbf{c}_i = \boldsymbol{b}_i(1 + \overline{\mathbf{u}}/\mathbf{u})\}_{i\in[n]}$ of the principal ideal lattice $\mathcal{I}_{\mathbf{u}+\overline{\mathbf{u}}} = \langle \mathbf{u} + \overline{\mathbf{u}} \rangle$. Observe that $\mathbf{u}+\overline{\mathbf{u}}$ is in the index-2 real subfield $K^+ = \mathbb{Q}(\zeta_m + \zeta_m^{-1})$. Project the basis C down to a $n/2$-dimensional basis C_{K^+} of the ideal $\mathcal{I}_{\mathbf{u}+\overline{\mathbf{u}},K^+} = \mathcal{I}_{\mathbf{u}+\overline{\mathbf{u}}} \cap K^+ \subset \mathcal{O}_{K^+}$. Observe that C_{K^+} is a set of the form $\{(\mathbf{u} + \overline{\mathbf{u}}) \cdot \boldsymbol{r} : \boldsymbol{r} \in \mathcal{O}_{K^+}\}$. Multiply each of the elements in C_{K^+} by $\mathbf{u}/(\mathbf{u} + \overline{\mathbf{u}})$ to get a basis $B_{K^+} = \{\mathbf{u} \cdot \boldsymbol{r} : \boldsymbol{r} \in \mathcal{O}_{K^+}\}$ of the lattice $\Lambda = \mathcal{L}(B_{K^+})$.

By Lemma 9.13, Λ has a nonzero vector of length at most $2\lambda_1(\mathcal{I})$. Therefore, we can solve γ-approximate-SVP in $\mathcal{I}$ by solving $\gamma/2$-approximate-SVP in Λ, proving the theorem. □

Note that non-principal ideal lattices, which in general can be expressed in terms of *two* generators, do not appear to be vulnerable to this dimension-halving attack.

The params in our constructions implicitly reveal principal ideal lattices – e.g., the lattice $\langle \mathbf{h} \cdot \boldsymbol{g}^{\kappa-1} \rangle$ will likely be generated as an $\mathcal{O}_K$-linear combination of the terms of the form $\mathbf{h} \cdot \boldsymbol{b}_i^{\kappa}/\boldsymbol{g}$ that can be computed from params as explained in Section 7.3.1. Therefore, we recommend using $\mathcal{O}_K$ of degree twice what one would normally use for general ideal lattices.

Previous schemes have also used, or raised the possibility of using, principal ideals, including fully homomorphic encryption schemes [Gen09b, SV10, GH11], homomorphic signatures schemes [BF11a], and key agreement schemes [Buc91]. Use of cyclotomics with higher degrees is also recommended in these settings.

CHAPTER 10

One-Round Key-Exchange

Diffie and Hellman in their seminal paper [DH76] provided the first construction of a one-round two-party key-exchange protocol and laid the foundations for the work on public key cryptography. Joux [Jou00] constructed the first one-round three-party key-exchange protocol using Weil and Tate pairings. Boneh and Silverberg [BS03] showed how this result could be extended to get a one-round N-party key-exchange protocol if multilinear maps existed. Our encoding schemes easily support the Boneh-Silverberg construction, with one subtle difference: Since our public parameters hide some secrets (i.e., the elements $\mathbf{g}, \boldsymbol{h}, \mathbf{z}$) therefore our construction of one-round N-party secret key exchange protocol is in the common reference string model.

10.1 Definitions

Consider a setting with N parties who wish to set up a shared key using a one-round protocol. The "one-round" refers to the setting in which each party is only allowed to broadcast one value to all other parties. Furthermore all N broadcasts occur simultaneously. Once all the N parties broadcast their values, each party should be able to locally compute a global shared secret s. Using the notation from [BS03], a one-round N-party key-exchange scheme consists of the following three randomized PPT algorithms:

- $\mathsf{Setup}(\lambda, N)$: Takes a security parameter $\lambda \in \mathbb{Z}^+$ and the number of participants N as input. It runs in time polynomial in λ, N and outputs public parameters params.
- $\mathsf{Publish}(\mathsf{params}, i)$: Given an input $i \in \{1, \ldots, N\}$, the algorithm outputs a pair $(pub_i, priv_i)$, with both in $\{0,1\}^*$. Every party i execute this algorithm with its input i and broadcasts the generated value pub_i to all other parties, and keeping $priv_i$ secret.

- $\mathsf{KeyGen}(\mathsf{params}, j, priv_j, \{pub_i\}_{i\neq j})$: Party $j \in \{1, \ldots N\}$ collects the public broadcasts sent by all other parties and executes KeyGen on all these public values and its secret value $priv_j$. On this execution the algorithm KeyGen outputs a key s_j.

The *consistency* requirement for the above scheme is that all N parties generate the same shared key with high probability. The scheme is said to be secure if no polynomial time algorithm, given all N public values $(pub_1, \ldots pub_N)$, can distinguish the true shared key s from random.

10.2 Our Construction.

We present a one-round N-party key-exchange protocol using an encoding schemes with $\kappa = N - 1$, under the GDDH assumption. The construction is a straightforward adaptation of [BS03]:

$\mathsf{Setup}(1^\lambda, 1^N)$**.** We just run the $\mathsf{InstGen}$ algorithm of the underlying encoding scheme, getting $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, 1^{N-1})$, and outputting $(\mathsf{params}, \mathbf{p}_{zt})$ as the public parameter. Note that $\mathbf{p}_{zt}$ is a level-$N-1$ zero-test parameter. Let q, n, σ be the corresponding parameters of the encoding scheme. Note also that in this construction we insist that the order of the quotient ring $R/\mathcal{I}$ be a large prime (or at least that it does not have any small divisors).

$\mathsf{Publish}(\mathsf{params}, \mathbf{p}_{zt}, i)$**.** Each party i chooses a random level-zero encoding $\boldsymbol{d} \leftarrow \mathsf{samp}(\mathsf{params})$ as a secret key, and publishes the corresponding level-one public key $\boldsymbol{w}_i \leftarrow \mathsf{enc}(\mathsf{params}, 1, \boldsymbol{d})$.

$\mathsf{KeyGen}(\mathsf{params}, \mathbf{p}_{zt}, j, \boldsymbol{d}_j, \{\boldsymbol{w}_i\}_{i\neq j})$**.** Each party j multiplies its secret key $\boldsymbol{d}_j$ by the public keys of all its peers, $\boldsymbol{v}_j \leftarrow \boldsymbol{d}_j \cdot \prod_{i\neq j} \boldsymbol{w}_i$, thus getting a level-$N-1$ encoding of the product coset $\prod_i \boldsymbol{d}_i + \mathcal{I}$. Then the party uses the extraction routine to compute the key, $s_j \leftarrow \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}_j)$. (Recall that in out case extraction consists of multiplying by the zero-test parameter and outputting the high-order bits.)

The consistency requirement follows directly from the agreement property of the extraction procedure in the underlying encoding scheme: Notice that all the parties get valid encodings of the same uniformly-chosen coset, hence the extraction property implies that they should extract the same key with high probability.

Similarly, security follows directly from a combination of the GDDH assumption and the randomness property of the extraction property of the extraction procedure in the underlying encoding scheme.

Theorem 10.1. *The protocol described above is a one-round N-party Key Exchange protocol if the GDDH assumption holds for the underlying encoding scheme.*

Proof. We need to show that an attacker that sees all the public keys cannot distinguish the output of the first party (say) from a uniformly random string. By GDDH, the adversary cannot distinguish between the level-$(N-1)$ encoding $\boldsymbol{v}_1 \leftarrow \boldsymbol{d}_1 \cdot \prod_{i>1} \boldsymbol{w}_i$ that Party 1 computes and an element $\boldsymbol{v}'_1 \leftarrow \boldsymbol{d}'_1 \cdot \prod_{i>1} \boldsymbol{w}_i$ that is obtained for a random and independent $\boldsymbol{d}'_1 \leftarrow \mathsf{samp}(\mathsf{params})$ (which is a level-$N-1$ encoding of the coset $(\boldsymbol{d}'_1 \cdot \prod_{i>1} \boldsymbol{d}_i) + \mathcal{I}$).

By the randomness property of the sampling procedure, $\boldsymbol{d}'_1$ is nearly uniformly distributed among the cosets of $\mathcal{I}$. Since $|R/\mathcal{I}|$ is a large prime then with high probability $\prod_{i>1} \boldsymbol{d}_i \not\equiv 0 \pmod{\mathcal{I}}$, and thus $\boldsymbol{d}'_1 \cdot \prod_{i>1} \boldsymbol{d}_i$ is also nearly uniformly distributed among the cosets of $\mathcal{I}$. We can now use the randomness property of the extraction function to conclude that $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}'_1)$ is a nearly uniform string, completing the proof. □

APPENDIX A

Generalizing Graded Encoding Systems

Here we generalize the definitions of graded encodings schemes from Section 3.2 to deal with the "asymmetric case," where there are many different "level-one sets" (corresponding to the many different source groups). We view the different level-one sets as separate dimensions, and correspondingly replace the index i from the symmetric case by an index-vector $\boldsymbol{v} \in \mathbb{N}^\tau$ (with $\mathbb{N}$ the natural numbers and τ the equivalent of the number of different groups). The different level-one sets correspond to the standard (τ-dimensional) unit vectors $\boldsymbol{e}_i$, and an encoding of $\alpha \in R$ relative to the index $\boldsymbol{e}_i$ (i.e., an element $a \in S_{\boldsymbol{e}_i}^{(\alpha)}$) is playing a role analogous to $\alpha \cdot g_i$ in asymmetric multilinear maps.

Note that in our case we can have τ "different groups" and yet we can multiply up to some number κ of different encodings, potentially $\kappa \neq \tau$. Hence we can also get a mix of the symmetric and asymmetric cases. If $u_1, \ldots, u_\kappa$ are encodings of $\alpha_1, \ldots, \alpha_\kappa \in R$ relative to indexes $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_\kappa \in \mathbb{N}^\tau$, respectively, then $u^* = u_1 \times \cdots \times u_\kappa$ is an encoding of the product $\alpha^* = \prod_i \alpha_i \in R$ relative to the sum of the indexes $\boldsymbol{v} = \sum_i \boldsymbol{v}_i \in \mathbb{N}^\tau$.

For this general setting, we replace the parameter κ by a set $\varkappa \subset \mathbb{N}^\tau$ which specifies the subset of indexes where we can test for zero. Additionally the set of levels $\mathsf{Below}(\varkappa) \subset \mathbb{N}^\tau$ includes the indexes for which we can get valid encodings, and of course, we preclude encoding "above the zero-testing levels," since for those levels we cannot check equality of encodings. Hence the zero-test indexes implicitly define also the subset $\mathsf{Below}(\varkappa)$. We begin by formalizing the notions of "above" and "below" for our indexes, which is defined entry-wise.

Definition A.1 (Partial order on $\mathbb{N}^\tau$). *For an integer $\tau > 0$ and two vector $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{N}^\tau$, we define*

$$\boldsymbol{v} \leq \boldsymbol{w} \;\Leftrightarrow\; \boldsymbol{v}[j] \leq \boldsymbol{w}[j] \text{ for all } j = 1, 2, \ldots, \tau.$$

As usual, we have $\boldsymbol{v} < \boldsymbol{w}$ if $\boldsymbol{v} \leq \boldsymbol{w}$ and $\boldsymbol{v} \neq \boldsymbol{w}$.

Definition A.2 (Below $\varkappa$)**.** *For an arbitrary subset of indexes $\varkappa \subset \mathbb{N}^\tau$ we denote the set of indexes "below $\varkappa$" as:*

$$\mathsf{Below}(\varkappa) \stackrel{\text{def}}{=} \{\boldsymbol{v} \in \mathbb{N}^\tau : \exists \boldsymbol{w} \in \varkappa \text{ s.t. } \boldsymbol{v} \le \boldsymbol{w}\}.$$

We can now extend Definition 3.2 to the asymmetric case by defining $\varkappa$-graded encoding systems, where we think of $\varkappa$ as the subset of indexes that admit zero-testing.

Definition A.3 ($\varkappa$-Graded Encoding System)**.** *Let $\varkappa \subset \mathbb{N}^\tau$ be a finite set (for some integer $\tau > 0$), and let R be a ring. A $\varkappa$-Graded Encoding System for R is a system of sets $\mathcal{S} = \{S_{\boldsymbol{v}}^{(\alpha)} \subset \{0,1\}^* : \boldsymbol{v} \in \mathsf{Below}(\varkappa), \alpha \in R\}$, with the following properties:*

1. *For every fixed index $\boldsymbol{v} \in \mathsf{Below}(\varkappa)$, the sets $\{S_{\boldsymbol{v}}^{(\alpha)} : \alpha \in R\}$ are disjoint (hence they form a partition of $S_{\boldsymbol{v}} \stackrel{\text{def}}{=} \bigcup_\alpha S_{\boldsymbol{v}}^{(\alpha)}$).*

2. *There are binary operations '$+$' and '$-$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every $\boldsymbol{v} \in \mathsf{Below}(\varkappa)$, and every $u_1 \in S_{\boldsymbol{v}}^{(\alpha_1)}$ and $u_2 \in S_{\boldsymbol{v}}^{(\alpha_2)}$, it holds that*

$$u_1 + u_2 \in S_{\boldsymbol{v}}^{(\alpha_1+\alpha_2)} \quad \text{and} \quad u_1 - u_2 \in S_{\boldsymbol{v}}^{(\alpha_1-\alpha_2)} \tag{A.1}$$

   where $\alpha_1 + \alpha_2$ and $\alpha_1 - \alpha_2$ are addition and subtraction in R.

3. *There is an associative binary operation '$\times$' (on $\{0,1\}^*$) such that for every $\alpha_1, \alpha_2 \in R$, every $\boldsymbol{v}_1, \boldsymbol{v}_2$ with $\boldsymbol{v}_1 + \boldsymbol{v}_2 \in \mathsf{Below}(\varkappa)$, and every $u_1 \in S_{\boldsymbol{v}_1}^{(\alpha_1)}$ and $u_2 \in S_{\boldsymbol{v}_2}^{(\alpha_2)}$, it holds that*

$$u_1 \times u_2 \in S_{\boldsymbol{v}_1+\boldsymbol{v}_2}^{(\alpha_1\cdot\alpha_2)}. \tag{A.2}$$

   Here $\alpha_1 \cdot \alpha_2$ is multiplication in R, and $\boldsymbol{v}_1 + \boldsymbol{v}_2$ is vector addition in $\mathbb{N}^\tau$.

Clearly, Definition A.3 implies that if we have a collection of n encodings $u_i \in S_{\boldsymbol{v}_i}^{(\alpha_i)}$, $i = 1, 2 \ldots, n$, then as long as $\sum_i \boldsymbol{v}_i \in \mathsf{Below}(\varkappa)$ we get $u_1 \times \cdots \times u_n \in S_{\sum_i \boldsymbol{v}_i}^{(\prod_i \alpha_i)}$. We note that symmetric κ-multilinear maps as per Definition 3.2 correspond to $\{\kappa\}$-graded encoding systems (with $\tau = 1$), the asymmetric <u>bi</u>linear case corresponds to $\{(1,1)\}$-graded systems (with $\tau = 2$), etc.

A.1 Efficient Procedures, the Dream Version

As before, we first describe a "dream version" of the efficient procedures and then explain how to modify them to deal with technicalities that arise from our use of lattices in the realization.

Instance Generation. The randomized $\mathsf{InstGen}(1^\lambda, \tau, \varkappa)$ takes as inputs the parameters λ, τ the subset $\varkappa \subset \mathbb{N}^\tau$. It outputs $(\mathsf{params}, \mathbf{p}_{zt})$, where params is a description of a $\varkappa$-Graded Encoding System as above, and $\mathbf{p}_{zt}$ is a set of zero-test parameters for the indexes in $\varkappa$.

Ring Sampler. The randomized $\mathsf{samp}(\mathsf{params})$ outputs a "level-zero encoding" $a \in S_{\mathbf{0}}^{(\alpha)}$ for a nearly uniform element $\alpha \in_R R$. (Note that we require that the "plaintext" $\alpha \in R$ is nearly uniform, but not that the encoding a is uniform in $S_{\mathbf{0}}^{(\alpha)}$.)

Encoding. The (possibly randomized) $\mathsf{enc}(\mathsf{params}, \boldsymbol{v}, a)$ takes a "level-zero" encoding $a \in S_{\mathbf{0}}^{(\alpha)}$ for some $\alpha \in R$ and index $\boldsymbol{v} \in \mathsf{Below}(\varkappa)$, and outputs the "level-$\boldsymbol{v}$" encoding $u \in S_{\boldsymbol{v}}^{(\alpha)}$ for the same α.

Addition and negation. Given params and two encodings relative to the same index, $u_1 \in S_{\boldsymbol{v}}^{(\alpha_1)}$ and $u_2 \in S_{\boldsymbol{v}}^{(\alpha_2)}$, we have $\mathsf{add}(\mathsf{params}, i, u_1, u_2) = u_1 + u_2 \in S_{\boldsymbol{v}}^{(\alpha_1+\alpha_2)}$, and $\mathsf{sub}(\mathsf{params}, i, u_1, u_2) = u_1 + u_2 \in S_{\boldsymbol{v}}^{(\alpha_1+\alpha_2)}$,

Multiplication. For $u_1 \in S_{\boldsymbol{v}_1}^{(\alpha_1)}$, $u_2 \in S_{\boldsymbol{v}_2}^{(\alpha_2)}$ with $\boldsymbol{v}_1 + \boldsymbol{v}_2 \in \mathsf{Below}(\varkappa)$, we have $\mathsf{mul}(\mathsf{params}, \boldsymbol{v}_1, u_1, \boldsymbol{v}_2, u_2) = u_1 \times u_2 \in S_{\boldsymbol{v}_1+\boldsymbol{v}_2}^{(\alpha_1 \cdot \alpha_2)}$.

Zero-test. The procedure $\mathsf{isZero}(\mathsf{params}, \boldsymbol{v}, u)$ output 1 if $\boldsymbol{v} \in \varkappa$ and $u \in S_{\boldsymbol{v}}^{(0)}$ and 0 otherwise. Note that in conjunction with the subtraction procedure, this lets us test if $u_1, u_2 \in S_{\boldsymbol{v}}$ encode the same element $\alpha \in R$.

Extraction. This procedure extracts a "canonical" and "random" representation of ring elements from their level-$\boldsymbol{v}$ encoding. Namely $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, u)$ outputs (say) $s \in \{0,1\}^\lambda$, such that:

(a) For any $\alpha \in R$, $\boldsymbol{v} \in \varkappa$ and two $u_1, u_2 \in S_{\boldsymbol{v}}^{(\alpha)}$, $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u_1) = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u_2)$,
(b) For any $\boldsymbol{v} \in \varkappa$, the distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u) : \alpha \in_R R, u \in S_{\boldsymbol{v}}^{(\alpha)}\}$ is nearly uniform over $\{0,1\}^\lambda$.

A.2 Efficient Procedures, the Real-Life Version

As before, our real-life procedures have noise bounds and we are only ensured of their properties when the bounds are valid and small enough. Also as before, we relax the requirements on the zero-test and the extraction routines, as we now describe.

Zero-test. We sometime allow false positives for this procedure, but not false negatives. Namely, $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u) = 1$ for every $\boldsymbol{v} \in \varkappa$ and $u \in S_{\boldsymbol{v}}^{(0)}$, but we may have $\mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u) = 1$ also in other cases. Again our weakest functionality requirement that we make is that for a uniform random choice of $\alpha \in_R R$, we have for every $\boldsymbol{v} \in \varkappa$

$$\Pr_{\alpha \in_R R}\left[\exists\, u \in S_{\boldsymbol{v}}^{(\alpha)} \text{ s.t } \mathsf{isZero}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u) = 1\right] = \text{negligible}(\lambda). \tag{A.3}$$

Additional requirements are considered security features (that a scheme may or may not possess), and are discussed later in this section.

Extraction. We replace[1] properties (a)-(b) from the dream-version above by the weaker requirements:

(a$'$) For a randomly chosen $a \leftarrow \mathsf{samp}(\mathsf{params})$ and every $\boldsymbol{v} \in \varkappa$, if we run the encoding algorithm twice to encode a at level $\boldsymbol{v}$ and then extract from both copies then we get:

$$\Pr\left[\begin{array}{l} \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u_1) \\ = \mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u_2) \end{array} : \begin{array}{l} a \leftarrow \mathsf{samp}(\mathsf{params}) \\ u_1 \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{v}, a) \\ u_2 \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{v}, a) \end{array}\right] \geq 1 - \text{negligible}(\lambda).$$

(b$'$) The distribution $\{\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u) : a \leftarrow \mathsf{samp}(\mathsf{params}), u \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{v}, a)\}$ is nearly uniform over $\{0,1\}^\lambda$.

We typically need these two conditions to hold even if the noise bound that the encoding routine takes as input is larger than the one output by samp (upto some maximum value).

A.3 Hardness Assumptions

The MDDH analog for this case says that it is hard to recognize encoding of products, except relative to indexes in $\mathsf{Below}(\varkappa)$. One way to formalize it is by letting the adversary choose the level "above $\varkappa$" on which it wants to be tested. This is formalized by the following process. (Below we suppress the noise bounds for readability):

```
1. $(\mathsf{params}, \mathbf{p}_{zt}) \leftarrow \mathsf{InstGen}(1^\lambda, \tau, \varkappa)$
2. $\boldsymbol{v}, \boldsymbol{v}^* \leftarrow \mathcal{A}(\mathsf{params}, \mathbf{p}_{zt})$          // $\boldsymbol{v} \in \varkappa$ and $\boldsymbol{v}^* \notin \mathsf{Below}(\varkappa)$
3. For $i = 1, \ldots, \tau$, for $j = 1, \ldots v_i^*$:     // $v_i^*$ denotes the $i^{th}$ component of $\boldsymbol{v}_i^*$
4.     Choose $a_{i,j} \leftarrow \mathsf{samp}(\mathsf{params})$      // level-0 encoding of random $\alpha_{i,j} \in_R R$
5.     Set $u_{i,j} \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{e}_i, a_{i,j})$   // encoding of $\alpha_{i,j}$ w.r.t the $i$'th unit vector
6. Set $\tilde{a} = \prod_{i,j} a_{i,j}$                // level-0 encoding of the product
7. Choose $\hat{a} \leftarrow \mathsf{samp}(\mathsf{params})$          // level-0 encoding of a random element
8. Set $\tilde{u} \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{v}, \tilde{a})$          // level-$\boldsymbol{v}$ encoding of the product
9. Set $\hat{u} \leftarrow \mathsf{enc}(\mathsf{params}, \boldsymbol{v}, \hat{a})$          // level-$\boldsymbol{v}$ encoding of random
```

The adversary $\mathcal{A}$ then gets all the $u_{i,j}$'s and either $\tilde{u}$ or $\hat{u}$, and it needs to guess which is the case. It is considered successful if the guess is correct and in addition $\boldsymbol{v} \in \varkappa$ and $\boldsymbol{v} \lneq \boldsymbol{v}*$. The generalized GDDH says that for any setting of the parameters, the following two distributions, defined over the experiment above, are computationally indistinguishable:

$$\mathcal{D}_{GenGDDH} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \tilde{u})\} \text{ and } \mathcal{D}_{GenRAND} = \{(\mathsf{params}, \mathbf{p}_{zt}, \{u_i\}_i, \hat{u})\}.$$

Zero-test security. Zero-testing security is defined exactly as in the symmetric case, except that we require it to work relative to all the indexes $\boldsymbol{v} \in \varkappa$.

[1] Our construction from Section 6 does not support full canonicalization. Instead, we settle for $\mathsf{ext}(\mathsf{params}, \mathbf{p}_{zt}, \boldsymbol{v}, u)$ that has a good chance of producing the same output when applied to different encoding of the same elements.

References

[AGHS12] Shweta Agrawal, Craig Gentry, Shai Halevi, and Amit Sahai. Sampling discrete gaussians efficiently and obliviously. Cryptology ePrint Archive, Report 2012/714, 2012. http://eprint.iacr.org/. 19, 20, 21, 22, 67

[AJLA$^+$12] Gilad Asharov, Abhishek Jain, Adriana López-Alt, Eran Tromer, Vinod Vaikuntanathan, and Daniel Wichs. Multiparty computation with low communication, computation and interaction via threshold FHE. In David Pointcheval and Thomas Johansson, editors, *Advances in Cryptology – EUROCRYPT 2012*, volume 7237 of *Lecture Notes in Computer Science*, pages 483–501, Cambridge, UK, April 15–19, 2012. Springer, Berlin, Germany. 8

[AR05] Dorit Aharonov and Oded Regev. Lattice problems in np cap conp. *J. ACM*, 52(5):749–765, 2005. 19

[BC10] Nir Bitansky and Ran Canetti. On strong simulation and composable point obfuscation. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 520–537, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. 7

[BDHG99] Dan Boneh, Glenn Durfee, and Nick Howgrave-Graham. Factoring $N = p^r q$ for large r. In Michael J. Wiener, editor, *Advances in Cryptology – CRYPTO'99*, volume 1666 of *Lecture Notes in Computer Science*, pages 326–337, Santa Barbara, CA, USA, August 15–19, 1999. Springer, Berlin, Germany. 71

[Bei11] Amos Beimel. Secret-sharing schemes: A survey. In Yeow Meng Chee, Zhenbo Guo, San Ling, Fengjing Shao, Yuansheng Tang, Huaxiong Wang, and Chaoping Xing, editors, *Coding and Cryptology - Third International Workshop, IWCC 2011*, volume 6639 of *Lecture Notes in Computer Science*, pages 11–46, Qingdao, China, May 30-June 3 2011. Springer. 6

[BF01] Dan Boneh and Matthew K. Franklin. Identity-based encryption from the Weil pairing. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 213–229, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. 1, 9

[BF11a] Dan Boneh and David Mandell Freeman. Homomorphic signatures for polynomial functions. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 149–168, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany. 30, 73

[BF11b] Dan Boneh and David Mandell Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In Dario Catalano, Nelly Fazio, Rosario Gennaro, and Antonio Nicolosi, editors, *PKC 2011: 14th International Workshop on Theory and Practice in Public Key Cryptography*, volume 6571 of *Lecture Notes in Computer Science*, pages 1–16, Taormina, Italy, March 6–9, 2011. Springer, Berlin, Germany. 20

[BGI+01] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *Advances in Cryptology – CRYPTO 2001*, volume 2139 of *Lecture Notes in Computer Science*, pages 1–18, Santa Barbara, CA, USA, August 19–23, 2001. Springer, Berlin, Germany. 5, 7, 8

[BGI+12] Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. *J. ACM*, 59(2):6, 2012. 5, 7

[BGK+13] Boaz Barak, Sanjam Garg, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Protecting obfuscation against algebraic attacks. Manuscript, 2013. 8

[BL96] Dan Boneh and Richard J. Lipton. Algorithms for black-box fields and their application to cryptography (extended abstract). In Neal Koblitz, editor, *Advances in Cryptology – CRYPTO'96*, volume 1109 of *Lecture Notes in Computer Science*, pages 283–297, Santa Barbara, CA, USA, August 18–22, 1996. Springer, Berlin, Germany. 2, 43

[BLS04] Dan Boneh, Ben Lynn, and Hovav Shacham. Short signatures from the Weil pairing. *Journal of Cryptology*, 17(4):297–319, September 2004. 9

[BR93] Mihir Bellare and Phillip Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In V. Ashby, editor, *ACM CCS 93: 1st Conference on Computer and Communications Security*, pages 62–73, Fairfax, Virginia, USA, November 3–5, 1993. ACM Press. 9

[BR96] Mihir Bellare and Phillip Rogaway. The exact security of digital signatures: How to sign with RSA and Rabin. In Ueli M. Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer*

Science, pages 399–416, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany. 9

[BR13a] Zvika Brakerski and Guy N. Rothblum. Black-box obfuscation for d-cnfs. Cryptology ePrint Archive, Report 2013/557, 2013. http://eprint.iacr.org/. 8

[BR13b] Zvika Brakerski and Guy N. Rothblum. Obfuscating conjunctions. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part II*, volume 8043 of *Lecture Notes in Computer Science*, pages 416–434. Springer, 2013. 7

[BR13c] Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. Cryptology ePrint Archive, Report 2013/563, 2013. http://eprint.iacr.org/. 8

[BRS03] John Black, Phillip Rogaway, and Thomas Shrimpton. Encryption-scheme security in the presence of key-dependent messages. In Kaisa Nyberg and Howard M. Heys, editors, *SAC 2002: 9th Annual International Workshop on Selected Areas in Cryptography*, volume 2595 of *Lecture Notes in Computer Science*, pages 62–75, St. John's, Newfoundland, Canada, August 15–16, 2003. Springer, Berlin, Germany. 2

[BS96] Eric Bach and Jeffrey Shallit. *Algorithmic Number Theory, Volume I: Efficient Algorithms*. MIT Press, 1996. 29, 68

[BS03] Dan Boneh and Alice Silverberg. Applications of multilinear forms to cryptography. *Contemporary Mathematics*, 324:71–90, 2003. 1, 2, 5, 10, 74, 75

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: definitions and challenges. In *TCC*, pages 253–273, 2011. 6

[Buc91] Johannes Buchmann. Number theoretic algorithms and cryptology. In Lothar Budach, editor, *FCT*, volume 529 of *Lecture Notes in Computer Science*, pages 16–21. Springer, 1991. 73

[BW13] Dan Boneh and Brent Waters. Constrained pseudorandom functions and their applications. Cryptology ePrint Archive, Report 2013/352, 2013. http://eprint.iacr.org/. 8

[Can97] Ran Canetti. Towards realizing random oracles: Hash functions that hide all partial information. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 455–469, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. 7

[CCV12] Nishanth Chandran, Melissa Chase, and Vinod Vaikuntanathan. Functional re-encryption and collusion-resistant obfuscation. In Ronald Cramer, editor, *TCC 2012: 9th Theory of Cryptography Conference*, volume 7194 of *Lecture Notes in Computer Science*, pages 404–421, Taormina, Sicily, Italy, March 19–21, 2012. Springer, Berlin, Germany. 7

[CD08] Ran Canetti and Ronny Ramzi Dakdouk. Obfuscating point functions with multibit output. In Nigel P. Smart, editor, *Advances in Cryptology – EUROCRYPT 2008*, volume 4965 of *Lecture Notes in Computer Science*, pages 489–508, Istanbul, Turkey, April 13–17, 2008. Springer, Berlin, Germany. 7

[CDNO97] Ran Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 90–104, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. 8

[CGH98] Ran Canetti, Oded Goldreich, and Shai Halevi. The random oracle methodology, revisited (preliminary version). In *30th Annual ACM Symposium on Theory of Computing*, pages 209–218, Dallas, Texas, USA, May 23–26, 1998. ACM Press. 9

[CH11] Henry Cohn and Nadia Heninger. Ideal forms of coppersmith's theorem and guruswami-sudan list decoding. In Bernard Chazelle, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 7-9, 2011. Proceedings*, pages 298–308. Tsinghua University Press, 2011. 71

[CIJ$^+$13] Angelo De Caro, Vincenzo Iovino, Abhishek Jain, Adam O'Neill, Omer Paneth, and Giuseppe Persiano. On the achievability of simulation-based security for functional encryption. In *CRYPTO*, 2013. 7

[CL01] Jan Camenisch and Anna Lysyanskaya. An efficient system for non-transferable anonymous credentials with optional anonymity revocation. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 93–118, Innsbruck, Austria, May 6–10, 2001. Springer, Berlin, Germany. 2

[CLT13] Jean-Sebastien Coron, Tancrede Lepoint, and Mehdi Tibouchi. Practical multilinear maps over the integers. Cryptology ePrint Archive, Report 2013/183, 2013. http://eprint.iacr.org/. 4

[CMR98] Ran Canetti, Daniele Micciancio, and Omer Reingold. Perfectly one-way probabilistic hash functions (preliminary version). In *30th Annual ACM Symposium on Theory of Computing*, pages 131–140, Dallas, Texas, USA, May 23–26, 1998. ACM Press. 7

[Cop96a] Don Coppersmith. Finding a small root of a bivariate integer equation; factoring with high bits known. In Ueli M. Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 178–189, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany. 58, 71

[Cop96b] Don Coppersmith. Finding a small root of a univariate modular equation. In Ueli M. Maurer, editor, *Advances in Cryptology – EUROCRYPT'96*, volume 1070 of *Lecture Notes in Computer Science*, pages 155–165, Saragossa, Spain, May 12–16, 1996. Springer, Berlin, Germany. 58, 71

[CRV10] Ran Canetti, Guy N. Rothblum, and Mayank Varia. Obfuscation of hyperplane membership. In Daniele Micciancio, editor, *TCC 2010: 7th Theory of Cryptography Conference*, volume 5978 of *Lecture Notes in Computer Science*, pages 72–89, Zurich, Switzerland, February 9–11, 2010. Springer, Berlin, Germany. 7

[CS97] Don Coppersmith and Adi Shamir. Lattice attacks on NTRU. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 52–61, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Germany. 3

[DH76] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, 22(6):644–654, 1976. 1, 5, 6, 74

[DN12a] L. Ducas and P. Q. Nguyen. Faster gaussian lattice sampling using lazy floating-point arithmetic. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 415–432, Beijing, China, December 2-6 2012. Springer, Berlin, Germany. 50

[DN12b] L. Ducas and P. Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In Xiaoyun Wang and Kazue Sako, editors, *Advances in Cryptology - ASIACRYPT 2012*, volume 7658 of *Lecture Notes in Computer Science*, pages 433–450, Beijing, China, December 2-6 2012. Springer, Berlin, Germany. 3, 57, 66

[DPSZ11] I. Damgard, V. Pastro, N.P. Smart, and S. Zakarias. Multiparty computation from somewhat homomorphic encryption. Cryptology ePrint Archive, Report 2011/535, 2011. `http://eprint.iacr.org/`. 33

[FHPS13] Eduarda S. V. Freire, Dennis Hofheinz, Kenneth G. Paterson, and Christoph Striecks. Programmable hash functions in the multilinear setting. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology - CRYPTO 2013 - 33rd Annual Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2013. Proceedings, Part I*, volume 8042 of *Lecture Notes in Computer Science*, pages 513–530. Springer, 2013. 8, 9

[Gen01] Craig Gentry. Key recovery and message attacks on NTRU-composite. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 182–194, Innsbruck, Austria, May 6–10, 2001. Springer, Berlin, Germany. 3, 71

[Gen09a] Craig Gentry. *A fully homomorphic encryption scheme*. PhD thesis, Stanford University, 2009. crypto.stanford.edu/craig. 2, 3, 26

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In Michael Mitzenmacher, editor, *41st Annual ACM Symposium on Theory of Computing*, pages 169–178, Bethesda, Maryland, USA, May 31 – June 2, 2009. ACM Press. 73

[Gen10] Craig Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 116–137, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. 56

[GGH97] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Eliminating decryption errors in the Ajtai-Dwork cryptosystem. In Burton S. Kaliski Jr., editor, *Advances in Cryptology – CRYPTO'97*, volume 1294 of *Lecture Notes in Computer Science*, pages 105–111, Santa Barbara, CA, USA, August 17–21, 1997. Springer, Berlin, Germany. 3, 65

[GGH12] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. Cryptology ePrint Archive, Report 2012/610, 2012. http://eprint.iacr.org/. 2

[GGH13a] Sanjam Garg, Craig Gentry, and Shai Halevi. Candidate multilinear maps from ideal lattices. In Thomas Johansson and Phong Q. Nguyen, editors, *Advances in Cryptology – EUROCRYPT 2013*, volume 7881 of *Lecture Notes in Computer Science*, pages 1–17, Athens, Greece, May 26–30 2013. Springer, Berlin, Germany. 2

[GGH$^+$13b] Sanjam Garg, Craig Gentry, Shai Halevi, Mariana Raykova, Amit Sahai, and Brent Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. Cryptology ePrint Archive, Report 2013/451, 2013. http://eprint.iacr.org/. 2, 5, 7, 53

[GGH$^+$13c] Sanjam Garg, Craig Gentry, Shai Halevi, Amit Sahai, and Brent Waters. Attribute-based encryption for circuits from multilinear maps. Cryptology ePrint Archive, Report 2013/128, 2013. http://eprint.iacr.org/. 6, 39

[GGHR13] Sanjam Garg, Craig Gentry, Shai Halevi, and Mariana Raykova. Two-round secure mpc from indistinguishability obfuscation. Cryptology ePrint Archive, Report 2013/601, 2013. http://eprint.iacr.org/. 7, 8

[GGSW13] Sanjam Garg, Craig Gentry, Amit Sahai, and Brent Waters. Witness encryption and its applications. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 467–476. ACM, 2013. 6

[GH10] Craig Gentry and Shai Halevi. Implementing gentry's fully-homomorphic encryption scheme. Cryptology ePrint Archive, Report 2010/520, 2010. http://eprint.iacr.org/. 26

[GH11] Craig Gentry and Shai Halevi. Implementing Gentry's fully-homomorphic encryption scheme. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 129–148, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany. 73

[GKP$^+$12] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Reusable garbled circuits and succinct functional encryption. Cryptology ePrint Archive, Report 2012/733, 2012. http://eprint.iacr.org/. 7

[GKP$^+$13] Shafi Goldwasser, Yael Kalai, Raluca Ada Popa, Vinod Vaikuntanathan, and Nickolai Zeldovich. Overcoming the worst-case curse for cryptographic constructions. Cryptology ePrint Archive, Report 2013/229, 2013. http://eprint.iacr.org/. 6, 7

[GOS06] Jens Groth, Rafail Ostrovsky, and Amit Sahai. Perfect non-interactive zero knowledge for NP. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 339–358, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany. 1

[GPSW06] Vipul Goyal, Omkant Pandey, Amit Sahai, and Brent Waters. Attribute-based encryption for fine-grained access control of encrypted data. In Ari Juels, Rebecca N. Wright, and Sabrina De Capitani di Vimercati, editors, *ACM CCS 06: 13th Conference on Computer and Communications Security*, pages 89–98, Alexandria, Virginia, USA, October 30 – November 3, 2006. ACM Press. Available as Cryptology ePrint Archive Report 2006/309. 6

[GPV08] Craig Gentry, Chris Peikert, and Vinod Vaikuntanathan. Trapdoors for hard lattices and new cryptographic constructions. In Richard E. Ladner and Cynthia Dwork, editors, *40th Annual ACM Symposium on Theory of Computing*, pages 197–206, Victoria, British Columbia, Canada, May 17–20, 2008. ACM Press. 19, 20, 21, 50, 51, 58, 65, 67

[GR07] Shafi Goldwasser and Guy N. Rothblum. On best-possible obfuscation. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 194–213, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany. 7

[GS02] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In Lars R. Knudsen, editor, *Advances in Cryptology – EUROCRYPT 2002*, volume 2332 of *Lecture Notes in Computer Science*, pages 299–320, Amsterdam, The Netherlands, April 28 – May 2, 2002. Springer, Berlin, Germany. 3, 57, 58, 59, 61, 62, 63, 64, 65, 68

[GVW12] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Functional encryption with bounded collusions via multi-party computation. Cryptology ePrint Archive, Report 2012/521, 2012. `http://eprint.iacr.org/`. 7

[GVW13] Sergey Gorbunov, Vinod Vaikuntanathan, and Hoeteck Wee. Attribute-based encryption for circuits. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 545–554. ACM, 2013. 6

[Had10] Satoshi Hada. Secure obfuscation for encrypted signatures. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 92–112, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. 7

[Hal05] Sean Hallgren. Fast quantum algorithms for computing the unit group and class group of a number field. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 468–474, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press. 56

[HGS04] Nick Howgrave-Graham and Michael Szydlo. A method to solve cyclotomic norm equations. In Duncan A. Buell, editor, *ANTS*, volume 3076 of *Lecture Notes in Computer Science*, pages 272–279. Springer, 2004. 3, 64, 65

[HHGP$^+$03] 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*, volume 2612 of *Lecture Notes in Computer Science*, pages 122–140. Springer, 2003. 3, 42, 64, 65, 66

[HK08] Dennis Hofheinz and Eike Kiltz. Programmable hash functions and their applications. In David Wagner, editor, *Advances in Cryptology – CRYPTO 2008*, volume 5157 of *Lecture Notes in Computer Science*, pages 21–38, Santa Barbara, CA, USA, August 17–21, 2008. Springer, Berlin, Germany. 9

[HKL$^+$00] Jeffrey Hoffstein, Burton S. Kaliski, Daniel Bennett Lieman, Matthew John Barton Robshaw, and Yiqun Lisa Yin. Secure user identification based on constrained polynomials. *US Patent 6,076,163*, 2000. 3, 42, 57, 58

[HMLS07] Dennis Hofheinz, John Malone-Lee, and Martijn Stam. Obfuscation for cryptographic purposes. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 214–232, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany. 7

[HPS98] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. Ntru: A ring-based public key cryptosystem. In *ANTS*, pages 267–288, 1998. 41

[HPS01] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NSS: An NTRU lattice-based signature scheme. In Birgit Pfitzmann, editor, *Advances in Cryptology – EUROCRYPT 2001*, volume 2045 of *Lecture Notes in Computer Science*, pages 211–228, Innsbruck, Austria, May 6–10, 2001. Springer, Berlin, Germany. 3, 42, 58

[HRSV07] Susan Hohenberger, Guy N. Rothblum, Abhi Shelat, and Vinod Vaikuntanathan. Securely obfuscating re-encryption. In Salil P. Vadhan, editor, *TCC 2007: 4th Theory of Cryptography Conference*, volume 4392 of *Lecture Notes in Computer Science*, pages 233–252, Amsterdam, The Netherlands, February 21–24, 2007. Springer, Berlin, Germany. 7

[HSW13] Susan Hohenberger, Amit Sahai, and Brent Waters. Full domain hash from (leveled) multilinear maps and identity-based aggregate signatures. Cryptology ePrint Archive, Report 2013/434, 2013. http://eprint.iacr.org/. 8, 9

[Jan96] Gerald J. Janusz. *Algebraic Number Fields.* American Mathematical Society, 1996. 23, 25

[Jou00] Antoine Joux. A one round protocol for tripartite diffie-hellman. In *Algorithmic Number Theory - ANTS'00*, volume 1838 of *Lecture Notes in Computer Science*, pages 385–394. Springer, 2000. 1, 5, 74

[Kal85a] Erich Kaltofen. Computing with polynomials given by straight-line programs i: Greatest common divisors. In Robert Sedgewick, editor, *STOC*, pages 131–142. ACM, 1985. 2, 41, 43

[Kal85b] Erich Kaltofen. Computing with polynomials given by straight-line programs ii: Sparse factorization. In *FOCS*, pages 451–458. IEEE Computer Society, 1985. 2, 41, 43

[Knu97] Donald Ervin Knuth. *The art of computer programming, Vol 2, 3rd ed.* 1997. 1

[KS98] Erich Kaltofen and Victor Shoup. Subquadratic-time factoring of polynomials over finite fields. *Math. Comput.*, 67(223):1179–1197, 1998. 56

[KSW08] Jonathan Katz, Amit Sahai, and Brent Waters. Predicate encryption supporting disjunctions, polynomial equations, and inner products. In *EUROCRYPT*, 2008. 7

[Lan90] S. Lang. *Cyclotomic Fields I and II: With and Appendix by Karl Rudin*. Graduate Texts in Mathematics. Springer-Verlag, 1990. 29

[Len83] Arjen K. Lenstra. Factoring polynominals over algebraic number fields. In J. A. van Hulzen, editor, *EUROCAL*, volume 162 of *Lecture Notes in Computer Science*, pages 245–254. Springer, 1983. 71

[Len13] Hendrik Lenstra. Lattices with symmetry. In *Proceedings of the 38th international symposium on International symposium on symbolic and algebraic computation*, ISSAC '13, pages 3–4, New York, NY, USA, 2013. ACM. 57

[LL93] Arjen K. Lenstra and Hendrik W. Lenstra. *The Development of the Number Field Sieve*, volume 1554 of *Lecture notes in mathematics*. Springer-Verlag, 1993. 64

[LLL82] A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982. 55, 72

[LLMP90] Arjen K. Lenstra, Hendrik W. Lenstra, Mark S. Manasse, and J.M. Pollard. The number field sieve. In *STOC*, volume 1554 of *Lecture Notes in Computer Science*, pages 564–572. ACM, 1990. 56, 64

[LM06] Vadim Lyubashevsky and Daniele Micciancio. Generalized compact Knapsacks are collision resistant. In Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener, editors, *ICALP 2006: 33rd International Colloquium on Automata, Languages and Programming, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 144–155, Venice, Italy, July 10–14, 2006. Springer, Berlin, Germany. 26

[LPR10] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 1–23, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. 23, 56

[LPR12] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. Cryptology ePrint Archive, Report 2012/230, 2012. http://eprint.iacr.org/. 30

[LPS04] Ben Lynn, Manoj Prabhakaran, and Amit Sahai. Positive results and techniques for obfuscation. In Christian Cachin and Jan Camenisch, editors, *Advances in Cryptology – EUROCRYPT 2004*, volume 3027 of *Lecture Notes in Computer Science*, pages 20–39, Interlaken, Switzerland, May 2–6, 2004. Springer, Berlin, Germany. 7

[Mic01] Daniele Micciancio. Improving lattice based cryptosystems using the hermite normal form. In Joseph H. Silverman, editor, *CaLC*, volume 2146 of *Lecture Notes in Computer Science*, pages 126–145. Springer, 2001. 55

[MR07] Daniele Micciancio and Oded Regev. Worst-case to average-case reductions based on gaussian measures. *SIAM J. Computing*, 37(1):267–302, 2007. 19, 20

[NR06] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. In Serge Vaudenay, editor, *Advances in Cryptology – EUROCRYPT 2006*, volume 4004 of *Lecture Notes in Computer Science*, pages 271–288, St. Petersburg, Russia, May 28 – June 1, 2006. Springer, Berlin, Germany. 3

[NR09] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. *Journal of Cryptology*, 22(2):139–160, April 2009. 3, 57, 65

[O'N10] Adam O'Neill. Definitional issues in functional encryption. Cryptology ePrint Archive, Report 2010/556, 2010. http://eprint.iacr.org/. 6

[Oss08] Brian Osserman. *Algebraic Number Theory*. Lecture Notes, 2008. https://www.math.ucdavis.edu/~osserman/classes/numthy/numthybook.pdf. 23, 27, 28

[Pei10] Chris Peikert. An efficient and parallel gaussian sampler for lattices. In Tal Rabin, editor, *Advances in Cryptology – CRYPTO 2010*, volume 6223 of *Lecture Notes in Computer Science*, pages 80–97, Santa Barbara, CA, USA, August 15–19, 2010. Springer, Berlin, Germany. 21, 50, 67

[PR07] Chris Peikert and Alon Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In David S. Johnson and Uriel Feige, editors, *39th Annual ACM Symposium on Theory of Computing*, pages 478–487, San Diego, California, USA, June 11–13, 2007. ACM Press. 56

[PTT10] Charalampos Papamanthou, Roberto Tamassia, and Nikos Triandopoulos. Optimal authenticated data structures with multilinear forms. In Marc Joye, Atsuko Miyaji, and Akira Otsuka, editors, *PAIRING 2010: 4th International Conference on Pairing-based Cryptography*, volume 6487 of *Lecture Notes in Computer Science*, pages 246–264, Yamanaka Hot Spring, Japan, December 13–15, 2010. Springer, Berlin, Germany. 2, 11

[Ram67] K. Ramachandra. On the units of cyclotomic fields. *Acta Arith.*, 12:165–173, 1966/67. 54

[Reg04] Oded Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004. 19

[Reg05] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 84–93, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press. 1, 41, 56

[Rot13] Ron Rothblum. On the circular security of bit-encryption. In Amit Sahai, editor, *TCC 2013: 10th Theory of Cryptography Conference*, volume 7785 of *Lecture Notes in Computer Science*, pages 579–598, Tokyo, Japan, March 3-6 2013. Springer. 2, 10, 11

[RS09] Markus Rückert and Dominique Schröder. Aggregate and verifiably encrypted signatures from multilinear maps without random oracles. In Jong Hyuk Park, Hsiao-Hwa Chen, Mohammed Atiquzzaman, Changhoon Lee, Tai-Hoon Kim, and Sang-Soo Yeo, editors, *ISA*, volume 5576 of *Lecture Notes in Computer Science*, pages 750–759. Springer, 2009. 1

[RSA78] Ronald L. Rivest, Adi Shamir, and Leonard M. Adleman. A method for obtaining digital signature and public-key cryptosystems. *Communications of the Association for Computing Machinery*, 21(2):120–126, 1978. 1, 6

[Rud89] Steven Rudich. Unpublished, 1989. 6

[Sch87] Claus-Peter Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theor. Comput. Sci.*, 53:201–224, 1987. 56, 72

[Sha85] Adi Shamir. Identity-based cryptosystems and signature schemes. In G. R. Blakley and David Chaum, editors, *Advances in Cryptology – CRYPTO'84*, volume 196 of *Lecture Notes in Computer Science*, pages 47–53, Santa Barbara, CA, USA, August 19–23, 1985. Springer, Berlin, Germany. 6

[Sho97a] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997. 56, 64

[Sho97b] Victor Shoup. Lower bounds for discrete logarithms and related problems. In Walter Fumy, editor, *Advances in Cryptology – EUROCRYPT'97*, volume 1233 of *Lecture Notes in Computer Science*, pages 256–266, Konstanz, Germany, May 11–15, 1997. Springer, Berlin, Germany. 41

[SL96] Peter Stevenhagen and Hendrik W Lenstra. Chebotarëv and his density theorem. *The Mathematical Intelligencer*, 18(2):26–37, 1996. 29

[SOK00] Ryuichi Sakai, Kiyoshi Ohgishi, and Masao Kasahara. Cryptosystems based on pairing. In *SCIS 2000*, Okinawa, Japan, January 2000. 9

[SS10] Amit Sahai and Hakan Seyalioglu. Worry-free encryption: functional encryption with public keys. In Ehab Al-Shaer, Angelos D. Keromytis, and Vitaly Shmatikov, editors, *ACM CCS 10: 17th Conference on Computer and Communications Security*, pages 463–472, Chicago, Illinois, USA, October 4–8, 2010. ACM Press. 7

[SS11] Damien Stehlé and Ron Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In Kenneth G. Paterson, editor, *Advances in Cryptology – EUROCRYPT 2011*, volume 6632 of *Lecture Notes in Computer Science*, pages 27–47, Tallinn, Estonia, May 15–19, 2011. Springer, Berlin, Germany. 30

[Ste04] William Stein. *A Brief Introduction to Classical and Adelic Algebraic Number Theory*. 2004. http://modular.math.washington.edu/129/ant/ant.pdf. 23, 24, 25, 27, 28

[Ste08] Peter Stevenhagen. The arithmetic of number rings. *Algorithmic Number Theory, Lattices, Number Fields, Curves and Cryptography*, 44:209–266, 2008. 56

[Ste10] C. L. Stewart. On divisors of lucas and lehmer numbers. 2010. 29

[SV05] Arthur Schmidt and Ulrich Vollmer. Polynomial time quantum algorithm for the computation of the unit group of a number field. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 475–480, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press. 56

[SV10] Nigel P. Smart and Frederik Vercauteren. Fully homomorphic encryption with relatively small key and ciphertext sizes. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010: 13th International Conference on Theory and Practice of Public Key Cryptography*, volume 6056 of *Lecture Notes in Computer Science*, pages 420–443, Paris, France, May 26–28, 2010. Springer, Berlin, Germany. 30, 73

[SW05] Amit Sahai and Brent R. Waters. Fuzzy identity-based encryption. In Ronald Cramer, editor, *Advances in Cryptology – EUROCRYPT 2005*, volume 3494 of *Lecture Notes in Computer Science*, pages 457–473, Aarhus, Denmark, May 22–26, 2005. Springer, Berlin, Germany. 6

[SW13] Alice Silverberg and Lawrence Washingoton, 2013. Personal Communication. 8, 29

[Szy03] Michael Szydlo. Hypercubic lattice reduction and analysis of GGH and NTRU signatures. In Eli Biham, editor, *Advances in Cryptology – EUROCRYPT 2003*, volume 2656 of *Lecture Notes in Computer Science*, pages 433–448, Warsaw, Poland, May 4–8, 2003. Springer, Berlin, Germany. 3, 64, 65

[vDGHV10] Marten van Dijk, Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. Fully homomorphic encryption over the integers. In Henri Gilbert, editor, *Advances in Cryptology – EUROCRYPT 2010*, volume 6110 of *Lecture Notes in Computer Science*, pages 24–43, French Riviera, May 30 – June 3, 2010. Springer, Berlin, Germany. 4

[Ver13] Fré Vercauteren. Final report on main computational assumptions in cryptography. 2013. http://www.ecrypt.eu.org/documents/D.MAYA.6.pdf. Last Accessed: 19 May 2013. 1

[Was82] L.C. Washington. *Introduction to Cyclotomic Fields*. Graduate texts in mathematics. Springer-Verlag, 1982. 54

[Was97] L.C. Washington. *Introduction to Cyclotomic Fields*. Graduate Texts in Mathematics. Springer-Verlag, 1997. 29

[Wee05] Hoeteck Wee. On obfuscating point functions. In Harold N. Gabow and Ronald Fagin, editors, *37th Annual ACM Symposium on Theory of Computing*, pages 523–532, Baltimore, Maryland, USA, May 22–24, 2005. ACM Press. 7

[Wes99] Tom Weston. *Algebraic Number Theory*. Course Notes, 1999. https://www.math.umass.edu/~weston/cn/notes.pdf. 23, 25, 26, 27, 28