A Decade of Lattice Cryptography

Chris Peikert[1]

February 17, 2016

[1]Department of Computer Science and Engineering, University of Michigan. Much of this work was done while at the School of Computer Science, Georgia Institute of Technology. This material is based upon work supported by the National Science Foundation under CAREER Award CCF-1054495, by DARPA under agreement number FA8750-11-C-0096, and by the Alfred P. Sloan Foundation. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation, DARPA or the U.S. Government, or the Sloan Foundation. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation thereon.

Abstract

Lattice-based cryptography is the use of conjectured hard problems on point lattices in $\mathbb{R}^n$ as the foundation for secure cryptographic systems. Attractive features of lattice cryptography include apparent resistance to *quantum* attacks (in contrast with most number-theoretic cryptography), high asymptotic efficiency and parallelism, security under *worst-case* intractability assumptions, and solutions to long-standing open problems in cryptography.

This work surveys most of the major developments in lattice cryptography over the past ten years. The main focus is on the foundational *short integer solution* (SIS) and *learning with errors* (LWE) problems (and their more efficient ring-based variants), their provable hardness assuming the worst-case intractability of standard lattice problems, and their many cryptographic applications.

Contents

Chapter 1

Introduction

This survey provides an overview of *lattice-based cryptography*, the use of apparently hard problems on point lattices in $\mathbb{R}^n$ as the foundation for secure cryptographic constructions. Lattice cryptography has many attractive features, some of which we now describe.

Conjectured security against *quantum* attacks. Most number-theoretic cryptography, such as the Diffie-Hellman protocol [DH76] and RSA cryptosystem [RSA78], relies on the conjectured hardness of integer factorization or the discrete logarithm problem in certain groups. However, Shor [Sho97] gave efficient *quantum* algorithms for all these problems, which would render number-theoretic systems insecure in a future where large-scale quantum computers are available. By contrast, no efficient quantum algorithms are known for the problems typically used in lattice cryptography; indeed, generic (and relatively modest) quantum speedups provide the only known advantage over non-quantum algorithms.

Algorithmic simplicity, efficiency, and parallelism. Lattice-based cryptosystems are often algorithmically simple and highly parallelizable, consisting mainly of linear operations on vectors and matrices modulo relatively small integers. Moreover, constructions based on "algebraic" lattices over certain rings (e.g., the NTRU cryptosystem [HPS98]) can be especially efficient, and in some cases even outperform more traditional systems by a significant margin.

Strong security guarantees from *worst-case* hardness. Cryptography inherently requires *average-case* intractability, i.e., problems for which *random* instances (drawn from a specified probability distribution) are hard to solve. This is qualitatively different from the *worst-case* notion of hardness usually considered in the theory of algorithms and NP-completeness, where a problem is considered hard if there merely *exist* some intractable instances. Problems that appear hard in the worst case often turn out to be easier on the average, especially for distributions that produce instances having some extra "structure," e.g., the existence of a secret key for decryption.

In a seminal work, Ajtai [Ajt96] gave a remarkable connection between the worst case and the average case for lattices: he proved that certain problems are hard on the average (for cryptographically useful distributions), as long as some related lattice problems are hard in the worst case. Using results of this kind, one can design cryptographic constructions and prove that they are infeasible to break, unless *all* instances of certain lattice problems are easy to solve.[1]

[1] Note that many number-theoretic problems used in cryptography, such as discrete logarithm and quadratic residuosity, also admit

Constructions of versatile and powerful cryptographic objects. Historically, cryptography was mainly about sending secret messages. Yet over the past few decades, the field has blossomed into a discipline having much broader and richer goals, encompassing almost any scenario involving communication or computation in the presence of potentially malicious behavior. For example, the powerful notion of *fully homomorphic encryption* (FHE), first envisioned by Rivest *et al.* [RAD78], allows an untrusted worker to perform arbitrary computations on encrypted data, without learning anything about that data. For three decades FHE remained an elusive "holy grail" goal, until Gentry [Gen09b, Gen09a] proposed the first candidate construction of FHE, which was based on lattices (as were all subsequent constructions). More recently, lattices have provided the only known realizations of other versatile and powerful cryptographic notions, such as attribute-based encryption for arbitrary access policies [GVW13, BGG$^+$14] and general-purpose code obfuscation [GGH$^+$13b].

1.1 Scope and Organization

This work surveys most of the major developments in lattice cryptography over the past decade (since around 2005). The main focus is on two foundational average-case problems, called the *short integer solution* (SIS) and *learning with errors* (LWE) problems; their provable hardness assuming the worst-case intractability of lattice problems; and the plethora of cryptographic constructions that they enable.

Most of this survey should be generally accessible to early-stage graduate students in theoretical computer science, or even to advanced undergraduates. However, understanding the finer details of the cryptographic constructions—especially the outlines of their security proofs, which we have deliberately left informal so as to highlight the main ideas—may require familiarity with basic cryptographic definitions and paradigms, which can be obtained from any graduate-level course or the textbooks by, e.g., Katz and Lindell [KL14] or Goldreich [Gol01]. The reader who lacks such background is encouraged to focus on the essential ideas and mechanics of the cryptosystems, and may safely skip over the proof summaries.

The survey is organized as follows:

- Chapter 2 recalls the necessary mathematical and cryptographic background.
- Chapter 3 gives a high-level conceptual overview of the seminal works in the area and their significance.
- Chapter 4 covers the modern foundations of the area, which have largely subsumed the earlier works. Here we formally define the SIS and LWE problems and recall the theorems which say that these problems are at least as hard to solve as certain worst-case lattice problems. We also cover their more compact and efficient *ring-based* analogues, ring-SIS and ring-LWE.
- Chapter 5 describes a wide variety of essential lattice-based cryptographic constructions, ranging from basic encryption and digital signatures to more powerful objects like identity-based encryption. These schemes are presented within a unified framework, using just a handful of concepts and technical tools that are developed throughout the chapter.
- Chapter 6 describes a few more advanced cryptographic constructions, with a focus on fully homomorphic encryption and attribute-based encryption.
- Chapter 7 concludes with a discussion of some important open questions in the area.

(comparatively simple) worst-case/average-case reductions, but only within a *fixed* group. Such a reduction gives us a distribution over a group which is as hard as the worst case for the *same* group, but says nothing about whether the group itself is hard, or which groups are hardest. Indeed, the complexity of these problems appears to vary quite widely depending on the type of group (e.g., multiplicative groups of integers modulo a prime or of other finite fields, elliptic curve groups, etc.).

While we have aimed to convey a wide variety of lattice-based cryptographic constructions and their associated techniques, our coverage of such a large and fast-growing area is necessarily incomplete. For one, we do not discuss cryptanalysis or concrete parameters (key sizes etc.) of lattice-based cryptosystems; representative works on these topics include [GN08, MR09, GNR10, LP11, CN11, LN13]. We also do not include any material on the recent seminal constructions of candidate *multilinear maps* [GGH13a, CLT13, GGH15, CLT15] and their many exciting applications, such as general-purpose code obfuscation [GGH$^+$13b, SW14]. While all multilinear map constructions to date are related to lattices, their conjectured security relies on new, ad-hoc problems that are much less well-understood than SIS/LWE. In particular, it is not known whether any of the proposed constructions can be proved secure under worst-case hardness assumptions, and some candidates have even been broken in certain ways (see, e.g., [CHL$^+$15, HJ15, CL15, CGH$^+$15]). Note that early constructions of fully homomorphic encryption also relied on ad-hoc assumptions, but constructions based on more standard assumptions like (ring-)LWE soon followed; the same may yet occur for multilinear maps and the applications they enable.

1.2 Other Resources

There are several other resources on modern lattice cryptography, or specialized subtopics thereof. (However, due to the rapid development of the field over the past few years, these surveys are already a bit dated in their coverage of advanced cryptographic constructions and associated techniques.) Some excellent options include:

- The 2007 survey by Micciancio [Mic07] on cryptographic functions from worst-case complexity assumptions, including ring-based functions;
- the 2009 survey by Micciancio and Regev [MR09] on lattice-based cryptographic constructions and their cryptanalysis;
- the 2010 survey by Regev [Reg10] on the learning with errors (LWE) problem, its worst-case hardness, and some early applications;
- the overviews of fully homomorphic encryption by Gentry [Gen10a] and Vaikuntanathan [Vai11];
- videos from the 2012 Bar-Ilan Winter School on Lattice Cryptography and Applications [Bar12];
- other surveys, books, and course notes [NS01, MG02, Reg04, Mic14] on computational aspects of lattices, including cryptanalysis.

Acknowledgments. I warmly thank Vadim Lyubashevsky, Dieter van Melkebeek, Oded Regev, Noah Stephens-Davidowitz, Madhu Sudan, and an anonymous reviewer for many valuable comments on earlier drafts.

Chapter 2

Background

2.1 Notation

For a real number $x \in \mathbb{R}$, we let $\lfloor x \rfloor$ denote the largest integer not greater than x, and $\lfloor x \rceil := \lfloor x + 1/2 \rfloor$ denote the integer closest to x, with ties broken upward.

We use bold lower-case letters like $\mathbf{x}$ to denote column vectors; for row vectors we use the transpose $\mathbf{x}^t$. We use bold upper-case letters like $\mathbf{A}$ to denote matrices, and sometimes identify a matrix with its ordered set of column vectors. We denote the horizontal concatenation of vectors and/or matrices using a vertical bar, e.g., $[\mathbf{A} \mid \mathbf{Ax}]$. We sometimes apply functions entry-wise to vectors, e.g., $\lfloor \mathbf{x} \rceil$ rounds each entry of $\mathbf{x}$ to its nearest integer.

For a positive integer q, let $\mathbb{Z}_q = \mathbb{Z}/q\mathbb{Z}$ denote the quotient ring of integers modulo q, i.e., the collection of cosets $a + q\mathbb{Z}$ with the induced addition and multiplication operations. Similarly, because $\mathbb{Z}_q$ is an additive group, it supports multiplication by integers, i.e., $z \cdot a \in \mathbb{Z}_q$ for an integer $z \in \mathbb{Z}$ and $a \in \mathbb{Z}_q$. We often write $z \bmod q$ to denote the coset $z + q\mathbb{Z}$, and $y = z \pmod q$ to denote that $y + q\mathbb{Z} = z + q\mathbb{Z}$.

We use standard asymptotic notation $O(\cdot), \Omega(\cdot), \Theta(\cdot), o(\cdot)$, etc. In addition, tildes (e.g., $\tilde{O}(\cdot)$) indicate that logarithmic factors in the main parameter are suppressed.

2.2 Lattices

This survey requires minimal knowledge of lattices beyond some basic definitions and computational problems, which we recall here. (See the other resources listed in the introduction for much more background.)

2.2.1 Basic Definitions

An n-dimensional *lattice* $\mathcal{L}$ is any subset of $\mathbb{R}^n$ that is both:

1. an *additive subgroup*: $\mathbf{0} \in \mathcal{L}$, and $-\mathbf{x}, \mathbf{x} + \mathbf{y} \in \mathcal{L}$ for every $\mathbf{x}, \mathbf{y} \in \mathcal{L}$; and
2. *discrete*: every $\mathbf{x} \in \mathcal{L}$ has a neighborhood in $\mathbb{R}^n$ in which $\mathbf{x}$ is the only lattice point.

Examples include the integer lattice $\mathbb{Z}^n$, the scaled lattice $c\mathcal{L}$ for any real number c and lattice $\mathcal{L}$, and the "checkerboard" lattice $\{\mathbf{x} \in \mathbb{Z}^n : \sum_i x_i \text{ is even}\}$.

The *minimum distance* of a lattice $\mathcal{L}$ is the length of a shortest nonzero lattice vector:

$$\lambda_1(\mathcal{L}) := \min_{\mathbf{v} \in \mathcal{L} \setminus \{\mathbf{0}\}} \|\mathbf{v}\|.$$

(Unless otherwise specified, $\|\cdot\|$ denotes the Euclidean norm.) More generally, the ith successive minimum $\lambda_i(\mathcal{L})$ is the smallest r such that $\mathcal{L}$ has i linearly independent vectors of norm at most r.

Because a lattice $\mathcal{L}$ is an additive subgroup of $\mathbb{R}^n$, we have the quotient group $\mathbb{R}^n/\mathcal{L}$ of cosets

$$\mathbf{c} + \mathcal{L} = \{\mathbf{c} + \mathbf{v} : \mathbf{v} \in \mathcal{L}\}, \quad \mathbf{c} \in \mathbb{R}^n,$$

with the usual induced addition operation $(\mathbf{c}_1 + \mathcal{L}) + (\mathbf{c}_2 + \mathcal{L}) = (\mathbf{c}_1 + \mathbf{c}_2) + \mathcal{L}$. A *fundamental domain* of $\mathcal{L}$ is a set $\mathcal{F} \subset \mathbb{R}^n$ that contains exactly one representative $\bar{\mathbf{c}} \in (\mathbf{c} + \mathcal{L}) \cap \mathcal{F}$ of every coset $\mathbf{c} + \mathcal{L}$. For example, the half-open intervals $[0, 1)$ and $[-\frac{1}{2}, \frac{1}{2})$ are fundamental domains of the integer lattice $\mathbb{Z}$, where coset $c + \mathbb{Z}$ has representative $c - \lfloor c \rfloor$ and $c - \lfloor c \rceil$, respectively.

Bases and fundamental parallelepipeds. Although every (non-trivial) lattice $\mathcal{L}$ is infinite, it is always finitely generated as the integer linear combinations of some linearly independent *basis* vectors $\mathbf{B} = \{\mathbf{b}_1, \ldots, \mathbf{b}_k\}$:

$$\mathcal{L} = \mathcal{L}(\mathbf{B}) := \mathbf{B} \cdot \mathbb{Z}^k = \left\{ \sum_{i=1}^{k} z_i \mathbf{b}_i : z_i \in \mathbb{Z} \right\}.$$

The integer k is called the *rank* of the basis, and is an invariant of the lattice. For the remainder of this survey we restrict our attention to *full-rank* lattices, where $k = n$. A lattice basis $\mathbf{B}$ is not unique: for any unimodular matrix $\mathbf{U} \in \mathbb{Z}^{n \times n}$ (i.e., one having determinant ± 1), $\mathbf{B} \cdot \mathbf{U}$ is also a basis of $\mathcal{L}(\mathbf{B})$, because $\mathbf{U} \cdot \mathbb{Z}^n = \mathbb{Z}^n$.

For a lattice $\mathcal{L}$ having basis $\mathbf{B}$, a commonly used fundamental domain is the origin-centered *fundamental parallelepiped* $\mathcal{P}(\mathbf{B}) := \mathbf{B} \cdot [-\frac{1}{2}, \frac{1}{2})^n$, where coset $\mathbf{c} + \mathcal{L}$ has representative $\mathbf{c} - \mathbf{B} \cdot \lfloor \mathbf{B}^{-1} \cdot \mathbf{c} \rceil$.

The dual lattice. The *dual* (sometimes called *reciprocal*) of a lattice $\mathcal{L} \subset \mathbb{R}^n$ is defined as

$$\mathcal{L}^* := \{\mathbf{w} : \langle \mathbf{w}, \mathcal{L} \rangle \subseteq \mathbb{Z}\},$$

i.e., the set of points whose inner products with the vectors in $\mathcal{L}$ are all integers. It is straightforward to verify that $\mathcal{L}^*$ is a lattice. For example, $(\mathbb{Z}^n)^* = \mathbb{Z}^n$, and $(c\mathcal{L})^* = c^{-1}\mathcal{L}^*$ for any nonzero real c and lattice $\mathcal{L}$. It is also easy to verify that if $\mathbf{B}$ is a basis of $\mathcal{L}$, then $\mathbf{B}^{-t} := (\mathbf{B}^t)^{-1} = (\mathbf{B}^{-1})^t$ is a basis of $\mathcal{L}^*$.

2.2.2 Computational Problems

We now define some of the computational problems on lattices that have been most useful in cryptography, and recall some results about their complexity. (There are many other problems that have been extensively studied in mathematics and computational complexity, but so far have had less direct importance to cryptography.) Perhaps the most well-studied computational problem on lattices is the *shortest vector problem*:

Definition 2.2.1 (Shortest Vector Problem (SVP)). Given an arbitrary basis $\mathbf{B}$ of some lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$, find a shortest nonzero lattice vector, i.e., a $\mathbf{v} \in \mathcal{L}$ for which $\|\mathbf{v}\| = \lambda_1(\mathcal{L})$.

Particularly important to lattice cryptography are *approximation* problems, which are parameterized by an approximation factor $\gamma \geq 1$ that is typically taken to be a function of the lattice dimension n, i.e., $\gamma = \gamma(n)$. For example, the approximation version of SVP is as follows (note that by setting $\gamma(n) = 1$ we recover the problem defined above):

Definition 2.2.2 (Approximate Shortest Vector Problem (SVP_γ)). Given a basis $\mathbf{B}$ of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$, find a nonzero vector $\mathbf{v} \in \mathcal{L}$ for which $\|\mathbf{v}\| \leq \gamma(n) \cdot \lambda_1(\mathcal{L})$.

As described in later sections, several cryptosystems can be proved secure assuming the hardness of certain lattice problems, in the worst case. However, to date no such proof is known for the *search* version of SVP_γ. Instead, there are proofs based on the following *decision* version of approximate-SVP, as well as a *search* problem related to the nth successive minimum:

Definition 2.2.3 (Decisional Approximate SVP (GapSVP_γ)). Given a basis $\mathbf{B}$ of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$ where either $\lambda_1(\mathcal{L}) \leq 1$ or $\lambda_1(\mathcal{L}) > \gamma(n)$, determine which is the case.

Definition 2.2.4 (Approximate Shortest Independent Vectors Problem (SIVP_γ)). Given a basis $\mathbf{B}$ of a full-rank n-dimensional lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$, output a set $\mathbf{S} = \{\mathbf{s}_i\} \subset \mathcal{L}$ of n linearly independent lattice vectors where $\|\mathbf{s}_i\| \leq \gamma(n) \cdot \lambda_n(\mathcal{L})$ for all i.

A final important problem for cryptography is the following *bounded-distance decoding* BDD_γ problem, which asks to find the lattice vector that is closest to a given target point $\mathbf{t} \in \mathbb{R}^n$, where the target is promised to be "rather close" to the lattice. This promise, and the uniqueness of the solution, are what distinguish BDD_γ from the approximate *closest vector problem* CVP_γ, wherein the target can be an arbitrary point. (Because no cryptosystem has yet been proved secure based on CVP_γ, we do not formally define that problem here.)

Definition 2.2.5 (Bounded Distance Decoding Problem (BDD_γ)). Given a basis $\mathbf{B}$ of an n-dimensional lattice $\mathcal{L} = \mathcal{L}(\mathbf{B})$ and a target point $\mathbf{t} \in \mathbb{R}^n$ with the guarantee that $\mathrm{dist}(\mathbf{t}, \mathcal{L}) < d = \lambda_1(\mathcal{L})/(2\gamma(n))$, find the unique lattice vector $\mathbf{v} \in \mathcal{L}$ such that $\|\mathbf{t} - \mathbf{v}\| < d$.

Algorithms and complexity. The above lattice problems have been intensively studied and appear to be intractable, except for very large approximation factors. Known *polynomial-time* algorithms like the one of Lenstra, Lenstra, and Lovász [LLL82] and its descendants (e.g., [Sch87] with [AKS01] as a subroutine) obtain only slightly subexponential approximation factors $\gamma = 2^{\Theta(n \log\log n / \log n)}$ for all the above problems (among many others that are less relevant to cryptography). Known algorithms that obtain polynomial $\mathrm{poly}(n)$ or better approximation factors, such as [Kan83, AKS01, MV10, ADRS15], either require super-exponential $2^{\Theta(n \log n)}$ time, or exponential $2^{\Theta(n)}$ time *and* space. There are also time-approximation tradeoffs that interpolate between these two classes of results, to obtain $\gamma = 2^k$ approximation factors in $2^{\tilde{\Theta}(n/k)}$ time [Sch87]. Importantly, the above also represents the state of the art for *quantum* algorithms, though in some cases the hidden constant factors in the exponents are somewhat smaller (see, e.g., [LMvdP13]). By contrast, recall that the integer factorization and discrete logarithm problem (in essentially any group) can be solved in polynomial time using Shor's quantum algorithm [Sho97].

On the complexity side, many lattice problems are known to be *NP-hard* (sometimes under randomized reductions), even to approximate to within various sub-polynomial $n^{o(1)}$ approximation factors. E.g., for the hardness of SVP, see [Ajt98, Mic98, Kho03, HR07]. However, such hardness is not of any direct consequence to cryptography, since lattice-based cryptographic constructions so far rely on polynomial approximation problems factors $\gamma(n) \geq n$. Indeed, there is evidence that for factors $\gamma(n) \geq \sqrt{n}$, the lattice problems relevant to cryptography are *not* NP-hard, because they lie in NP $\cap$ coNP [GG98, AR04].

2.3 (Discrete) Gaussians and Subgaussians

Many modern works on lattices in complexity and cryptography rely on Gaussian-like probability distributions over lattices, called *discrete Gaussians*. Here we recall the relevant definitions.

Gaussians. For any positive integer n and real $s > 0$, which is taken to be $s = 1$ when omitted, define the *Gaussian function* $\rho_s \colon \mathbb{R}^n \to \mathbb{R}^+$ of *parameter* (or *width*) s as

$$\rho_s(\mathbf{x}) := \exp(-\pi \|\mathbf{x}\|^2 / s^2) = \rho(\mathbf{x}/s).^1$$

Notice that ρ_s is invariant under rotations of $\mathbb{R}^n$, and that $\rho_s(\mathbf{x}) = \prod_{i=1}^n \rho_s(x_i)$.

The (continuous) *Gaussian distribution* D_s of parameter s over $\mathbb{R}^n$ is defined to have probability density function proportional to ρ_s, i.e.,

$$f(\mathbf{x}) := \rho_s(\mathbf{x}) / \int_{\mathbb{R}^n} \rho_s(\mathbf{z})\, d\mathbf{z} = \rho_s(\mathbf{x})/s^n.$$

For a lattice coset $\mathbf{c} + \mathcal{L} \subset \mathbb{R}^n$ and parameter $s > 0$, the *discrete Gaussian* probability distribution $D_{\mathbf{c}+\mathcal{L},s}$ is simply the Gaussian distribution restricted to the coset:

$$D_{\mathbf{c}+\mathcal{L},s}(\mathbf{x}) \propto \begin{cases} \rho_s(\mathbf{x}) & \text{if } \mathbf{x} \in \mathbf{c} + \mathcal{L} \\ 0 & \text{otherwise.} \end{cases}$$

Smoothing parameter. Micciancio and Regev [MR04] introduced a very important quantity called the *smoothing parameter* of a lattice $\mathcal{L}$. Informally, this is the amount of Gaussian "blur" required to "smooth out" essentially all the discrete structure of $\mathcal{L}$. Alternatively, it can be seen as the smallest width $s > 0$ such that every coset $\mathbf{c} + \mathcal{L}$ has nearly the same Gaussian mass $\rho_s(\mathbf{c} + \mathcal{L}) := \sum_{\mathbf{x} \in \mathbf{c}+\mathcal{L}} \rho_s(\mathbf{x})$, up to some small relative error.

Formally, the smoothing parameter $\eta_\varepsilon(\mathcal{L})$ is parameterized by a tolerance $\varepsilon > 0$, and is defined using the dual lattice as the minimal $s > 0$ such that $\rho_{1/s}(\mathcal{L}^*) \leq 1 + \varepsilon$. This condition can be used to formalize and prove the above-described "smoothing" properties. For the purposes of this survey, we often omit ε and implicitly take it to be very small, e.g., a negligible $n^{-\omega(1)}$ function in the dimension n of the lattice.

The smoothing parameter is closely related to other standard lattice quantities. For example:

Theorem 2.3.1 ([Ban93, MR04]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$, we have $\eta_{2^{-n}}(\mathcal{L}) \leq \sqrt{n}/\lambda_1(\mathcal{L}^*)$.*

Theorem 2.3.2 ([MR04, GPV08]). *For any full-rank lattice $\mathcal{L} \subseteq \mathbb{R}^n$ and $\varepsilon \in (0, 1/2)$,*

$$\eta_\varepsilon(\mathcal{L}) \leq \min_{\text{basis } \mathbf{B} \text{ of } \mathcal{L}} \|\widetilde{\mathbf{B}}\| \cdot \sqrt{\log O(n/\varepsilon)} \leq \lambda_n(\mathcal{L}) \cdot \sqrt{\log O(n/\varepsilon)},$$

where $\|\widetilde{\mathbf{B}}\| = \max_i \|\widetilde{\mathbf{b}}_i\|$ denotes the maximal length of the Gram-Schmidt orthogonalized vectors $\{\widetilde{\mathbf{b}}_i\}$ of the ordered basis $\mathbf{B} = \{\mathbf{b}_i\}$.

Several works, e.g., [Ban93, Ban95, AR04, MR04, Reg05, Pei07, Pei10, BF11, ADRS15], have shown that when when $s \geq \eta(\mathcal{L})$, the discrete Gaussian distribution $D_{\mathbf{c}+\mathcal{L},s}$ behaves very much like the continuous Gaussian D_s in many important respects. For example, their moments and tails are nearly the same, and the sum of independent discrete Gaussians is a discrete Gaussian.

[1] The normalization factor π in the exponential is chosen so that ρ is its own Fourier transform.

Subgaussianity. Informally, a random variable (or its distribution) is *subgaussian* if it is dominated by a Gaussian. Formally, a real random variable X is subgaussian with parameter s if for every $t \geq 0$, we have[2]

$$\Pr[|X| > t] \leq 2\exp(-\pi t^2/s^2).$$

More generally, a random vector $\mathbf{x}$ over $\mathbb{R}^n$ is subgaussian with parameter s if every marginal $\langle \mathbf{x}, \mathbf{u} \rangle$ is, for all unit vectors $\mathbf{u} \in \mathbb{R}^n$. It is not hard to show that the concatenation of independent subgaussian random variables or vectors of common parameter s is itself a subgaussian vector of parameter s.

Examples of subgaussian distributions with parameter s include any symmetric random variable having magnitude bounded by $s/\sqrt{2\pi}$; the continuous Gaussian D_s and discrete Gaussian $D_{\mathcal{L},s}$ over any lattice $\mathcal{L}$; and the discrete Gaussian $D_{\mathbf{c}+\mathcal{L},s}$ over any lattice coset when $s \geq \eta(\mathcal{L})$ (under a slight relaxation of subgaussianity; see [MP12, Section 2.4]).

2.4 Cryptographic Background

Cryptography is concerned with a variety of different kinds of objects and security properties they can satisfy. Here we give a brief, informal overview of the main concepts that are relevant to this survey. For further details and formalisms, see, e.g., [KL14, Gol01].

In complexity-theoretic (as opposed to information-theoretic) cryptography, the *security parameter* λ regulates the running times of all algorithms, including the attacker, along with the latter's measure of success, called its *advantage*. A typical requirement is that all algorithms (including the attacker) have running times that are polynomial $\lambda^{O(1)}$ in the security parameter, and that the attacker's advantage is *negligible* $\lambda^{-\omega(1)}$, i.e., asymptotically smaller than the inverse of any polynomial in λ. (One can be even more permissive about the adversary, e.g., allowing its running time and/or inverse advantage to be subexponential in λ, allowing it to be non-uniform, etc.) In what follows, all function families are implicitly indexed by λ, and all algorithms (including the attacker) are given λ as an input.

2.4.1 Function Families and Security Properties

A function *family* is a function $f\colon K \times X \to Y$ for some space K of *keys*, domain X, and range Y. We call it a family because it defines the set of functions $f_k(\cdot) = f(k, \cdot)$ for keys $k \in K$. If $|X| > |Y|$, i.e., the function "compresses" its input by some amount, it is often called a *hash* function. The following are two commonly used security properties of function families:

- A family f is *one way* if it is "hard to invert" for random inputs. More precisely, given $k \in K$ and $y = f_k(x) \in Y$ where k, x are randomly chosen from prescribed distributions, it is infeasible to find any preimage $x' \in f_k^{-1}(y)$.
- A family f is *collision resistant* if it is hard to find a collision for a random key. More precisely, given a random $k \in K$ (chosen from a prescribed distribution), it is infeasible to find distinct $x, x' \in X$ such that $f_k(x) = f_k(x')$.

[2]There are other equivalent definitions and slight relaxations of subgaussianity that are often useful in lattice cryptography; see [Ver12, Lemma 5.5] and [MP12, Section 2.4] for details.

2.4.2 Public-Key Encryption

An asymmetric (also known as public-key) encryption scheme is a triple of randomized algorithms having the following interfaces:

- The *key generator*, given the security parameter, outputs a public key and secret key.
- The *encryption* algorithm takes a public key and a message (from some known set of valid messages) and outputs a ciphertext.
- The *decryption* algorithm takes a secret key and a ciphertext, and outputs either a message or a distinguished "failure" symbol.

Naturally, the scheme is said to be *correct* if generating a key pair, then encrypting a valid message using the public key, then decrypting the resulting ciphertext using the secret key, yields the original message (perhaps with all but negligible probability).

A standard notion of security, called *semantic security* [GM82], or *indistinguishability under chosen-plaintext attack* (IND-CPA), informally guarantees that encryption reveals nothing about encrypted messages to a passive (eavesdropping) adversary. Formally, the definition considers the following experiment, which is parameterized by a bit $b \in \{0, 1\}$:

1. Generate a public/secret key pair, and give the public key to the attacker, who must reply with two valid messages m_0, m_1. (If the valid message space is just $\{0, 1\}$, then we may assume that $m_b = b$ without loss of generality.)
2. Encrypt m_b using the public key and give the resulting "challenge ciphertext" to the attacker.
3. Finally, the attacker either accepts or rejects.

An encryption scheme is said to be semantically (or IND-CPA) secure if it is infeasible for an attacker to distinguish between the two cases $b = 0$ and $b = 1$. That is, its probabilities of accepting in the two cases should differ by only a negligible amount.

A much stronger notion of security, called *active* security—or more formally, indistinguishability under chosen-*ciphertext* attack (IND-CCA)—augments the above experiment by giving the attacker access to a *decryption* oracle, i.e., one that runs the decryption algorithm (with the secret key) on any ciphertext the attacker may query, *except* for the challenge ciphertext. (This restriction is of course necessary, because otherwise the attacker could just request to decrypt the challenge ciphertext and thereby learn the value of b.) The scheme is said to be actively (or IND-CCA) secure if it is infeasible for an attacker to distinguish between the two cases $b = 0$ and $b = 1$.

2.4.3 Richer Forms of Encryption

Over the past several years, there has been an increasing interest in encryption systems having additional useful features. For example, *homomorphic encryption* allows an untrusted worker to perform meaningful computations on encrypted data, without revealing anything about the data to the worker. In this context, the basic syntax and notion of IND-CPA security remain exactly the same, but there is an additional algorithm for performing a desired homomorphic computation on ciphertexts.

Another example is *identity-based encryption* (IBE), in which any string (e.g., a user's email address) can serve as a public key. Here the model is slightly different: an IBE is a four-tuple of randomized algorithms having the following interfaces:

- The *setup* algorithm, given the security parameter, outputs a "master" public and secret key pair.
- The *key extraction* algorithm takes a master secret key and an identity string, and outputs a secret key for that particular identity.
- The *encryption* algorithm takes a master public key, an identity string, and a valid message, and outputs a ciphertext.
- The *decryption* algorithm takes an identity secret key and a ciphertext, and outputs a message (or a failure symbol).

Correctness is defined in the expected way: for a ciphertext encrypted to a particular identity string, decrypting using a corresponding secret key (produced by the key extraction algorithm) should return the original message.

Informally, semantic security for IBE is defined by modifying the standard IND-CPA experiment to model the property that even if many users collude by combining their secret keys, they still learn nothing about messages encrypted to another user. More formally, we consider the following experiment: the attacker is given the master public key, along with oracle access to the key extraction algorithm (with the master secret key built in), thus allowing it to obtain secret keys for any identities of its choice. At some point, the attacker produces two messages m_0, m_1 and a *target* identity, which must be different from all the queries it ever makes to its key-extraction oracle. The attacker is then given a ciphertext that encrypts m_b to the target identity, and may make further queries to its oracle before either accepting or rejecting.

Chapter 3

Early Results

In this chapter we briefly survey some of the pioneering works in lattice cryptography. At a conceptual level, the ideas from these works persist to this day throughout the area. However, their precise formulations and analytical techniques have largely been improved and subsumed by subsequent works, so we do not go into much technical detail in this chapter.

3.1 Ajtai's Function and Ajtai-Dwork Encryption

In a groundbreaking work, Ajtai [Ajt96] gave the first *worst-case to average-case reductions* for lattice problems, and with them the first cryptographic object with a proof of security assuming the hardness of well-studied computational problems on lattices. In particular, Ajtai's work gave the first cryptographic function based on a standard *worst-case* complexity assumption of any kind. Ajtai introduced the (average-case) "short integer solution" (SIS) problem and its associated one-way function, and proved that solving it is at least as hard as approximating various lattice problems in the worst case. Both SIS and Ajtai's function are still heavily used to this day in cryptographic applications; we recall them in detail in Section 4.1.

In a subsequent work from 1997, Ajtai and Dwork [AD97] gave a lattice-based *public-key encryption* scheme. (See also [AD07].) Because all lattice-based encryption schemes inherit the basic template of this system, we describe it in some detail here. At the highest level, Ajtai and Dwork give two main results: first, they show that a certain *average-case* "hidden hyperplanes problem" HHP in $\mathbb{R}^n$ is at least as hard as the "γ-unique shortest vector problem" uSVP_γ on *arbitrary* n-dimensional lattices, for some polynomial $\gamma = \mathrm{poly}(n)$ in the dimension n. (We informally define these problems below.) Second, they construct a public-key cryptosystem where semantic security can be proved based on the hardness of the hidden hyperplanes problem, and hence on the conjectured worst-case hardness of uSVP_γ.

As the first encryption scheme with a security proof under a worst-case complexity assumption, Ajtai-Dwork was a theoretical breakthrough. However, from a practical (and even theoretical) point of view, it has some significant drawbacks: its public keys are of size $\tilde{O}(n^4)$, and its secret keys and ciphertexts are of size $\tilde{O}(n^2)$, with matching runtimes for encryption and decryption, respectively. For concrete security against cryptanalytic attacks on the hidden hyperplanes problem [NS98] (and thus to prevent key-recovery attacks), the value of n must be in the hundreds, thus yielding public key sizes of several gigabits. Moreover, each ciphertext encrypts only a single bit, so to encrypt (say) a 128-bit symmetric key requires ciphertext sizes in the several megabits. Later works, described in subsequent sections, largely resolved (or at least greatly mitigated) these drawbacks.

Hidden hyperplanes and unique-SVP. Informally, the *hidden hyperplanes problem* introduced by Ajtai and Dwork is as follows: let $\mathbf{s} \in \mathbb{R}^n$ be a secret, random short vector. The goal is to find $\mathbf{s}$, given many points $\mathbf{y}_i \in \mathbb{R}^n$ such that $\langle \mathbf{s}, \mathbf{y}_i \rangle$ is close to an integer, i.e., $\langle \mathbf{s}, \mathbf{y}_i \rangle \approx 0 \pmod 1$. In other words, each $\mathbf{y}_i$ is close to one of the parallel $(n-1)$-dimensional hyperplanes $H_j = \{\mathbf{z} \in \mathbb{R}^n : \langle \mathbf{s}, \mathbf{z} \rangle = j\}, j \in \mathbb{Z}$. Note that this is an *average-case* problem, because the secret $\mathbf{s}$ and points $\mathbf{y}_i$ are chosen at random from some prescribed distributions.

The *unique shortest vector problem* uSVP_γ is defined as follows: let $\mathcal{L} \subset \mathbb{R}^n$ be any lattice having a "γ-unique" shortest vector, which means that the length of a shortest nonzero vector $\mathbf{v} \in \mathcal{L}$ is at least a γ factor smaller than the lengths of all lattice vectors not parallel to $\mathbf{v}$. More concisely: $\lambda_2(\mathcal{L}) \geq \gamma \cdot \lambda_1(\mathcal{L})$, where recall that λ_i denotes the ith successive minimum of the lattice. The goal in uSVP_γ is to find a shortest nonzero vector in $\mathcal{L}$, given an arbitrary basis of $\mathcal{L}$. Note that uSVP is a *promise problem*, because the input lattice $\mathcal{L}$ must satisfy $\lambda_2(\mathcal{L})/\lambda_1(\mathcal{L}) \geq \gamma$. (Alternatively, if it does not satisfy the promise then any answer is considered correct.) It is also a *worst-case* problem, because there is no distribution over the lattice $\mathcal{L}$ or its basis; an algorithm that purports to solve uSVP_γ must work for *any* $\mathcal{L}$ satisfying the promise.

Cryptosystem and security. In brief, the Ajtai-Dwork cryptosystem works as follows: the public and secret keys are respectively a random instance $\{\mathbf{y}_i\}$, and its solution $\mathbf{s}$, of the hidden hyperplanes problem. To encrypt a bit, one generates a ciphertext which is either a "random" point in $\mathbb{R}^n$ (to encrypt a 0), or the sum of a random subset of the points $\{\mathbf{y}_i\}$ given in the public key (to encrypt a 1). The resulting point $\mathbf{y}$ is respectively either "far" from all the hidden hyperplanes H_j, or "close" to one of them. The receiver can distinguish between these two possibilities (and thereby decrypt) using its secret key $\mathbf{s}$, simply by testing whether $\langle \mathbf{s}, \mathbf{y} \rangle$ is close to an integer.

As part of their security proof, Ajtai and Dwork give a *search-to-decision* reduction, which says that any eavesdropper that can distinguish (with any noticeable advantage) between the above two cases can be efficiently converted into an algorithm that solves HHP (with probability very close to one). In other words, breaking the semantic security of the cryptosystem is at least as hard as solving HHP. The second part of their proof shows that any algorithm that solves HHP can be transformed into one that solves uSVP_γ, in the worst case, for some $\gamma = \mathrm{poly}(n)$.

3.2 NTRU

In a concurrent work with Ajtai's in 1996 (but not published until early 1998), Hoffstein, Pipher, and Silverman [HPS98] devised the public-key encryption scheme NTRU (also known as NTRUEncrypt).[1] This was the first cryptographic construction using polynomial rings, which is most usefully interpreted in terms of *algebraically structured* lattices. The NTRU cryptosystem is practically efficient and has quite compact keys, and it has withstood significant cryptanalytic efforts when appropriately parameterized. (Note that early parameterizations were a bit *too* compact, and were shown to have insufficient concrete security; see, e.g., [CS97].) Unlike Ajtai-Dwork and its ilk, however, there is relatively little theoretical understanding of the NTRU cryptosystem and its associated average-case computational problems. In particular, there is no known reduction from any worst-case lattice problem to any standard version of the NTRU problem, nor from the NTRU problem to breaking the cryptosystem's semantic security. (However, a variant of the NTRU cryptosystem has been proved secure [SS11], assuming the hardness of ring-LWE; see Section 5.2.4.)

[1] The meaning of the acronym NTRU is somewhat mysterious; plausible candidates include "Nth degree *tru*ncated polynomial ring" and "Number Theorists 'R' Us."

The NTRU cryptosystem is parameterized by a certain polynomial ring $R = \mathbb{Z}[X]/(f(X))$, e.g., $f(X) = X^n - 1$ for a prime n or $f(X) = X^n + 1$ for an n that is a power of two, and a sufficiently large odd modulus q that defines the quotient ring $R_q = R/qR$. In brief, the public key is $h = 2g \cdot s^{-1} \in R_q$ for two "short" polynomials $g, s \in R$, i.e., ones having relatively small integer coefficients, where the secret key s is also chosen to be invertible modulo both q and two. Encryption essentially involves multiplying h by a short "blinding" factor $r \in R$ and adding a short error term $e \in R$ that encodes the message bits in its coefficients modulo two, to get a ciphertext $c = h \cdot r + e \in R_q$. Decryption is done by multiplying the ciphertext by the secret key to get $c \cdot s = 2g \cdot r + e \cdot s \in R_q$ and interpreting the result as a short element of R, which works because all of g, r, e, and s are short. From this one recovers $e \cdot s$ modulo two, and thereby e modulo two, to recover the message bits. (There are slightly more efficient variants of this basic template, e.g., choosing $s = 1 \pmod 2$, so that $e \cdot s = e \pmod 2$.)

One can define many search and decision problems associated with NTRU, e.g., finding the secret key given the public key, distinguishing the public key from uniformly random, and breaking the scheme's semantic security. Some of these are discussed in further detail in Section 4.4.4.

3.3 Goldreich-Goldwasser-Halevi Encryption and Signatures

Inspired by Ajtai's seminal work [Ajt96] along with McEliece's code-based cryptosystem [McE78], Goldreich, Goldwasser, and Halevi (GGH) [GGH97] proposed a public-key encryption scheme and digital signature scheme based on lattice problems. Unlike the works of Ajtai and Ajtai-Dwork [AD97], the GGH proposals did not come with any worst-case security guarantees; their conjectured security was merely heuristic. Indeed, the GGH encryption scheme was successfully cryptanalyzed for practical parameter sizes (but not broken asymptotically) [Ngu99], and the GGH signature scheme was later broken completely [NR06]. However, the central ideas underlying the GGH proposals were later resurrected and instantiated in ways that admit security proofs under worst-case hardness assumptions, and have subsequently led to an enormous variety of applications (see Sections 5.4 and 5.5 for details).

The main idea behind GGH encryption and signatures is that a public key is a "bad" basis of some lattice, while the corresponding secret key is a "good" basis of the same lattice. Roughly speaking, a "bad" basis is one consisting of long and highly non-orthogonal lattice vectors, while a "good" basis consists of relatively short lattice vectors. Such bases can be generated together, e.g., by first choosing the good basis and then multiplying it by some randomly chosen unimodular transformation (which preserves the lattice) to obtain the bad basis.[2] Alternatively, every integer lattice has a special basis, called the *Hermite normal form*, which is in a precise sense a "hardest possible" basis for the lattice, because it can be efficiently computed from any other basis. So the Hermite normal form is a best-possible choice for the public basis [Mic01].

In the GGH encryption scheme, the sender uses the public key to choose a "random" lattice point $\mathbf{v} \in \mathcal{L}$ that somehow encodes the message, and then adds to it some small error $\mathbf{e} \in \mathbb{R}^n$, letting the ciphertext be $\mathbf{c} = \mathbf{v} + \mathbf{e} \in \mathbb{R}^n$. The error is small enough that $\mathbf{c}$ is much closer to $\mathbf{v}$ than to any other lattice point, so the ciphertext unambiguously represents the message, and recovering $\mathbf{v}$ from $\mathbf{c}$ is a random instance of the *bounded-distance decoding* problem (see Definition 2.2.5). The receiver, using its knowledge of the good basis, can easily decode $\mathbf{c}$ back to $\mathbf{v}$ and recover the message. For security, one may conjecture that an eavesdropper who knows only the bad basis cannot decode $\mathbf{c}$, or even learn anything about $\mathbf{v}$, which implies that the message is hidden.

[2] Such a procedure may be seen as analogous to choosing two random primes as a secret key, and then multiplying them to obtain a public key in factoring-based systems like RSA.

In the GGH signature scheme, a message to be signed is mapped to a point $\mathbf{m} \in \mathbb{R}^n$, e.g., by a suitable public hash function. The signer then uses its good basis to find a lattice vector $\mathbf{v} \in \mathcal{L}$ relatively close to $\mathbf{m}$, which serves as the signature. A verifier, using only the public bad basis, can verify that $\mathbf{v}$ is a lattice vector and is sufficiently close to $\mathbf{m}$. For security, one may conjecture that a forger who knows only the bad basis and some previous message-signature pairs cannot find a lattice vector sufficiently close to $\mathbf{m}'$ for an unsigned message $\mathbf{m}'$. It turns out, however, that this conjecture is *false*, as shown most severely in [NR06]. The main problem is that signatures leak significant information about the geometry of the secret "good" basis, and after a relatively small number of signatures, an adversary can eventually recover the secret basis entirely, allowing it to forge signatures for arbitrary messages.

NTRU meets GGH. Following the ideas in [GGH97], compact ring-based instantiations using NTRU-type lattices were proposed in [HPS01, HHGP+03]. These were subject to various practical attacks, in addition to the generic ones that apply to all GGH-type signatures. Of note is that the second proposal [HHGP+03] includes a "perturbation" technique that is intended to make signatures reveal significantly less information about the secret key (at the cost of larger keys and parameters). The main idea is that the algorithm that decodes the (hashed) message $\mathbf{m} \in \mathbb{R}^n$ to a nearby lattice vector $\mathbf{v} \in \mathcal{L}$ is substantially less linear, because it involves two unrelated lattice bases. However, the ideas of [NR06] were extended to also break this variant, both asymptotically [MPSW09, Wan10] and in practice [DN12b].

3.4 Micciancio's Compact One-Way Function

Inspired by the design ideas and efficiency of NTRU, in work published in 2002, Micciancio [Mic02] modified Ajtai's one-way/collision-resistant function from [Ajt96] to work over polynomial rings of the form $R = \mathbb{Z}[X]/(X^n - 1)$, and demonstrated how this yields major efficiency improvements, namely, quasi-linear $\tilde{O}(n)$ key sizes and runtimes (improving on quasi-quadratic $\tilde{O}(n^2)$). Micciancio also proved that the modified function is one-way, assuming that certain approximation problems on n-dimensional *cyclic* lattices are hard in the worst case.[3]

Unlike for Ajtai's original function, and somewhat curiously, the security proof from [Mic02] showed that the modified function is *one-way*, but not *collision resistant* (a strictly stronger property than one-wayness). Two independent follow-up works [PR06, LM06] (described in Section 4.3.4 below) showed that the function as defined in [Mic02] was in fact *not* collision resistant, but that slightly modifying the construction to work over alternative rings made it so, under the same flavor of worst-case complexity assumptions. We emphasize that all these works were limited to constructing one-way and collision-resistant hash functions; they did not obtain any public-key *encryption* schemes. However, their ideas were important precursors to the development of ring-LWE, which does yield encryption (see Section 4.4 below).

3.5 Regev's Improvements to Ajtai-Dwork

In an important work from 2003, Regev [Reg03] gave several improvements to the results of Ajtai and Dwork [AD97]. Regev's work is most notable for introducing the use of *Gaussian measures* (probability distributions) and *harmonic analysis* over lattices to the design and analysis of lattice-based cryptographic schemes, building upon their use in mathematics by Banaszczyk [Ban93, Ban95]. These techniques yield

[3] An n-dimensional lattice $\mathcal{L}$ is cyclic if $(x_1, x_2, \ldots, x_n) \in \mathcal{L}$ implies $(x_n, x_1, \ldots, x_{n-1}) \in \mathcal{L}$; such a lattice corresponds to an *ideal* in the ring R by identifying polynomial residues of degree less than n with their n-dimensional coefficient vectors.

substantially simpler algorithms and analysis, and much tighter approximation factors for the underlying worst-case lattice problems. More specifically, whereas Ajtai and Dwork showed that breaking their cryptosystem implies being able to solve uSVP_γ for some *large* polynomial $\gamma \approx n^7$ on n-dimensional lattices, for Regev's system the approximation factor is only $\gamma = \tilde{O}(n^{3/2})$. Since uSVP_γ can only become harder as γ decreases, this means that Regev's cryptosystem is provably secure under potentially milder complexity assumptions. In addition, the improved design of the cryptosystem itself withstands known cryptanalytic attacks for somewhat smaller keys, ciphertexts, and runtimes than for Ajtai-Dwork. However, the asymptotic costs of the cryptosystem remain essentially the same, with public keys of size $\tilde{O}(n^4)$, secret keys and ciphertexts of size $\tilde{O}(n^2)$, and corresponding runtimes for encryption and decryption.

Chapter 4

Modern Foundations

In this chapter we survey the main foundational works that directly underlie most modern lattice-based cryptographic schemes. These include the two main average-case problems SIS and LWE, their analogues over rings, and analytical techniques involving (discrete) Gaussian probability distributions.

4.1 Short Integer Solution (SIS)

The *short integer solution* (SIS) problem was first introduced in the seminal work of Ajtai [Ajt96], and has served as the foundation for one-way and collision-resistant hash functions, identification schemes, digital signatures, and other "minicrypt" primitives (but not public-key encryption). Here we define the SIS problem, survey its connection to worst-case lattice problems, and explore some of its basic properties and immediate cryptographic implications. (Further applications are described in Chapters 5 and 6.)

4.1.1 Definitions

Informally, the SIS problem asks, given many uniformly random elements of a certain large finite additive group, to find a sufficiently "short" nontrivial integer combination of them that sums to zero. More formally, SIS is parameterized by positive integers n and q defining the group $\mathbb{Z}_q^n$, a positive real β, and a number m of group elements. (As we shall see, the parameter m is of secondary importance, so we sometimes leave it unspecified.) For concreteness, one should think of n as being the main hardness parameter (e.g., $n \geq 100$), and $q > \beta$ as both being at least a small polynomial in n.

Definition 4.1.1 (Short Integer Solution ($\mathbf{SIS}_{n,q,\beta,m}$)). Given m uniformly random vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$, forming the columns of a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, find a nonzero integer vector $\mathbf{z} \in \mathbb{Z}^m$ of norm $\|\mathbf{z}\| \leq \beta$ such that

$$f_{\mathbf{A}}(\mathbf{z}) := \mathbf{A}\mathbf{z} = \sum_i \mathbf{a}_i \cdot z_i = \mathbf{0} \in \mathbb{Z}_q^n. \tag{4.1.1}$$

We now highlight several simple but useful observations about the SIS problem:

- Without the constraint on $\|\mathbf{z}\|$, it is easy to find a solution via Gaussian elimination. Similarly, we must take $\beta < q$ because otherwise $\mathbf{z} = (q, 0, \ldots, 0) \in \mathbb{Z}^m$ would always be a legitimate (but trivial) solution.

- Any solution for a matrix $\mathbf{A}$ can trivially be converted to one for any extension $[\mathbf{A} \mid \mathbf{A}']$, simply by appending the solution vector with zeros (which leaves the norm of the solution unchanged). In other words, we can ignore columns $\mathbf{a}_i$ as desired, so the SIS problem can only become easier as m increases. Similarly, it can only become harder as n increases.

- The norm bound β and the number m of vectors $\mathbf{a}_i$ must be large enough that a solution is guaranteed to exist. This is the case whenever $\beta \geq \sqrt{\bar{m}}$ and $m \geq \bar{m}$, where $\bar{m}$ is the smallest integer greater than $n \log q$, by a pigeonhole argument: first, by the previous observation we can assume without loss of generality that $m = \bar{m}$. Then because there are more than q^n vectors $\mathbf{x} \in \{0,1\}^m$, there must be two distinct $\mathbf{x}, \mathbf{x}'$ such that $\mathbf{A}\mathbf{x} = \mathbf{A}\mathbf{x}' \in \mathbb{Z}_q^n$, so their difference $\mathbf{z} = \mathbf{x} - \mathbf{x}' \in \{0, \pm 1\}^m$ is a solution of norm at most β.

- The above pigeonhole argument in fact shows more: the induced function family $\{f_{\mathbf{A}} \colon \{0,1\}^m \to \mathbb{Z}_q^n\}$ defined in Equation (4.1.1) is *collision resistant*, assuming the hardness of the corresponding SIS problem. This is because a collision $\mathbf{x}, \mathbf{x}' \in \{0,1\}^m$ for $f_{\mathbf{A}}$ immediately yields an SIS solution for $\mathbf{A}$. (Of course the domain $\{0,1\}^m$ is somewhat arbitrary here, and can be replaced by essentially any other large enough set of sufficiently short integer vectors.)

The SIS problem can be seen as an *average-case* short-vector problem on a certain family of so-called "q-ary" m-dimensional integer lattices, namely, the lattices

$$\mathcal{L}^{\perp}(\mathbf{A}) := \{\mathbf{z} \in \mathbb{Z}^m : \mathbf{A}\mathbf{z} = \mathbf{0} \in \mathbb{Z}_q^n\} \supseteq q\mathbb{Z}^m. \tag{4.1.2}$$

Borrowing the terminology of coding theory, here $\mathbf{A}$ acts as a "parity-check" (or more accurately, "arity-check") matrix that defines the lattice $\mathcal{L}^{\perp}(\mathbf{A})$. The SIS problem asks to find a sufficiently short nonzero vector in $\mathcal{L}^{\perp}(\mathbf{A})$, where $\mathbf{A}$ is chosen uniformly at random.

One can also consider an *inhomogeneous* version of the SIS problem, which is to find a short integer solution to $\mathbf{A}\mathbf{x} = \mathbf{u} \in \mathbb{Z}_q^n$, where $\mathbf{A}, \mathbf{u}$ are uniformly random and independent. Notice that, disregarding the norm constraint, the set of all solutions is the lattice coset $\mathcal{L}^{\perp}_{\mathbf{u}}(\mathbf{A}) := \mathbf{c} + \mathcal{L}^{\perp}(\mathbf{A})$, where $\mathbf{c} \in \mathbb{Z}^m$ is an arbitrary (not necessarily short) solution. It is not hard to show that the homogeneous and inhomogeneous problems are essentially equivalent for typical parameters.

Normal form. The SIS problem admits a small but important optimization, called the (Hermite) "*normal form*" (HNF), which compresses the size of the instance $\mathbf{A}$ by n columns, at no cost in cryptographic functionality or hardness.[1] It works as follows: first, we can assume that the leftmost n columns $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ of $\mathbf{A} = [\mathbf{A}_1 \mid \mathbf{A}_2] \in \mathbb{Z}_q^{n \times m}$ form an invertible matrix over $\mathbb{Z}_q$, which is without loss of generality because, as observed above, we can ignore columns as desired. We then replace $\mathbf{A}$ with

$$\mathbf{A}_1^{-1} \cdot \mathbf{A} = [\mathbf{I}_n \mid \bar{\mathbf{A}} = \mathbf{A}_1^{-1}\mathbf{A}_2],$$

and treat the $\mathbf{I}_n$ submatrix as implicit. Note that $\bar{\mathbf{A}}$ is uniformly random, because $\mathbf{A}_2$ is uniform and independent of $\mathbf{A}_1$. Moreover, $\mathbf{A}$ and $[\mathbf{I}_n \mid \bar{\mathbf{A}}]$ have exactly the same set of (short) SIS solutions. Therefore, SIS instances of the latter form are at least as hard to solve as those of the former type.

[1] The HNF optimization for SIS is analogous to working with a *systematic* generator matrix of an error-correcting code. Its formal connection to the Hermite normal form for lattices is explained in [MR09, end of Section 5].

4.1.2 Hardness

Starting from Ajtai's seminal work [Ajt96], a long sequence of works has established progressively stronger results about the hardness of the SIS problem relative to worst-case lattice problems. All such results are instances of the following template:

Theorem 4.1.2. *For any $m = \mathrm{poly}(n)$, any $\beta > 0$, and any sufficiently large $q \geq \beta \cdot \mathrm{poly}(n)$, solving $\mathrm{SIS}_{n,q,\beta,m}$ with non-negligible probability is at least as hard as solving the decisional approximate shortest vector problem GapSVP_γ and the approximate shortest independent vectors problems SIVP_γ (among others) on* arbitrary *n-dimensional lattices (i.e., in the worst case) with overwhelming probability, for some $\gamma = \beta \cdot \mathrm{poly}(n)$.*

Notice that the exact values of m and q (apart from its lower bound) play essentially no role in the ultimate hardness guarantee, but that the approximation factor γ degrades with the norm bound β on the SIS solution. The theorem is proved by giving a polynomial-time *reduction* that uses an oracle for SIS (which works on the average, with noticeable probability) to solve GapSVP_γ and SIVP_γ on any n-dimensional lattice. The central challenge in obtaining such a reduction is in generating a *uniformly random* SIS instance whose solution somehow helps in finding short vectors of an *arbitrary* lattice; below we give a high-level description of how this is done.

In Ajtai's original work, the $\mathrm{poly}(n)$ factors associated with the modulus q and approximation factor γ were quite large polynomials. Since then, several works have substantially improved these factors, thus yielding smaller instances (e.g., cryptographic keys) and stronger hardness guarantees. A few of the most notable improvements are as follows:

- The 2004 work of Micciancio and Regev [MR04] obtained approximation factors of $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$, which can be as small as $\gamma = \tilde{O}(n)$ for meaningful choices of β, with a modulus q that can be as small as $\beta \cdot \tilde{O}(n\sqrt{m})$.

  This work uses Gaussians over lattices and builds upon the harmonic analysis techniques first developed in [Ban93, Reg03, AR04]. In particular, it defines and analyzes the important notion of the lattice *smoothing parameter* $\eta(\mathcal{L})$ (already implicit in [Ban93, Reg03]), which is the amount of Gaussian error needed to "smooth out" the discrete structure of a lattice. Such smoothing is what lets the reduction generate uniformly random SIS instances that are meaningfully linked to an arbitrary input lattice.

- The 2008 work of Gentry, Peikert, and Vaikuntanathan improved the bound on q to be as small as $\beta \cdot \tilde{O}(\sqrt{n})$, while preserving the $\gamma = \beta \cdot \tilde{O}(\sqrt{n})$ approximation factor from [MR04]. The main new ingredient is a technically simpler reduction that works entirely with *discrete* Gaussians over lattices, which avoids the "round-off" error associated with continuous Gaussians. To do this the reduction uses GPV's discrete Gaussian sampling algorithm, which is described below in Theorem 5.4.2.

- The 2013 work of Micciancio and Peikert [MP13] further improved the bound on q to be as small as $\beta \cdot n^\varepsilon$ for any constant $\varepsilon > 0$. Recall that this bound is essentially optimal (up to the n^ε factor), because any SIS instance has trivial solutions of norm q. The approximation factor γ obtained by the reduction is somewhat subtle, as it can depend on the norm of the SIS solution in the ℓ_∞ norm, rather than the usual ℓ_2 norm. This is due to the reduction's use of the SIS oracle to produce a discrete Gaussian as a combination of several other discrete Gaussians, using a "convolution lemma" similar to one proved in [Pei10].

Overview of the reductions. The worst-case/average case reductions that prove Theorem 4.1.2 all work according to the following template, first due to Ajtai [Ajt96]. Recall that the reduction is given a basis $\mathbf{B}$ of an arbitrary n-dimensional lattice $\mathcal{L}$, along with an average-case SIS oracle, and its goal is to solve SIVP_γ for some $\gamma = \mathrm{poly}(n)$, i.e., to find n linearly independent lattice vectors all of length at most $\gamma \cdot \lambda_n(\mathcal{L})$. Here we give an informal overview of the reduction strategy and its analysis; see, e.g., [MR04, GPV08, MP13] for the full details of modern instantiations.

At the highest level, the reduction uses a set $\mathbf{S} \subset \mathcal{L}$ of linearly independent lattice vectors (starting with the input basis $\mathbf{B}$), along with its SIS oracle, to obtain a new set $\mathbf{S}' \subset \mathcal{L}$ such that $\|\mathbf{S}'\| \leq \|\mathbf{S}\|/2$, where $\|\mathbf{X}\| := \max_i \|\mathbf{x}_i\|$ for a set $\mathbf{X} = \{\mathbf{x}_i\}$. It then iteratively repeats this process until it stops working, at which point we can guarantee that the final set is indeed a solution to SIVP_γ.

To implement the above strategy, the reduction uses the following "core step:"

1. Using the current set $\mathbf{S}$ of lattice vectors, generate m *random* lattice vectors $\mathbf{v}_i \in \mathcal{L}$ that are "well spread" yet as short as is feasible.[2] Let these vectors form the columns of a matrix $\mathbf{V}$.
2. Define $\mathbf{a}_i \in \mathbb{Z}_q^n$ as $\mathbf{a}_i := \mathbf{B}^{-1}\mathbf{v}_i \bmod q\mathbb{Z}^n$, i.e., $\mathbf{A} := \mathbf{B}^{-1} \cdot \mathbf{V} \bmod q\mathbb{Z}^{n\times m}$, and give this instance to the SIS oracle. (Note that $\mathbf{B}^{-1}\mathbf{v}_i$ is integral because $\mathbf{v}_i$ is a lattice vector.)
3. If the oracle returns a valid solution $\mathbf{z} \in \mathbb{Z}^m$, output $\mathbf{v} := \mathbf{V}\mathbf{z}/q$.

By repeating this core step enough times, the reduction can obtain many vectors $\mathbf{v}$, (some of) which will make up the next set $\mathbf{S}'$. In order for all this to work as desired, one needs to prove the following:

- *If $\mathbf{z}$ is an SIS solution for $\mathbf{A}$, then $\mathbf{v} \in \mathcal{L}$ and $\|\mathbf{v}\| \leq \|\mathbf{S}\|/2$.* The first claim follows by construction: because $\mathbf{v}_i = \mathbf{B}\mathbf{a}_i \pmod{q\mathcal{L}}$, we have

  $$\mathbf{V}\mathbf{z} = \mathbf{B}(\mathbf{A}\mathbf{z}) = \mathbf{0} \pmod{q\mathcal{L}},$$

  so $\mathbf{v} = \mathbf{V}\mathbf{z}/q \in \mathcal{L}$. For the second claim, we can generate the vectors $\mathbf{v}_i$ such that $\|\mathbf{v}_i\| \leq \|\mathbf{S}\| \cdot \mathrm{poly}(n)$. Then because $\|\mathbf{z}\| \leq \beta$, we have $\|\mathbf{V}\mathbf{z}\| \leq \|\mathbf{S}\| \cdot \beta \cdot \mathrm{poly}(n)$. Therefore, for a sufficiently large $q = \beta \cdot \mathrm{poly}(n)$ we can ensure that $\|\mathbf{v}\| = \|\mathbf{V}\mathbf{z}\|/q \leq \|\mathbf{S}\|/2$.
- *The SIS instance $\mathbf{A} \in \mathbb{Z}_q^{n\times m}$ is (nearly) uniform.* Because the SIS oracle is only assumed to work (with noticeable probability) for *uniformly random* instances, we need to show that this is the case (up to negligible statistical error) for the instances constructed by the reduction. This holds if $\|\mathbf{S}\| > \gamma \cdot \lambda_n(\mathcal{L})$—i.e., if we don't already have an SIVP_γ solution—essentially because the $\mathbf{v}_i$ are sufficiently "well spread" modulo $q\mathcal{L}$.

  More specifically, for an appropriate distribution of $\mathbf{v}_i$ one can use the *smoothing parameter* (see Section 2.3) to prove that $\mathbf{v}_i \bmod q\mathcal{L}$ is nearly uniform over $\mathcal{L}/q\mathcal{L}$, because $\gamma \geq \beta \cdot \mathrm{poly}(n) \geq q$.[3] Finally, because multiplication by $\mathbf{B}^{-1}$ is a bijection from $\mathcal{L}/q\mathcal{L}$ to $\mathbb{Z}^n/q\mathbb{Z}^n$, each $\mathbf{a}_i = \mathbf{B}^{-1}\mathbf{v}_i \bmod q\mathbb{Z}^n$ is nearly uniform as well.
- *Some subset of the vectors $\mathbf{v}$ is full rank (i.e., the $\mathbf{v}$ do not all lie in a proper subspace of $\mathbb{R}^n$).* Finally, we must show that even a *malicious* SIS oracle cannot, e.g., force all the vectors $\mathbf{v}$ to be zero. This again follows from the fact that the $\mathbf{v}_i$ are "well spread," but in a slightly different way. Specifically, *conditioned on* any fixed value of $\mathbf{a}_i \in \mathbb{Z}_q^n$, which is all the oracle "sees" about $\mathbf{v}_i$, it remains well

[2] Concretely, we can use $\mathbf{S}$ to sample from a discrete Gaussian distribution over $\mathcal{L}$; see Section 2.3 and Theorem 5.4.2.

[3] In fact, we do not actually need an upper bound on q here: if $q > \gamma$, we can just sample the $\mathbf{v}_i$ from a distribution that is wider by a q/γ factor; the effect of this is exactly cancelled out when we divide by q in the end.

spread over the coset $\mathbf{B}\mathbf{a}_i \in \mathcal{L}/q\mathcal{L}$. Therefore, conditioned on any fixed SIS instance $\mathbf{A}$ and solution $\mathbf{z}$, the random variable $\mathbf{V}\mathbf{z}$ is still sufficiently well spread that any proper subspace of $\mathbb{R}^n$ lacks a noticeable amount of its probability mass, and so by obtaining enough vectors $\mathbf{v}$ we can obtain a full-rank set.

4.2 Learning With Errors (LWE)

A very important work of Regev [Reg05] from 2005 introduced the average-case *learning with errors* (LWE) problem, which is the "encryption-enabling" analogue of the SIS problem. Indeed, the two problems are syntactically very similar, and can meaningfully be seen as duals of each other. Here we describe LWE, its hardness, and a basic LWE-based cryptosystem in some detail. See Chapters 5 and 6 for a survey of LWE's many other cryptographic applications, and Regev's survey [Reg10] for more on its worst-case hardness, search/decision equivalence, and some basic applications.

4.2.1 Definitions

LWE is parameterized by positive integers n and q, and an error distribution χ over $\mathbb{Z}$. For concreteness, n and q can be thought of as roughly the same as in SIS, and χ is usually taken to be a discrete Gaussian of width αq for some $\alpha < 1$, which is often called the relative "error rate."[4]

Definition 4.2.1 (LWE distribution). For a vector $\mathbf{s} \in \mathbb{Z}_q^n$ called the *secret*, the LWE distribution $A_{\mathbf{s},\chi}$ over $\mathbb{Z}_q^n \times \mathbb{Z}_q$ is sampled by choosing $\mathbf{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \leftarrow \chi$, and outputting $(\mathbf{a}, b = \langle \mathbf{s}, \mathbf{a} \rangle + e \bmod q)$.[5]

There are two main versions of the LWE problem: *search*, which is to find the secret given LWE samples, and *decision*, which is to distinguish between LWE samples and uniformly random ones. We additionally parameterize these problems by the number m of available samples, which we typically take to be large enough that the secret is uniquely defined with high probability. (As with SIS, the parameter m is of secondary importance, so we often leave it unspecified.)

Definition 4.2.2 (Search-LWE$_{n,q,\chi,m}$). Given m independent samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ drawn from $A_{s,\chi}$ for a uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$ (fixed for all samples), find $\mathbf{s}$.

Definition 4.2.3 (Decision-LWE$_{n,q,\chi,m}$). Given m independent samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^n \times \mathbb{Z}_q$ where every sample is distributed according to either: (1) $A_{\mathbf{s},\chi}$ for a uniformly random $\mathbf{s} \in \mathbb{Z}_q^n$ (fixed for all samples), or (2) the uniform distribution, distinguish which is the case (with non-negligible advantage).

We highlight several useful observations about search- and decision-LWE:

- Without the error terms from χ, both problems are easy to solve, because we can efficiently recover $\mathbf{s}$ from LWE samples by Gaussian elimination. (In the uniform case of decision-LWE, with high probability no solution $\mathbf{s}$ will exist.)

[4] The original work of Regev [Reg05] considered a *continuous* Gaussian distribution, then discretized it by rounding to the nearest integer. While this does not quite yield a true discrete Gaussian, a special case of a theorem from [Pei10] says that a slightly different, randomized discretization method does so.

[5] It is worth mentioning that LWE is a generalization of "learning parities with noise," which is the special case where $q = 2$ and χ is a Bernoulli distribution over $\{0, 1\}$.

- Just as with SIS, it is often convenient to combine the given samples into a matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ (whose columns are the vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$) and a vector $\mathbf{b} \in \mathbb{Z}_q^m$ (whose entries are the $b_i \in \mathbb{Z}_q$), so that for LWE samples we have[6]
$$\mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t \pmod q,$$
where $\mathbf{e} \leftarrow \chi^m$. In the uniform case of decision-LWE, $\mathbf{b}$ is uniformly random and independent of $\mathbf{A}$.

- Search-LWE can be seen as an *average-case* bounded-distance decoding (BDD) problem on a certain family of q-ary m-dimensional integer lattices: for LWE samples, the vector $\mathbf{b}$ is relatively close to exactly one vector in the LWE lattice
$$\mathcal{L}(\mathbf{A}) := \{\mathbf{A}^t\mathbf{s} : \mathbf{s} \in \mathbb{Z}_q^n\} + q\mathbb{Z}^m, \tag{4.2.1}$$
and the goal is to find that lattice vector. (In the uniform case, $\mathbf{b}$ is far from all points in $\mathcal{L}(\mathbf{A})$ with very high probability.) Also, it is not hard to verify that the SIS and LWE lattices respectively defined in Equations (4.1.2) and (4.2.1) are *dual* to each other, up to a scaling factor of q, i.e., $\mathcal{L}(\mathbf{A}) = q \cdot \mathcal{L}^\perp(\mathbf{A})^*$.

- There is a high-level similarity between LWE and the Ajtai-Dwork hidden hyperplanes problem (described in Section 3.1): in LWE there is a secret vector $\mathbf{s}$, and the samples $(\mathbf{a}_i, b_i) \in \mathbb{Z}_q^{n+1}$ are either uniformly random, or are close to the mod-q subspace orthogonal to $(-\mathbf{s}, 1)$, because $\langle(-\mathbf{s}, 1), (\mathbf{a}_i, b_i)\rangle = e_i \approx 0 \bmod q$. (A more formal connection is given in Regev's survey [Reg10].)

Normal form. Similarly to SIS, the LWE problem also has a normal form, in which the coordinates of the secret $\mathbf{s}$ are chosen independently from the *error distribution* χ (modulo q). Using this form can yield substantial efficiency gains for certain cryptographic constructions, as discussed below in Section 5.2. Applebaum *et al.* [ACPS09] (following [MR09]) proved that the normal form, in either its search or decision variant, is at least as hard as the same variant for *any* distribution of the secret (e.g., uniform), up to a small difference in the number of samples m. The reduction that proves this is a slight extension of the one described for SIS in Section 4.1: given an instance
$$\mathbf{A} = [\mathbf{A}_1 \mid \mathbf{A}_2], \quad \mathbf{b}^t = [\mathbf{b}_1^t \mid \mathbf{b}_2^t]$$
where $\mathbf{A}_1 \in \mathbb{Z}_q^{n \times n}$ is invertible and $\mathbf{b}_1 \in \mathbb{Z}_q^n$, we transform it to the instance
$$\bar{\mathbf{A}} = -\mathbf{A}_1^{-1} \cdot \mathbf{A}_2, \quad \bar{\mathbf{b}}^t = \mathbf{b}_1^t\bar{\mathbf{A}} + \mathbf{b}_2^t.$$
As before, $\bar{\mathbf{A}}$ is uniformly random, because $\mathbf{A}_2$ is independent of $\mathbf{A}_1$. Now observe that when the input comes from an LWE distribution, each $\mathbf{b}_i^t = \mathbf{s}^t\mathbf{A}_i + \mathbf{e}_i^t$ for some $\mathbf{s} \in \mathbb{Z}_q^n$, where the entries of each $\mathbf{e}_i$ are independently drawn from χ, so we have
$$\begin{aligned}\bar{\mathbf{b}}^t &= \mathbf{s}^t\mathbf{A}_1 \cdot \bar{\mathbf{A}} + \mathbf{e}_1^t\bar{\mathbf{A}} + \mathbf{s}^t\mathbf{A}_2 + \mathbf{e}_2^t \\ &= \mathbf{e}_1^t\bar{\mathbf{A}} + \mathbf{e}_2^t.\end{aligned}$$
That is, the instance $\bar{\mathbf{A}}, \bar{\mathbf{b}}$ comes from the LWE distribution with secret $\mathbf{e}_1$, and finding $\mathbf{e}_1$ yields the original secret $\mathbf{s}^t = (\mathbf{b}_1^t - \mathbf{e}_1^t) \cdot \mathbf{A}_1^{-1}$. Also, for the decision problem, when $\mathbf{b}$ is uniformly random and independent of $\mathbf{A}$, it immediately follows that $\bar{\mathbf{b}}$ is uniformly random and independent of $\bar{\mathbf{A}}$.

[6] A useful convention for later cryptographic schemes is to multiply secrets on the right for SIS, and on the left for LWE.

Using the normal form, LWE can even be seen as syntactically identical to the normal form of SIS, but for parameters that correspond to *injective* rather than *surjective* functions: while SIS is concerned with the surjective function mapping a short $\mathbf{x} \in \mathbb{Z}^m$ to $[\mathbf{I}_n \mid \bar{\mathbf{A}}] \cdot \mathbf{x} \in \mathbb{Z}_q^n$, LWE deals with the injective function mapping a short $\mathbf{e} \in \mathbb{Z}^m$ to $[\bar{\mathbf{A}}^t \mid \mathbf{I}_{m-n}] \cdot \mathbf{e} \in \mathbb{Z}_q^{m-n}$. (For further details see [Mic10].) However, there is a major qualitative difference between injectivity and surjectivity, in terms of what cryptographic objects we can construct, and what we can prove about their hardness.

4.2.2 Hardness

Regev proved the following worst-case hardness theorem for LWE (stated here in a slightly stronger form, as discussed below):

Theorem 4.2.4 ([Reg05]). *For any $m = \mathrm{poly}(n)$, any modulus $q \leq 2^{\mathrm{poly}(n)}$, and any (discretized) Gaussian error distribution χ of parameter $\alpha q \geq 2\sqrt{n}$ where $0 < \alpha < 1$, solving the decision-LWE$_{n,q,\chi,m}$ problem is at least as hard as* quantumly *solving* GapSVP_γ *and* SIVP_γ *on* arbitrary *n-dimensional lattices, for some $\gamma = \tilde{O}(n/\alpha)$.*

Notice that, just as in the worst-case hardness theorem for SIS (Theorem 4.1.2), the exact values of m and q (apart from its lower bound of $2\sqrt{n}/\alpha$) play essentially no role in the ultimate hardness guarantee. However, the approximation factor γ degrades with the inverse error rate $1/\alpha$ of the LWE problem.

Theorem 4.2.4 is proved by giving a *quantum* polynomial-time reduction that uses an oracle for LWE to solve GapSVP_γ and SIVP_γ in the worst case, thereby transforming any algorithm (whether classical or quantum) that solves LWE into a quantum algorithm for lattice problems. The quantum nature of the reduction is meaningful because there are no known quantum algorithms for GapSVP_γ or SIVP_γ that significantly outperform classical ones, beyond generic quantum speedups. Still, it would be very useful to have a completely *classical* reduction to give further confidence in the hardness of LWE. Such a reduction was given in 2009 by Peikert [Pei09], and is discussed below in Section 4.2.4.

In [Reg05], the above theorem is proved in two main parts:

1. First, *search*-LWE is proved to be at least as hard as worst-case lattice problems, via a *quantum* reduction. This reduction consists of two main sub-parts:

   (a) Using an oracle for search-LWE, along with a source of discrete Gaussian samples over $\mathcal{L}$ with parameter r, one can *classically* solve bounded-distance decoding BDD on the dual lattice $\mathcal{L}^*$ to within distance $d \approx \alpha q/r$. This works by combining the BDD instance with the Gaussian samples to produce properly distributed LWE samples, whose underlying secret (which the oracle reveals) lets us compute the BDD solution.

   (b) Using an oracle for BDD on $\mathcal{L}^*$ to within distance d, one can *quantumly* generate discrete Gaussian samples over $\mathcal{L}$ with parameter $r' \approx \sqrt{n}/d$. This uses quantum computation to "uncompute" a known solution to a BDD instance, which lets us to set up a particular quantum state. Computing the quantum Fourier transform on this state and measuring yields a discrete Gaussian sample.

   Notice that for the value of d from the first sub-part, in the second sub-part we have $r' \leq r/\sqrt{2}$ because $\alpha q \geq 2\sqrt{n}$. The full reduction iterates the two steps, generating discrete Gaussian samples of successively narrower parameter until the steps stop working, at which point we have very narrow discrete Gaussian samples over $\mathcal{L}$, which easily yield solutions to SIVP and GapSVP.

2. Second, *decision*-LWE is proved to be equivalent to *search*-LWE (up to some polynomial blowup in the number m of samples), via an elementary *classical* reduction. Originally, this equivalence applied only to polynomially-bounded prime moduli $q = \mathrm{poly}(n)$, generalizing an earlier proof for the case $q = 2$ [BFKL93]. Subsequently, it has been improved to hold for essentially any modulus [Pei09, ACPS09, MM11, MP12, BLP+13], i.e., even exponentially large composite ones.

4.2.3 Cryptosystem

In [Reg05], Regev also gave a new public-key cryptosystem whose semantic security can provably be based on the LWE problem with an error rate of $\alpha = \tilde{\Omega}(1/\sqrt{n})$, and hence on the conjectured quantum hardness of GapSVP_γ or SIVP_γ for $\gamma = \tilde{O}(n^{3/2})$. In contrast with Ajtai-Dwork [AD97] and Regev's improvements to it [Reg03], two notable features of the LWE-based cryptosystem are:

1. *generality*: the underlying worst-case problems GapSVP_γ and SIVP_γ appear less "structured" than the unique-SVP problem uSVP_γ, and
2. *improved efficiency*: public keys are only $\tilde{O}(n^2)$ bits (versus $\tilde{O}(n^4)$), and secret keys and ciphertexts are only $\tilde{O}(n)$ bits (versus $\tilde{O}(n^2)$) per encrypted message bit.

In addition, in the multi-user setting the per-user public keys can be reduced to $\tilde{O}(n)$ bits, using $\tilde{O}(n^2)$ bits of trusted randomness that is shared across all users.[7]

At a high level, the construction and security proof for Regev's LWE-based cryptosystem are strongly reminiscent of Ajtai-Dwork: the secret key is a random vector $\mathbf{s} \in \mathbb{Z}_q^n$, and the public key is several LWE samples $(\mathbf{a}_i, b_i)$ for secret $\mathbf{s}$. One encrypts a bit by adding a random subset of the samples in the public key, then suitably hiding the message bit in the last coordinate of the result, so that the ciphertext is either "close to" or "far from" the subspace orthogonal to $(-\mathbf{s}, 1)$. Using the secret $\mathbf{s}$, one can decrypt by distinguishing between the two cases. Semantic security follows by considering a thought experiment in which the public key is "malformed," in the sense that it consists of uniformly random samples with no underlying secret. Assuming the hardness of LWE, no adversary can distinguish such a key from a properly formed one. Moreover, encryption under such a malformed key is "lossy," in the sense that the resulting ciphertext is *statistically independent* of the message bit, so an adversary has no advantage in distinguishing an encryption of 0 from an encryption of 1. More technical details on the construction and proof are given in Section 5.2.

4.2.4 More Hardness

Following [Reg05], several works provided additional hardness theorems for LWE, e.g., under classical reductions, for "leaky" secrets, for smaller errors and moduli, etc. We review many of these results below.

Classical hardness. A work of Peikert [Pei09] partially "dequantized" Regev's quantum worst-case reduction [Reg05] from Theorem 4.2.4 above. In particular, this was the first work to yield public-key encryption assuming the worst-case *classical* hardness of a problem on *general* lattices, as opposed to structured lattices with "unique" shortest vectors, as in [AD97, Reg03]. More specifically, Peikert proved that LWE with an error rate of α is classically at least as hard as worst-case GapSVP_γ, for the same $\gamma = \tilde{O}(n/\alpha)$ factor as in Theorem 4.4.3. However, there are two main caveats:

[7] It is not clear how to implement such sharing for [AD97, Reg03], and while Ajtai [Ajt05] later gave a different style of cryptosystem that does permit sharing, no worst-case security proof is known for it.

- The classical reduction works only for GapSVP, not SIVP, whereas the quantum reduction works for both problems.
- The reduction requires an exponentially large modulus $q \geq 2^{n/2}$ for the LWE problem, whereas the quantum reduction works for any modulus $q \geq 2\sqrt{n}/\alpha$. (The classical reduction can be adapted to work for moduli as small as $q = \mathrm{poly}(n)$, but for a non-standard variant of GapSVP.)

A large modulus means that LWE samples require many more bits to represent, and thereby implies larger key sizes and less-efficient cryptoschemes. However, it does not appear to restrict the *kinds* of cryptographic applications that can be constructed from LWE.

Shortly following [Pei09], Lyubashevsky and Micciancio [LM09] built upon its main idea to prove the equivalence, under classical reductions and up to small $\mathrm{poly}(n)$ approximation factors, of the GapSVP, uSVP, and BDD problems. In particular, this implies that the Ajtai-Dwork cryptosystem, which was originally based on uSVP, can actually be based on GapSVP as well.

A few years later, Brakerski *et al.* [BLP+13] gave a general dimension-modulus tradeoff for LWE, which roughly says that hardness for a particular error rate α is determined almost entirely by $n \log q$, and not by the particular choices of n and q, as long as q is bounded from below by some small polynomial. So for example, using Peikert's result that GapSVP on n-dimensional lattices classically reduces to LWE in dimension n with modulus $q \geq 2^{n/2}$, by [BLP+13] the same GapSVP problem classically reduces to LWE in dimension n^2 with modulus $q = \mathrm{poly}(n)$. The reductions from [BLP+13] were heavily influenced by techniques like "key-switching" and "modulus reduction" that were developed in the literature on fully homomorphic encryption (e.g., [BV11b, BGV12, Bra12]; see Section 6.1), but are more technically intricate due to the need to generate the proper LWE distribution rather than just samples with small error terms.

Robustness. LWE is a very "robust" problem, in the sense that it remains hard even if the attacker learns extra information about the secret and errors. For example, Goldwasser *et al.* [GKPV10] showed that LWE with "weak" secrets—i.e., where the adversary learns some bounded information about, or a hard-to-invert function of, the secret—remains as hard as LWE with "perfect" secrets, although for smaller dimension n and error rate α. They used this fact to give a symmetric-key encryption scheme that is robust to imperfect or leaky secret keys. Similarly, Dodis *et al.* [DGK+10] gave a public-key encryption scheme from LWE that is robust to leakage of any computationally hard-to-invert function of the secret key. Finally, several works [BF11, OPW11, AP12, BLP+13, LPSS14] give comparatively tight reductions showing that LWE remains hard, up to a small loss in the dimension and error rate, even if one reveals one or more linear relations (over the integers) on the secret and error.

Alternative errors and small parameters. In independent works, Döttling and Müller-Quade [DM13] and Micciancio and Peikert [MP13] also considered LWE with non-Gaussian and potentially small errors, e.g., uniform over an interval. Such distributions are algorithmically much easier to sample than Gaussians, and so may be more suitable in practical implementations. Another motivation is an algorithmic attack of Arora and Ge [AG11], which shows that for errors that come from a domain of size d, it is possible to solve LWE in time and space roughly 2^{d^2}, provided that the attacker is given sufficiently many LWE samples. Although the precise statements of the results from [DM13, MP13] are almost entirely disjoint, their qualitative flavor and techniques are very similar. For concreteness, we give a few details about the latter.

The work of [MP13] shows that LWE remains hard for non-Gaussian error distributions (e.g., uniform) that may also have small support (even as small as $\{0, 1\}$), provided that the number m of samples available to the attacker is appropriately bounded. Note that for small support, some sample bound is necessary in

light of the Arora-Ge attack [AG11] mentioned above. For example, for binary errors the number of samples is limited to $n(1 + O(1/\log n))$, while for supports of size $\Omega(n)$ the number of samples can be as large as $O(n)$. While the former sample bound is not large enough to support any known cryptographic application of LWE, the latter is large enough for certain uses. The main theorem from [MP13] relies on the assumed hardness of standard LWE for large enough Gaussian errors, and in somewhat larger dimensions. It uses a "lossiness" argument, very similar to the main idea from [PW08], which shows that search-LWE with small errors is *information-theoretically* unsolvable if the samples $\mathbf{a}_i$ are generated according to the underlying (standard) LWE distribution, rather than uniformly at random. Since the standard LWE distribution is indistinguishable from uniform, one cannot break the small-error version of LWE without also breaking standard decision-LWE.

4.3 Ring-SIS

Recall from Section 3.4 that, inspired by the ideas behind the NTRU cryptosystem [HPS98], Micciancio [Mic02] introduced a compact ring-based analogue of Ajtai's SIS problem and its associated function $f_{\mathbf{A}}$ from Definition 4.1.1. This analogue has come to be known as the *ring-SIS* problem. Here we define the problem, describe its connection with SIS, and survey its hardness relative to worst-case problems on *ideal lattices* in the underlying ring.

4.3.1 Definitions

The ring-SIS problem is parameterized by:

- A ring R, which is often (but not always) taken to be a degree-n polynomial ring of the form $R = \mathbb{Z}[X]/(f(X))$, e.g., $f(X) = X^n - 1$ as in [Mic02], or $f(X) = X^{2^k} + 1$ as in [LMPR08]. Note that elements of R can be canonically represented by their residues modulo $f(X)$, which are integer polynomials of degree less than n.

  We also endow R with a norm $\|\cdot\|$, which is not necessarily the norm of the argument's vector of coefficients; see Section 4.3.3 for further details. For a vector $\vec{z}$ over R we define $\|\vec{z}\| = (\sum_i \|z_i\|^2)^{1/2}$.

- A positive integer modulus q. We define $R_q := R/qR = \mathbb{Z}_q[X]/(f(X))$, whose canonical representatives are polynomials of degree less than n with coefficients from some set of canonical representatives of $\mathbb{Z}_q$.

- A real norm bound $\beta > 0$ for the "short" solution, and a number m of samples. (As usual, m tends to be of secondary importance, so we often leave it unspecified.)

For concreteness, the degree n, modulus q, and norm bound β can be thought of as roughly comparable to their counterparts in the SIS problem, whereas m is typically an n factor smaller for ring-SIS (as explained below).

Definition 4.3.1 (R-SIS$_{q,\beta,m}$). Given m uniformly random elements $a_i \in R_q$, defining a vector $\vec{a} \in R_q^m$, find a nonzero vector $\vec{z} \in R^m$ of norm $\|\vec{z}\| \leq \beta$ such that

$$f_{\vec{a}}(\vec{z}) := \langle \vec{a}, \vec{z} \rangle = \vec{a}^t \cdot \vec{z} = \sum_i a_i \cdot z_i = 0 \in R_q. \tag{4.3.1}$$

The primary advantage of R-SIS over SIS is its relative compactness and efficiency: the number m of elements $a_i \in R_q$ required to guarantee the existence of a sufficiently short solution is only $m \approx \log q$, rather than $m \approx n \log q$ for SIS. This is essentially because there are an exponential $2^{\Omega(n)}$ number of short ring elements $z_i \in R$ that can be used as coefficients for each $a_i \in R_q$, versus just a few small *integer* coefficients for each $\mathbf{a}_i \in \mathbb{Z}_q^n$ in the SIS problem. In addition, using FFT-like techniques one can compute each $z_i \cdot a_i \in R_q$ in quasi-linear $\tilde{O}(n)$ time, so the total time to compute $f_{\vec{a}}(\vec{z})$ is also quasi-linear for typical choices of q and m.

4.3.2 Relation to SIS

It is helpful to understand the formal algebraic similarities and differences between SIS and ring-SIS. In particular, this understanding provides a mechanical translation of many cryptographic constructions from one problem to the other. This often also yields a mechanical translation of the security proof, although sometimes more significant changes are needed.

In short, in ring-SIS each random element $a_i \in R_q$ corresponds to n *related* (non-independent) vectors $\mathbf{a}_i \in \mathbb{Z}_q^n$ in SIS, where n is the degree of the ring R over $\mathbb{Z}$. Similarly, each ring element $z_i \in R$ of a ring-SIS solution stands in for a corresponding block of n integers in an SIS solution. This rule of thumb can be formalized by treating ring-SIS as a special case of SIS with "structured" instances:

- By fixing an appropriate $\mathbb{Z}$-basis of R, we obtain an additive group isomorphism between the input domains R and $\mathbb{Z}^n$, and the output domains R_q and $\mathbb{Z}_q^n$, which additionally preserves "shortness" (at least approximately). For example, for $R = \mathbb{Z}[X]/(X^n - 1)$ the monomial X^i corresponds to the $(i+1)$st standard basis vector $\mathbf{e}_{i+1} \in \mathbb{Z}^n$, for $i = 0, \ldots, n-1$. This yields the correspondence between ring-SIS inputs $\vec{z} \in R^m$ and SIS inputs $\mathbf{z} \in \mathbb{Z}^{nm}$, and similarly for the outputs.
- Left-multiplication by any fixed $a \in R_q$ is a $\mathbb{Z}$-linear function from R to R_q, so it can be represented by a (structured) square matrix $\mathbf{A}_a \in \mathbb{Z}_q^{n \times n}$, which maps $\mathbb{Z}^n$ to $\mathbb{Z}_q^n$. For example, for $R = \mathbb{Z}[X]/(X^n - 1)$ any $a \in R_q$ corresponds to the circulant matrix whose first column is the coefficient vector of a. This yields the correspondence between a ring-SIS instance $\vec{a} = (a_1, \ldots, a_m) \in R_q^m$ and the (structured) SIS instance $\mathbf{A} = [\mathbf{A}_{a_1} \mid \cdots \mid \mathbf{A}_{a_m}] \in \mathbb{Z}_q^{n \times nm}$.

In abstract algebra terms, in SIS the random elements $\mathbf{a}_i$ are drawn from the domain $\mathbb{Z}_q^n$, which is treated solely as an additive group, i.e., a $\mathbb{Z}$-*module*. An SIS solution is a short $\mathbb{Z}$-combination of the $\mathbf{a}_i$ that sums to zero. Whereas in ring-SIS, the random elements are drawn from R_q, which is treated as an R-*module*, and a solution is a short R-combination that sums to zero. The richer R-module structure is the source of the increased efficiency, but also of the more specialized underlying worst-case hardness assumptions (described in Section 4.3.4 below).

4.3.3 Geometry of Rings

A solution $\vec{z} \in R^m$ to a ring-SIS instance must be sufficiently "short" according to an appropriate choice of norm on R. A naïve choice of norm, used in several early works, is given by the *coefficient embedding* that associates each $z \in R = \mathbb{Z}[X]/(f(X))$ with the n-dimensional integer vector of coefficients of its canonical representative in $\mathbb{Z}[X]$. This choice can be useful for developing intuition, but it is non-canonical—it depends on the choice of representatives of R, which need not be polynomials in X of degree less than n—and it leads to unwieldy constraints on the form of $f(X)$, as well as rather crude analysis. For example, the norm of $a \cdot b \in R$ may be only loosely related to the norms of a and b, due to the reduction modulo $f(X)$.

A much better notion of norm is given by the classical notion of the *canonical embedding* $\sigma\colon R \to \mathbb{C}^n$ from algebraic number theory. This embedding maps each ring element $z \in R$ to the vector $(z(\alpha_i))_i \in \mathbb{C}^n$, where the $\alpha_i \in \mathbb{C}$ are the n complex roots of $f(X)$. Note that any representative of z yields the same vector in $\mathbb{C}^n$, which makes the embedding canonical. It also has the nice feature that the sum $a + b \in R$ and product $a \cdot b \in R$ respectively embed as the coordinate-wise sum and product of $\sigma(a), \sigma(b) \in \mathbb{C}^n$, which yields simple and rather sharp bounds on the norms of ring elements under addition and multiplication. We stress that the canonical embedding and complex numbers are used mainly for *analysis*, e.g., in security proofs; they do not ever need to be *computed* explicitly. For further discussion of these points and the advantages of the canonical embedding, see, e.g., [LPR10].

4.3.4 Ideal Lattices and Hardness of Ring-SIS

In short, R-SIS and its associated cryptographic functions can be proved at least as hard as certain lattice problems in the worst case, similarly to SIS. However, the underlying lattice problems are specialized to *algebraically structured* lattices, called *ideal lattices*, arising from the ring R. In addition, the algebraic and geometric properties of R play a major role in what kinds of security properties R-SIS can be expected to have, and in the quantitative strength of the underlying worst-case guarantee.

Ideal lattices. An ideal lattice is simply a lattice corresponding to an *ideal* in R under some fixed choice of geometric embedding, e.g., the coefficient or canonical embedding described in Section 4.3.3 above. Recall that an ideal of a commutative ring R is an additive subgroup $\mathcal{I} \subseteq R$ that is also closed under multiplication by R, i.e., $v \cdot r \in \mathcal{I}$ for every $v \in \mathcal{I}, r \in R$. This multiplicative closure means that ideal lattices have geometric symmetries that lattices do not have in general. For example, under the coefficient embedding of $R = \mathbb{Z}[X]/(X^n - 1)$, an ideal corresponds to a *cyclic* lattice in $\mathbb{Z}^n$, i.e., one which is closed under cyclic rotations of the coordinates of $\mathbb{Z}^n$. This is because ideals of R are closed under multiplication by X, which corresponds to rotation by one coordinate in the coefficient embedding.

As described below, the known hardness proofs for R-SIS relate to lattice problems that are restricted to ideal lattices in R. The complexity of such problems is a bit different than for arbitrary lattices. For example, for typical choices of rings the *decision* problem GapSVP_γ for small $\gamma = \mathrm{poly}(n)$ factors is actually *easy* on ideal lattices, because the algebraic symmetries force the minimum distance of an ideal to lie within a narrow, easily computable range. In addition, the approximate SVP and SIVP problems are *equivalent* (sometimes up to a small loss in the approximation factor), because the symmetries allow one short nonzero vector to be converted into n linearly independent ones of the same length (or nearly so).

For typical choices of rings, and for cryptographically relevant approximation factors γ, the SVP_γ and SIVP_γ problems on ideal lattices appear to be very hard in the worst case, even for quantum algorithms. Indeed, despite the additional algebraic structure of ideal lattices, no significant speedup for these problems is known, relative to general lattices of the same dimension. In particular, the best known (quantum) algorithms for $\mathsf{SVP}_{\mathrm{poly}(n)}$ on ideal lattices in typical choices of rings take exponential $2^{\Omega(n)}$ time. However, ideal lattices have not been investigated nearly as deeply from a computational point of view, so hardness conjectures concerning them may not yet deserve as much confidence.

One-wayness. For the ring $R = \mathbb{Z}[X]/(X^n - 1)$ and other appropriate parameters, Micciancio proved that the function $f_{\vec{a}}$ defined in Equation (4.3.1) is *one-way*, assuming the worst-case hardness of certain problems on n-dimensional *cyclic* lattices, i.e., ideal lattices in R under the coefficient embedding. Equivalently, he showed the hardness of the *inhomogeneous* version of ring-SIS, in which one seeks a short solution to

Equation (4.3.1) for a uniformly random right-hand side (instead of zero). However, he left it as an open problem to determine whether the function is *collision-resistant*, as Ajtai's SIS-based function is. Recall that collision resistance is essentially equivalent to the hardness of homogeneous ring-SIS.

Collision resistance. The concurrent and independent works of Peikert and Rosen [PR06] and Lyubashevsky and Micciancio [LM06], published in 2006, showed that the function $f_{\vec{a}}$ over $R = \mathbb{Z}[X]/(X^n - 1)$ turns out *not* to be collision resistant, i.e., homogeneous R-SIS is easy. The reason turns out to be that the ring is not an *integral domain*, and zero divisors can be used to construct collisions.

On the positive side, however, the same works [PR06, LM06] showed that over appropriate integral domains R, the function $f_{\vec{a}}$ is indeed collision resistant (equivalently, R-SIS is hard), assuming that SVP_γ for $\gamma = \beta \cdot \mathrm{poly}(n)$ is hard in the worst case for ideal lattices in R. Prominent examples of suitable rings include *cyclotomic* rings like $\mathbb{Z}[X]/(X^n + 1)$ for power-of-two n, and $\mathbb{Z}[X]/(X^{p-1} + \cdots + X^1 + 1)$ for prime p. In general, cyclotomic rings admit quite fast and elegant ring operations, using FFT-like techniques described in detail in [LMPR08, LPR13].

Tighter approximation factors from number fields. Subsequent work by Peikert and Rosen [PR07] generalized the above results, demonstrating that R-SIS is at least as hard as worst-case SVP_γ on ideal lattices in R, where $R = \mathcal{O}_K$ is the ring of *algebraic integers* in any number field K. Notably, the approximation factor for the underlying SVP_γ problem can be as small as $\gamma = O(\sqrt{\log n})$ in certain families of number fields. This work revealed how the *discriminant* of the number field—essentially, the determinant of R under the canonical embedding—controls the worst-case approximation factors.

4.4 Ring-LWE

In work published in 2010, Lyubashevsky, Peikert, and Regev [LPR10] introduced *ring-LWE*, the ring-based analogue of learning with errors, and proved the hardness theorems described below in Section 4.4.2. (A concurrent and independent work by Stehlé *et al.* [SSTX09] also considered a special case of ring-LWE for rings of the form $R = \mathbb{Z}[X]/(X^n + 1)$ for power-of-two n, and proved weaker results, e.g., it lacked a hardness proof for the decision form).

We recommend a thorough understanding of the definitions and issues relating to ring-SIS (Section 4.3) before tackling ring-LWE, which involves some additional subtleties.

4.4.1 Definitions

Ring-LWE is parameterized by a ring R of degree n over $\mathbb{Z}$, a positive integer modulus q defining the quotient ring $R_q = R/qR$, and an error distribution χ over R. Typically, one takes R to be a *cyclotomic* ring, and χ to be some kind of discretized Gaussian in the canonical embedding of R, which we can roughly think of as having an "error rate" $\alpha < 1$ relative to q. (However, the precise form of the error distribution needed for proving hardness is somewhat subtle; see Section 4.4.2 for discussion.)

Definition 4.4.1 (Ring-LWE distribution). For an $s \in R_q$ called the *secret*, the ring-LWE distribution $A_{s,\chi}$ over $R_q \times R_q$ is sampled by choosing $a \in R_q$ uniformly at random, choosing $e \leftarrow \chi$, and outputting $(a, b = s \cdot a + e \bmod q)$.

We note that in the original definition of the ring-LWE distribution from [LPR10], the secret $s^\vee$ and noisy product $b^\vee$ actually belong to $R_q^\vee := R^\vee/qR^\vee$, where $R^\vee$ is a certain fractional ideal that is dual to R. It

turns out that this is the right definition for the hardness proof and cryptographic applications when using *(near-)spherical errors* $e^\vee$ in the canonical embedding of R, which are easiest to analyze and derive sharp bounds for (see [LPR10, Section 3.3] and [LPR13] for details). The form of the problem defined above (where $s, b \in R_q$) can be obtained from the version just described simply by multiplying $s^\vee$ and $b^\vee$ by a certain "tweak" factor t, where $tR^\vee = R$:

$$\underbrace{t \cdot b^\vee}_{b} = \underbrace{(t \cdot s^\vee)}_{s} \cdot a + \underbrace{(t \cdot e^\vee)}_{e} \in R/qR.$$

Notice that this yields a "tweaked," possibly non-spherical distribution χ for the error e. Yet these two forms of the problem are entirely equivalent in terms of computation, applications, analysis, etc., because the tweak is reversible; see, e.g., [LPR13, AP13] for details. (We mention that an alternative way of replacing $R^\vee$ with R was described in [DD12], but it incurs some computational and analytical losses, because it is not reversible.)

The decision version of the R-LWE problem is to distinguish between ring-LWE samples and uniformly random ones. As usual, we also parameterize the problem by the number m of available samples, which is sometimes left unspecified.

Definition 4.4.2 (Decision-R-LWE$_{q,\chi,m}$). Given m independent samples $(a_i, b_i) \in R_q \times R_q$ where every sample is distributed according to either: (1) $A_{s,\chi}$ for a uniformly random $s \in R_q$ (fixed for all samples), or (2) the uniform distribution, distinguish which is the case (with non-negligible advantage).

Just as in LWE, without errors the ring-LWE problem is easy, because in case (1) we can efficiently find s: given a sample (a_i, b_i) where $a_i \in R_q$ is invertible (most elements of R_q are), we have $s = b_i \cdot a_i^{-1}$, whereas in case (2) there will almost never be a single s that is consistent with all samples. Similarly, ring-LWE has a *normal form*, in which the secret s is chosen from the error distribution (modulo q), rather than uniformly. It is easy to show that this form of the problem is at least as hard as the one defined above, by adapting the proof described in Section 4.2.1.

The primary advantage of ring-LWE is its compactness and efficiency: each sample (a_i, b_i) yields an n-dimensional pseudorandom ring element $b_i \in R_q$, rather than just a single pseudorandom scalar $b_i \in \mathbb{Z}_q$ as in LWE. In addition, ring multiplication can be performed in only quasi-linear $\tilde{O}(n)$ time using FFT-like techniques, so we can generate these n pseudorandom scalars in just $\tilde{O}(1)$ amortized time each. For example, this all yields a public-key encryption scheme with only $\tilde{O}(1)$-factor overheads in encryption/decryption time and ciphertext space, versus sending the plaintext in the clear. See Section 5.2 for further details.

Relation to LWE. It is helpful to understand the algebraic similarities and differences between LWE and ring-LWE. Just as with (ring-)SIS, a single ring-LWE sample with a random $a_i \in R_q$ takes the place of n LWE samples with random $\mathbf{a}_i \in \mathbb{Z}_q^n$. So ring-LWE can be seen as a special case of LWE with "structured" (correlated) samples.

In abstract algebraic terms, in LWE the secret $\mathbf{s}$ and random $\mathbf{a}_i$ are elements $\mathbb{Z}_q^n$, which is treated as a $\mathbb{Z}$*-module*, and they are multiplied using the $\mathbb{Z}$*-bilinear* inner product $\langle \cdot, \cdot \rangle \colon \mathbb{Z}_q^n \times \mathbb{Z}_q^n \to \mathbb{Z}_q$. By contrast, in ring-LWE the secret s and random a_i are elements of R_q, which is treated as an R*-module*, and they are multiplied using standard R*-bilinear* multiplication in R_q.

4.4.2 Hardness

Like LWE, ring-LWE enjoys a worst-case hardness guarantee, informally stated here:

Theorem 4.4.3 ([LPR10]). *For any $m = \mathrm{poly}(n)$, cyclotomic ring R of degree n (over $\mathbb{Z}$), and appropriate choices of modulus q and error distribution χ of error rate $\alpha < 1$, solving the R-LWE$_{q,\chi,m}$ problem is at least as hard as* quantumly *solving the SVP_γ problem on* arbitrary *ideal lattices in R, for some $\gamma = \mathrm{poly}(n)/\alpha$.*

(See Section 4.3.4 for a discussion of ideal lattices and SVP_γ.) Notice that as with LWE, the approximation factor γ varies inversely with the error rate α of χ. Unlike with LWE, however, the factor γ also degrades slightly with the number of samples m. This degradation may be an artifact of the proof technique, and in any case it can be avoided by choosing the error distribution *itself* at random from a certain family. See the main theorem from [LPR10] for more details.

As mentioned above, the precise form of the error distribution χ in the above theorem is somewhat delicate, as it relies on the canonical embedding and the "tweak" factor that transforms $R^\vee$ to R (when using the form of the problem from Definition 4.4.1). In particular, the $\mathbb{Z}$-coefficients of the error terms $e \leftarrow \chi$ are *not* necessarily independent in any $\mathbb{Z}$-basis of R, but they can still be sampled very efficiently in an appropriate choice of basis. See [LPR10, LPR13] for full details.

The above theorem is proved in two parts: first, the *search* version of ring-LWE, which is to recover the secret s given many samples from $A_{s,\chi}$, is proved to be at least as hard as SVP_γ, using a *quantum* reduction. This part of the proof actually holds for *any* ring of integers R of a number field (not just cyclotomics) and any sufficiently large modulus q. Then, a *classical* search-to-decision reduction is used to prove that the decision version is at least as hard as the search version. This part of the proof relies on additional algebraic properties of cyclotomics and the form of the modulus q, namely, that cyclotomics are *Galois* over the rationals, and that q splits into the product of distinct small-norm prime ideals in R.

4.4.3 Generalizations

Brakerski, Gentry, and Vaikuntanathan [BGV12] introduced a generalized ring-LWE problem (R-GLWE), which essentially interpolates between LWE and ring-LWE: the secret is a vector $\vec{s} \in R_q^k$ of ring elements, and GLWE samples are of the form $(\vec{a}, b) \in R_q^k \times R_q$, where either $b = \langle \vec{s}, \vec{a} \rangle + e \bmod q$ for $e \leftarrow \chi$, or $b \in R_q$ is uniformly random. For $R = \mathbb{Z}$ this specializes to the k-dimensional LWE$_{k,q,\chi}$ problem, and for $k = 1$ it specializes to R-LWE$_{q,\chi}$. Generalizing Theorem 4.4.3, Langlois and Stehlé [LS15] proved that R-GLWE is at least as hard as quantumly approximating worst-case lattice problems on so-called *module lattices*, which are lattices corresponding to R-modules $M \subseteq R^k$.

4.4.4 Relation to NTRU

Recall from Section 3.2 that the NTRU cryptosystem of Hoffstein, Pipher, and Silverman [HPS98] was an early lattice-based cryptographic proposal. Several computational problems naturally relate to the NTRU system. One such problem is the following:

Definition 4.4.4 (NTRU learning problem). For an invertible $s \in R_q^*$ and a distribution χ on R, define $N_{s,\chi}$ to be the distribution that outputs $e/s \in R_q$ where $e \leftarrow \chi$. The *NTRU learning problem* is: given independent samples $a_i \in R_q$ where every sample is distributed according to either: (1) $N_{s,\chi}$ for some randomly chosen $s \in R_q^*$ (fixed for all samples), or (2) the uniform distribution, distinguish which is the case (with non-negligible advantage).

In the NTRU cryptosystem, the public key is one sample e/s for a short s, and ciphertexts essentially correspond to additional samples with the same denominator s, though with somewhat less-short numerators.

Just as with the normal form of ring-LWE, we can assume without loss of generality that the secret denominator s is chosen from the distribution χ, restricted to units in R_q. This is because given samples $a_i = e_i/s \in R_q$ from N_s for some arbitrary $s \in R_q^*$, we can take one that is a unit—call it a_0—and divide the remaining samples by it, yielding samples $a'_i = a_i/a_0 = e_i/e_0$ whose common denominators are the short unit e_0. Because the same transformation on uniformly random samples preserves the uniform distribution, being able to distinguish normal-form NTRU samples from uniform implies being able to do so for NTRU samples with an arbitrary denominator.

The NTRU and ring-LWE problems are syntactically very similar, and can even be viewed as homogeneous and inhomogeneous versions of the same problem. Specifically, for NTRU samples $a_i \in R_q$ there is a secret s such that every $a_i \cdot s = e_i \pmod q$ for some short $e_i \in R$, while for ring-LWE samples $(a_i, b_i) \in R_q \times R_q$, there is a secret s such that every $a_i \cdot s + b_i = e_i \pmod q$ for some short $e_i \in R$. This interpretation often makes it possible to adapt cryptographic constructions from one problem to the other (e.g., [BV11b] and [LTV12]).

Ring-LWE is at least as hard as NTRU. Here we sketch a proof that ring-LWE is at least as hard as the NTRU learning problem, for appropriate parameters. (Although the proof strategy is relatively standard by now, we have not seen this particular result documented before.) More specifically, we describe a reduction from the *decision* version of NTRU to the *search* version of ring-LWE. Using the search-decision equivalence for the latter problem, we can also extend the reduction to the decision version, with one caveat that we explain below. The reduction works by a "lossiness" argument, *à la* [PW08, Pei09, DM13, MP13]. Suppose we have an oracle $\mathcal{O}$ that solves search-R-LWE with high probability, given ℓ samples. Then an algorithm that solves the NTRU learning problem works as follows: given ℓ samples $a_i \in R_q$ as input, it chooses a secret s and errors e_i from the LWE error distribution χ and gives the pairs $(a_i, b_i = a_i \cdot s + e_i)$ to $\mathcal{O}$, which returns some $\hat{s}$. If $\hat{s} = s$, our algorithm accepts, otherwise it rejects.

To analyze the reduction, first consider the case where the a_i are uniformly random. Then the input we give to $\mathcal{O}$ consists of properly distributed ring-LWE samples with secret s, so $\mathcal{O}$ must return $\hat{s} = s$ with high probability, and our algorithm accepts. In the other case, we have $a_i = e'_i/s' \pmod q$ for some random short s', e'_i drawn from the distribution χ' used in the NTRU problem. Then as long as the LWE error distribution χ is sufficiently "wider" than χ', the pairs $(a_i, b_i = s \cdot e'_i/s' + e_i)$ *information-theoretically hide* the value of s. That is, there are multiple possibilities for the secret and errors that are consistent with the pairs; for example, they could instead respectively be the short elements $s + s'$ and $e_i - 1$. Therefore, no matter how the oracle $\mathcal{O}$ works internally, and even though its input pairs are not properly distributed ring-LWE samples, it cannot reliably guess the particular s that our algorithm chose, and so our algorithm rejects with noticeable probability. This implies that our algorithm has noticeable distinguishing advantage between NTRU samples and uniformly random ones, as desired.

The above outline of course omits the precise calculations needed for the lossiness argument, but these are by now rather routine (see, e.g., [PW08, GKPV10, BKPW12, DM13, MP13]). We note that in order for the lossiness property to hold, the ratio of the widths of χ and χ' must grow with ℓ, the number of ring-LWE samples used by the oracle $\mathcal{O}$ for *search*-ring-LWE. If we wish to connect the *decision* versions of NTRU and ring-LWE via the known search-decision equivalence for the latter problem, then ℓ can be an unbounded polynomial (it depends inversely on the advantage of the ring-LWE distinguisher). Therefore, the ratio of the widths of χ and χ' would need to be super-polynomial, which weakens the result. A possible way of circumventing this problem is to give a *sample-preserving* search-decision equivalence for ring-LWE, which would keep ℓ small; see Chapter 7 for details.

Chapter 5

Essential Cryptographic Constructions

In this chapter we survey a core collection of lattice-based cryptographic construction, which are all built upon the (ring-)SIS/LWE problems. The presentation is organized by type of cryptographic object:

- one-way and collision-resistant hash functions (Section 5.1);
- passively secure encryption (Section 5.2);
- actively secure encryption (Section 5.3);
- trapdoor functions (Section 5.4) and their applications, like digital signatures and identity-based encryption (Section 5.5);
- digital signatures without trapdoors (Section 5.6); and
- pseudorandom functions (Section 5.7).

Because the presentation necessarily departs from a linear chronology, we point out several instances in which advances in one subarea influenced works in another.

5.1 Collision-Resistant Hash Functions

Recall from Sections 4.1 and 4.3 that the functions $f_{\mathbf{A}}$ and $f_{\vec{a}}$ (respectively defined in Equations (4.1.1) and (4.3.1)) are collision resistant assuming the hardness of the corresponding (ring-)SIS problem.

Lyubashevsky *et al.* [LMPR08] defined SWIFFT, which is a concrete instantiation of the ring-SIS-based hash function $f_{\vec{a}}$. The instantiation was chosen to admit fast computation using various FFT and precomputation techniques, and to have an estimated 2^{100} security level for collision resistance against known cryptanalytic attacks. It should be noted that the parameterization of SWIFFT corresponds to a vacuous worst-case security guarantee, because the degree $n = 64$ of the ring is small enough that one can find relatively short vectors in n-dimensional lattices in a moderate amount of time. This does not mean that SWIFFT is insecure, however, because the worst-case guarantee simply provides a *lower bound* on the hardness of breaking the function. In practice, it appears that SWIFFT and other instantiations of Ajtai's function are substantially harder to break than the worst-case problems used in the hardness proofs.

5.2 Passively Secure Encryption

Since the introduction of the (ring-)LWE problems, a large number of encryption schemes and other applications have been based upon them. In this subsection we give a semi-chronology of LWE-based public-key

encryption schemes having *passive* (IND-CPA) security. (Recall that this means, informally, that a passive eavesdropper who sees the public key and encrypted messages learns nothing about the contents of those messages.) Many of the schemes presented in this section have additional useful properties (e.g., homomorphisms, security for key-dependent messages), which we largely omit from the discussion.

5.2.1 Regev's LWE Cryptosystem

Recall that Regev [Reg05] gave the first LWE-based public-key encryption scheme, in which public keys are $\tilde{O}(n^2)$ bits, secret keys and ciphertexts are $\tilde{O}(n)$ bits, and each ciphertext encrypts a single bit. (Here n is the dimension of the underlying LWE problem.) In the multi-user setting, if there is a trusted source of randomness that can be shared among all users, then the per-user public key size can be reduced to only $\tilde{O}(n)$ bits.

Description of the system. The cryptosystem is parameterized by an LWE dimension n, modulus q, error distribution χ over $\mathbb{Z}$, and number of samples m that all should satisfy various conditions needed for security and correct decryption, as described below.

- The secret key is a uniformly random LWE secret $\mathbf{s} \in \mathbb{Z}_q^n$, and the public key is some $m \approx (n+1)\log q$ samples $(\bar{\mathbf{a}}_i, b_i = \langle \mathbf{s}, \bar{\mathbf{a}}_i \rangle + e_i) \in \mathbb{Z}_q^{n+1}$ drawn from the LWE distribution $A_{\mathbf{s},\chi}$, collected as the columns of a matrix

$$\mathbf{A} = \left[\begin{array}{c} \bar{\mathbf{A}} \\ \mathbf{b}^t \end{array} \right] \in \mathbb{Z}_q^{(n+1)\times m}, \tag{5.2.1}$$

  where $\mathbf{b}^t = \mathbf{s}^t\bar{\mathbf{A}} + \mathbf{e}^t \bmod q$. (In the multi-user setting, $\bar{\mathbf{A}}$ can be shared among all users, and the user's public key is just $\mathbf{b}$.) Note that by definition, the secret and public keys satisfy the relation

$$(-\mathbf{s}, 1)^t \cdot \mathbf{A} = \mathbf{e}^t \approx \mathbf{0} \pmod q. \tag{5.2.2}$$

- To encrypt a bit $\mu \in \mathbb{Z}_2 = \{0,1\}$ using the public key $\mathbf{A}$, one just takes a random subset-sum of the LWE samples and appropriately encodes the message bit in the last coordinate.[1] More specifically, one chooses a uniformly random $\mathbf{x} \in \{0,1\}^m$ and outputs the ciphertext

$$\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil) \in \mathbb{Z}_q^{n+1}. \tag{5.2.3}$$

  Notice that, ignoring the $\mu \cdot \lfloor \frac{q}{2} \rceil$ term, encryption is merely evaluation of the function $f_{\mathbf{A}}$ from Equation (4.1.1) on a random binary input $\mathbf{x}$, although here the matrix $\mathbf{A}$ is not uniformly random, but is instead pseudorandom.

- To decrypt using the secret key $\mathbf{s}$, one computes

$$\begin{aligned} (-\mathbf{s}, 1)^t \cdot \mathbf{c} &= (-\mathbf{s}, 1)^t \cdot \mathbf{A} \cdot \mathbf{x} + \mu \cdot \lfloor \tfrac{q}{2} \rceil \\ &= \mathbf{e}^t \cdot \mathbf{x} + \mu \cdot \lfloor \tfrac{q}{2} \rceil && \text{(Equation (5.2.2))} \\ &\approx \mu \cdot \lfloor \tfrac{q}{2} \rceil \pmod q && (\mathbf{e}, \mathbf{x} \in \mathbb{Z}^m \text{ are short}) \end{aligned}$$

  and tests whether it is closer to 0 or to $\lfloor \frac{q}{2} \rceil$ modulo q.

[1] More generally, it was observed by [KTX08, PW08] that one can encrypt messages from $\mathbb{Z}_p$ using q/p in place of $q/2$, as long as q/p is sufficiently large.

Notice that decryption is correct as long as the accumulated error $\langle \mathbf{e}, \mathbf{x} \rangle \in \mathbb{Z}$ has magnitude less than $q/4$. This can be made to hold simply by choosing q to be large enough relative to the error distribution χ and the value of m. For example, if $\chi = D_{\mathbb{Z},r}$ is a discrete Gaussian, which is subgaussian with parameter r, then $\langle \mathbf{e}, \mathbf{x} \rangle$ is subgaussian with parameter at most $r\sqrt{m}$, and hence has magnitude less than $r\sqrt{m \ln(1/\varepsilon)/\pi}$ with probability at least $1 - 2\varepsilon$.[2] So to ensure correct decryption with overwhelming probability, along with security under a worst-case assumption (as discussed next), one can use parameters as small as $r = \Theta(\sqrt{n})$ and $q = \tilde{O}(n)$, which correspond to an LWE error rate of $\alpha = r/q = 1/\tilde{O}(\sqrt{n})$ and worst-case approximation factors of $\gamma = \tilde{O}(n^{3/2})$.

Security. Regev's system is semantically secure against passive eavesdroppers, assuming that decision-LWE$_{n,q,\chi,m}$ is hard, which for appropriate parameters is implied by the conjectured worst-case (quantum) hardness of lattice problems (see Section 4.2.2).

Here we give a reasonably detailed outline of the security proof, which follows a strategy that has come to be known as a "lossiness" argument. The two main ideas are: 1. a properly formed public key $\mathbf{A}$ is indistinguishable from a "malformed" uniformly random one, and 2. encrypting under such a malformed key is *information-theoretically* secure. More formally, recall that we wish to show that a public key $\mathbf{A}$ together with an encryption $\mathbf{c}$ of a fixed bit μ are indistinguishable for $\mu = 0, 1$ (see Section 2.4.2). We proceed by considering a sequence of alternative, or "hybrid," experiments that produce $\mathbf{A}, \mathbf{c}$ in different ways:

- In the first hybrid experiment, the public key $\mathbf{A}$ is "malformed" in the sense that it is chosen uniformly at random from $\mathbb{Z}_q^{(n+1)\times m}$, instead of being generated from LWE samples. (Note that there is no corresponding secret key.) The ciphertext $\mathbf{c}$ is generated by encrypting μ using $\mathbf{A}$ in the usual way, as $\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + (\mathbf{0}, \mu \cdot \lfloor \frac{q}{2} \rceil) \in \mathbb{Z}_q^{n+1}$.

  We claim that this experiment is indistinguishable from the real one, under the LWE assumption. This is shown by a reduction: any hypothetical attacker $\mathcal{A}$ that aims to distinguish the two experiments can be transformed into an algorithm $\mathcal{D}$ that aims to distinguish LWE samples from uniformly random ones, i.e., it attacks decision-LWE$_{n,q,\chi,m}$: $\mathcal{D}$ simply collects its input samples into a matrix $\mathbf{A}$, encrypts μ using $\mathbf{A}$ to get a ciphertext $\mathbf{c}$, and invokes $\mathcal{A}$ on $(\mathbf{A}, \mathbf{c})$, outputting the same accept/reject decision. It is clear that $\mathcal{D}$ perfectly simulates the real or hybrid experiment, depending on whether its input samples are LWE or uniform (respectively); therefore, $\mathcal{D}$ and $\mathcal{A}$ have equal distinguishing advantages. Because $\mathcal{D}$'s advantage must be negligible by hypothesis, so is $\mathcal{A}$'s.

- In the second hybrid experiment, the public key $\mathbf{A}$ is still uniformly random, but now the ciphertext $\mathbf{c} \in \mathbb{Z}_q^{n+1}$ is also chosen uniformly and independently of $\mathbf{A}$.

  We claim that this experiment is *statistically* indistinguishable from the previous one, i.e., even a computationally unbounded attacker has only negligible advantage in distinguishing them. In other words, encrypting under a uniformly random public key is "lossy," in that it hides the message information-theoretically. The claim follows immediately from the fact that $m \approx (n+1)\log q$ is sufficiently large, and by a *regularity lemma* (also known as the *leftover hash lemma*) [HILL99], which says that $(\mathbf{A}, \mathbf{u} = \mathbf{A} \cdot \mathbf{x})$ for uniform and independent $\mathbf{A} \leftarrow \mathbb{Z}_q^{(n+1)\times m}$ and $\mathbf{x} \leftarrow \{0,1\}^m$ is statistically indistinguishable from uniformly random. (Clearly, adding any fixed vector $(\mathbf{0}, \mu \cdot \lfloor \frac{q}{2} \rceil)$ to $\mathbf{u}$ preserves its uniform distribution.)

[2] Using a slightly larger modulus q, one can even ensure correct decryption *with certainty* by rejecting any (negligibly rare) error vector $\mathbf{e}$ that is too long in Euclidean norm during key generation.

In conclusion, because the above experiments are indistinguishable for any fixed bit μ, and the last one does not depend on μ at all, the two real experiments for $\mu = 0, 1$ are also indistinguishable.

As a final remark, we note that the system is trivially breakable under an *active*, or chosen-ciphertext, attack. We discuss actively secure LWE-based encryption in Section 5.3 below.

Normal form optimization. As documented in [MR09], the above cryptosystem, along with essentially all other LWE-based systems, is amenable to a mild optimization using the "normal forms" of SIS/LWE defined in Sections 4.1.1 and 4.2.1. (Indeed, some systems described below incorporate this optimization explicitly.) For the same parameters n, m as above, we let the matrix $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times (m-n)}$ have only $m - n$ columns, and define $\mathbf{A} \in \mathbb{Z}_q^{(n+1)\times(m-n)}$ as in Equation (5.2.1) above, where the coordinates of $\mathbf{s} \in \mathbb{Z}^n$ are chosen from the *error distribution* χ. To encrypt a bit $\mu \in \{0, 1\}$, one chooses a uniformly random $\mathbf{x} \in \{0, 1\}^{m+1}$ and outputs the ciphertext

$$\mathbf{c} = [\mathbf{I}_{n+1} \mid \mathbf{A}] \cdot \mathbf{x} + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil) \in \mathbb{Z}_q^{n+1}.$$

To decrypt given the secret key $\mathbf{s}$, one computes

$$\begin{aligned}
(-\mathbf{s}, 1)^t \cdot \mathbf{c} &= (-\mathbf{s}, 1)^t \cdot [\mathbf{I}_{n+1} \mid \mathbf{A}] \cdot \mathbf{x} + \mu \cdot \lfloor \tfrac{q}{2} \rceil \\
&= (-\mathbf{s}, 1, \mathbf{e})^t \cdot \mathbf{x} + \mu \cdot \lfloor \tfrac{q}{2} \rceil && \text{(Equation (5.2.2))} \\
&\approx \mu \cdot \lfloor \tfrac{q}{2} \rceil \pmod q && (\mathbf{s}, \mathbf{e}, \mathbf{x} \text{ are short})
\end{aligned}$$

and tests whether it is closer to 0 or to $\lfloor \frac{q}{2} \rceil$ modulo q.

The security proof for this variant is essentially the same as the one outlined above, but it now relies on the hardness of the normal form of decision-LWE, as well as a regularity lemma for matrices of the form $[\mathbf{I}_{n+1} \mid \mathbf{A}]$ for uniformly random $\mathbf{A}$.

Longer messages. Typically, one wishes to encrypt several bits at a time, e.g., to transmit a key for a symmetric encryption scheme. In this context, Peikert, Vaikuntanathan, and Waters [PVW08] described a significant efficiency improvement using an *amortization* technique. In their variant, one can encrypt $\ell = O(n)$ bits per ciphertext, with no asymptotic increase in the sizes of the public key or ciphertexts, nor in the runtime of encryption. However, the secret key size and decryption runtimes are increased to $\tilde{O}(\ell \cdot n)$, versus $\tilde{O}(n)$ in the original system.

The main idea is, instead of using an $(n+1)$-row public key of the form $\mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{b}^t \approx \mathbf{s}^t \bar{\mathbf{A}} \end{bmatrix}$, to generate an $(n + \ell)$-row key of the form

$$\mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{B} \approx \mathbf{S}^t \cdot \bar{\mathbf{A}} \end{bmatrix} \in \mathbb{Z}_q^{(n+\ell)\times m},$$

where the ℓ rows of $\mathbf{S}^t \in \mathbb{Z}_q^{\ell \times n}$ are independent LWE secrets, and each entry of $\mathbf{S}^t \cdot \mathbf{A}$ is perturbed by independent error drawn from χ. Encrypting a message $\mathbf{m} \in \{0, 1\}^\ell$ works essentially as before, by choosing uniformly random $\mathbf{x} \in \{0, 1\}^m$ and outputting the ciphertext

$$\mathbf{c} = \mathbf{A} \cdot \mathbf{x} + (\mathbf{0}, \mathbf{m} \cdot \lfloor \tfrac{q}{2} \rceil) \in \mathbb{Z}_q^{n+\ell}.$$

For security, by a routine hybrid argument, it can be shown that a public key $\mathbf{A}$ is indistinguishable from uniform assuming the hardness of decision-LWE. Moreover, for $m \approx (n + \ell) \log q = \tilde{O}(n)$, the regularity lemma and lossiness argument described above still apply, thus establishing semantic security.

A separate mild optimization relates to the way that the message bits are encoded to be recoverable under noise. Throughout this survey, for simplicity we encode $\mathbf{m} \in \{0,1\}^\ell$ as $\mathbf{m} \cdot \lfloor \frac{q}{2} \rceil \in \mathbb{Z}_q^\ell$. This incurs a multiplicative overhead of $\log q$, along with the additive overhead of the ciphertext "preamble" $\bar{\mathbf{A}}\mathbf{x}$. Recently, Peikert [Pei14] described a more sophisticated "reconciliation" mechanism that encodes the message bit-for-bit, making the ciphertext overhead merely additive. This mechanism works for any value of ℓ and essentially any (ring-)LWE cryptosystem.

Generating fresh LWE samples. We conclude this coverage of Regev's cryptosystem by noting that its encryption algorithm (Equation (5.2.3)) implicitly contains a method for generating unboundedly many "fresh" LWE samples for a fixed secret and a somewhat wider Gaussian error distribution, given a sufficiently large number of initial samples. More specifically, an encryption $(\mathbf{a}, b) = \mathbf{A} \cdot \mathbf{x} \in \mathbb{Z}_q^{n+1}$ of zero is essentially a new LWE sample with secret $\mathbf{s}$, in the sense that $\mathbf{a}$ is negligibly far from uniform and independent of $\bar{\mathbf{A}}$, and

$$b = \langle \mathbf{s}, \mathbf{a} \rangle + \langle \mathbf{e}, \mathbf{x} \rangle \approx \langle \mathbf{s}, \mathbf{a} \rangle \pmod{q}.$$

Therefore, $(\mathbf{a}, b)$ constitutes a noisy linear equation in $\mathbf{s}$. However, for the system as described above, the distribution of the error term $\langle \mathbf{e}, \mathbf{x} \rangle \in \mathbb{Z}$ (over the random choice of $\mathbf{x}$) may not be so "nice"—it is not easy to analyze, and it may even vary with the value of $\mathbf{a}$. So the samples generated in this way may not quite be fresh LWE samples in the sense we usually mean, i.e., from a distribution $A_{\mathbf{s},\chi}$.

Fortunately, it was shown in [GPV08, ACPS09], using a key lemma from [Reg05], that a slightly modified procedure does indeed generate LWE samples having a true Gaussian error distribution (up to negligible statistical error). To do this, one instead chooses $\mathbf{x}$ according to a *discrete Gaussian* $D_{\mathbb{Z}^m,r}$ for appropriate $r = \tilde{O}(1)$, and adds a little "smoothing" error to the final coordinate of $\mathbf{A} \cdot \mathbf{x}$. The error in the resulting sample is then statistically close to Gaussian with parameter $O(r \cdot \|\mathbf{e}\|)$, where $\mathbf{e}$ is the error vector in the original LWE samples. (Note that this original error $\mathbf{e}$ can come from any distribution, as long as it is relatively short.) Moreover, when the input matrix $\mathbf{A}$ is uniformly random (instead of from the LWE distribution), the same procedure produces samples that are nearly uniformly random and independent of $\mathbf{A}$. Therefore, the procedure is a form of *randomized self-reduction* for both the search and decision forms of LWE.

5.2.2 Dual LWE Cryptosystem

Gentry, Peikert, and Vaikuntanathan (hereafter GPV) [GPV08] defined an LWE-based public-key encryption scheme which can be viewed as "dual" to the above-described ones of Regev [Reg05] and Peikert *et al.* [PVW08]. The systems are duals in the following sense: in the above schemes, public keys have a non-uniform (LWE) distribution with a unique secret key, yet there are many choices of encryption randomness that produce the same ciphertext (for a given public key). In the GPV system, by contrast, public keys are uniformly random with many possible secret keys, whereas the encryption randomness that produces a particular ciphertext is unique. It turns out that having many possible secret keys is tremendously useful for constructing a variety of more advanced cryptosystems, as covered in later sections.

Description of the system.

- To generate a key pair, one first chooses a uniformly random $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times m}$ for a sufficiently large $m \approx n \log q$. (In the multi-user setting, $\bar{\mathbf{A}}$ can be chosen by a trusted party and shared among all users.) The secret key is a uniformly random $\mathbf{x} \in \{0,1\}^m$, and the public key is $\mathbf{A} = [\bar{\mathbf{A}} \mid \mathbf{u} = \bar{\mathbf{A}}\mathbf{x}] \in \mathbb{Z}_q^{n \times (m+1)}$. Notice that the public and secret keys satisfy the relation

$$\mathbf{A} \cdot (-\mathbf{x}, 1) = \mathbf{0} \pmod{q}. \tag{5.2.4}$$

Also notice that finding a valid secret key for a given public key $\bar{\mathbf{A}}, \mathbf{u}$ is essentially the (inhomogeneous) SIS problem.

- To encrypt a bit $\mu \in \{0,1\}$, one chooses an LWE secret $\mathbf{s} \in \mathbb{Z}_q^n$ and outputs the ciphertext

$$\mathbf{c}^t \approx \mathbf{s}^t \mathbf{A} + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil)^t, \tag{5.2.5}$$

  where the approximations hide independent errors drawn from the LWE error distribution χ.

- To decrypt using the secret key $\mathbf{x}$, one computes

$$\begin{aligned} \mathbf{c}^t \cdot (-\mathbf{x}, 1) &\approx \mathbf{s}^t \cdot \mathbf{A} \cdot (-\mathbf{x}, 1) + \mu \cdot \lfloor \tfrac{q}{2} \rceil && \text{(Equation (5.2.5); } \mathbf{x} \text{ is short)} \\ &= \mu \cdot \lfloor \tfrac{q}{2} \rceil && \text{(Equation (5.2.4))} \end{aligned} \tag{5.2.6}$$

  and tests whether the results is closer to 0 or to $q/2$ modulo q. Note that the total error in the above approximation is essentially the same as in Regev's system.

Security. The dual cryptosystem is semantically secure against passive eavesdroppers, assuming that decision-LWE$_{n,q,\chi,m+1}$ is hard. The proof proceeds very similarly to the one for Regev's system (see Section 5.2.1), but with some steps reordered. The two main ideas are: 1. a public key $\mathbf{A}$ is nearly uniformly random, so 2. a public key along with a ciphertext $\mathbf{c}$ (apart from the $\mu \cdot \lfloor \frac{q}{2} \rceil$ term) constitute a set of LWE samples, which are indistinguishable from uniform and hence hide the message.

Because many subsequent constructions build on the dual cryptosystem and its security proof, here we give a reasonably detailed outline, which again considers a sequence of hybrid experiments that produce a public key $\mathbf{A} \in \mathbb{Z}_q^{n \times (m+1)}$ and ciphertext $\mathbf{c} \in \mathbb{Z}_q^{m+1}$:

- In the first hybrid experiment, the public key $\mathbf{A}$ is chosen uniformly at random, instead of by the key-generation algorithm as $[\bar{\mathbf{A}} \mid \bar{\mathbf{A}}\mathbf{x}]$. The ciphertext $\mathbf{c}$ is generated by encrypting the bit μ under $\mathbf{A}$ in the usual way.

  This hybrid experiment is *statistically* indistinguishable from the real one. Just as in the previous proof, this follows from the fact that $m \approx n \log q$ is sufficiently large, and by the regularity lemma.

- In the next hybrid experiment, the public key $\mathbf{A}$ remains uniformly random, and now the ciphertext $\mathbf{c}$ is also chosen uniformly and independently of $\mathbf{A}$.

  This hybrid experiment is indistinguishable from the previous one, under the LWE assumption. This follows by a straightforward reduction: given $m+1$ samples $(\mathbf{A}; \mathbf{b}^t) \in \mathbb{Z}_q^{(n+1) \times (m+1)}$ drawn from either the LWE or uniform distribution, we can simulate either the previous hybrid experiment or this one (respectively) by outputting $\mathbf{A}$ as the public key and $\mathbf{c}^t = \mathbf{b}^t + (\mathbf{0}, \mu \cdot \lfloor \frac{q}{2} \rceil)^t$ as the ciphertext. (Note that in the uniform case, adding the fixed term preserves uniformity of $\mathbf{b}$.) So any hypothetical distinguisher for these two hybrid experiments would directly translate to one having the same advantage against the decision-LWE problem.

Because the above experiments are indistinguishable for any fixed bit μ, and the last one does not depend on μ at all, the two real experiments for $\mu = 0, 1$ are also indistinguishable.

Variants. A few variants of the dual LWE cryptosystem are worth briefly mentioning:

- As with Regev's system, there is an amortized version of the dual system that encrypts ℓ bits at a time. Here one uses a uniformly random secret key $\mathbf{X} \in \{0,1\}^{m \times \ell}$ and public key $\mathbf{A} = [\bar{\mathbf{A}} \mid \mathbf{U} = \bar{\mathbf{A}}\mathbf{X}] \in \mathbb{Z}_q^{n \times (m+\ell)}$; each bit of the message is then hidden by the corresponding (noisy) entry of $\mathbf{s}^t\mathbf{U}$.

- The entries of the secret key $\mathbf{X}$ need not be binary nor uniform, but may be chosen from any distribution on small integers such that $[\bar{\mathbf{A}} \mid \mathbf{U} = \bar{\mathbf{A}}\mathbf{X}]$ is statistically close to uniform. Most notably, the scheme can be made *identity-based* when the entries of $\mathbf{X}$ are chosen from, e.g., a discrete Gaussian over $\mathbb{Z}$. See Section 5.5.2 for further details.

5.2.3 More Compact LWE Cryptosystem

Lindner and Peikert [LP11] gave a public-key cryptosystem in which the public keys, and the secret keys and/or ciphertexts, are smaller than those in the above LWE-based schemes by a factor of about $\log q$, which is around ten or more for typical choices of parameters. This system adapts the code-based cryptosystem of Alekhnovich [Ale03] and the subset-sum-based cryptosystem of Lyubashevsky, Palacio, and Segev [LPS10], and their security proof strategies, to LWE. In particular, and in contrast to the systems described above, both the secret key and the encryption randomness (for a given public key and ciphertext) are *unique*.

Description of the system.

- Let $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times n}$ be a (possibly shared) uniformly random *square* matrix. A secret key is a vector $\mathbf{s} \in \mathbb{Z}^n$ with coordinates drawn independently from the error distribution χ, and the public key is

$$\mathbf{A} = \begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{b}^t \approx \mathbf{s}^t\bar{\mathbf{A}} \end{bmatrix} \in \mathbb{Z}_q^{(n+1) \times n},$$

  where the approximation hides independent errors drawn from χ. Notice that the secret and public keys satisfy the relation

$$(-\mathbf{s}, 1)^t \cdot \mathbf{A} \approx \mathbf{0} \pmod{q}. \tag{5.2.7}$$

- To encrypt a bit $\mu \in \{0,1\}$, one chooses an $\mathbf{r} \in \mathbb{Z}^n$ with coordinates drawn from the error distribution and outputs a ciphertext

$$\mathbf{c} \approx \mathbf{A}\mathbf{r} + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil) \in \mathbb{Z}_q^{n+1} \tag{5.2.8}$$

  where the approximation hides independent errors drawn from χ.

- To decrypt using the secret key $\mathbf{s}$, one computes

$$(-\mathbf{s}, 1)^t \cdot \mathbf{c} \approx (-\mathbf{s}, 1)^t \cdot \mathbf{A} \cdot \mathbf{r} + \mu \cdot \lfloor \tfrac{q}{2} \rceil \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil$$

  where the first approximation relies on Equation (5.2.8) and the shortness of $\mathbf{s}$, and the second relies on Equation (5.2.7) and the shortness of $\mathbf{r}$.

Similarly to the prior schemes, there is an amortized variant that can encrypt ℓ bits per ciphertext. Here the secret key is a matrix $\mathbf{S} \in \mathbb{Z}^{n \times \ell}$ of independent error terms, and the public key is $\begin{bmatrix} \bar{\mathbf{A}} \\ \mathbf{B}^t \approx \mathbf{S}^t\bar{\mathbf{A}} \end{bmatrix}$.

Analysis. The above scheme is semantically secure against passive eavesdroppers, assuming that decision-LWE is hard. The proof is a bit different than for the previously described schemes: it relies on the normal-form LWE assumption *twice*, and does not require a statistical regularity lemma. In more a bit detail, first notice that under the LWE assumption, the public key is indistinguishable from uniform. So consider a hybrid experiment in which the public key is uniformly random: then the public key and ciphertext (ignoring the $\mu \cdot \lfloor \frac{q}{2} \rceil$ term) together just constitute $n+1$ LWE samples, which are indistinguishable from uniform by assumption, so they hide the message bit.

Notice that if we wish to have a worst-case hardness guarantee from Theorem 4.2.4, the entries of both $\mathbf{r}, \mathbf{s}$ (which are drawn from χ) need to have magnitudes on the order of $\sqrt{n}$. Therefore, the accumulated error in the decryption relation is somewhat larger, by about a $\sqrt{n}$ factor, than in the prior LWE-based schemes. This induces a somewhat larger modulus q, and thereby a smaller LWE error rate for the public keys and ciphertexts, which yields somewhat looser approximation factors $\gamma = \tilde{O}(n^2)$ (versus $\tilde{O}(n^{3/2})$) for the underlying worst-case lattice problems. However, after normalizing to the same estimated hardness against concrete attacks, the compact scheme still has much smaller keys and ciphertexts; see [LP11] for further details.

5.2.4 Ring-LWE Cryptosystems

Using the correspondence between LWE and ring-LWE outlined in Section 4.4 above, the vast majority of LWE-based schemes and applications can be mechanically converted to more compact and efficient ones that remain secure under a corresponding ring-LWE assumption.

Compact ring-LWE encryption. The ring-LWE analogue of the compact LWE-based cryptosystem from Section 5.2.3 is relatively easy to describe and analyze.[3] This system was first described in [LPR10] for the simplest cyclotomic rings $R = \mathbb{Z}[X]/(X^n+1)$ for power-of-two n, and in full generality for arbitrary cyclotomic rings in [LPR13]. In brief, the system works as follows: a public key is a normal-form ring-LWE sample

$$(a, b \approx s \cdot a) \in R_q \times R_q,$$

for some (possibly shared) uniformly random $a \in R_q$ and short secret key $s \in R$, where both s and the approximation error are drawn from χ. To encrypt a message $\mu \in R_2$ (corresponding to an n-bit string), one generates a ciphertext

$$(u \approx a \cdot r, v \approx b \cdot r + \mu \cdot \lfloor \tfrac{q}{2} \rceil) \in R_q \times R_q,$$

where $r \in R$ and the approximation errors are drawn from χ. Decryption works by computing

$$v - s \cdot u \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil + b \cdot r - s \cdot a \cdot r \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil,$$

and recovering μ by removing the approximation error. (In the construction for general cyclotomic rings, obtaining the tightest parameters is somewhat subtle, and the system needs to be tweaked slightly; see [LPR13] for details.)

[3] Chronologically, the ring-LWE analogue was actually discovered *before* the LWE-based system, which was then "backported" from the former.

Ring-LWE meets NTRU. Stehlé and Steinfeld [SS11] gave a variant of the NTRU cryptosystem that has a security proof under the ring-LWE assumption. In their system, as in NTRU, the public key is a ratio $h = g/f \in R_q$ of two "somewhat short" polynomials. However, here the coefficients of f and g are chosen to have magnitudes of roughly $\sqrt{q} \cdot \mathrm{poly}(n)$, so that $g/f \in R_q$ is *statistically* very close to uniformly random. (By contrast, in NTRU the coefficients are very small, and g/f is conjectured to be *computationally* indistinguishable from uniform.) Proving this statistical fact is the main technical obstacle; once it is obtained, semantic security from ring-LWE follows relatively easily. In contrast with the ring-LWE cryptosystem described above, here public keys and ciphertexts are only *one* ring element each. However, in order to obtain the statistical property along with correctness, the parameters must be substantially larger, to the point where the scheme is concretely less efficient than other ring-LWE schemes.

Regularity. When adapting other LWE-based applications to ring-LWE, one subtlety is that certain schemes require an appropriate *regularity* (or "leftover hash") lemma for the ring-SIS function $f_{\vec{a}}$ (Equation (4.3.1)). Such lemmas are substantially harder to prove for rings than for additive groups like $\mathbb{Z}_q^n$, mainly because for typically used parameters, the ring R_q is very far from being a field, i.e., it has many zero divisors.[4] Micciancio [Mic02] gave the first such regularity lemma for rings like these, where q is a prime integer. One drawback is that proof depends on using "short" ring elements $z_i \in R$ that, when viewed as polynomials, have *uniformly random* integer coefficients in a $\mathrm{poly}(n)$-sized interval. However, many applications require or work best with *Gaussian-distributed* ring elements. A subsequent regularity lemma that works for this setting was given in [LPR13]. It relies on ring-SIS instances $\vec{a}$ in normal form, where the first R_q-component is unity, and it still requires the z_i to have polynomially large integer coefficients.

5.3 Actively Secure Encryption

The encryption schemes described in Section 5.2 are only semantically secure against *passive* eavesdroppers, i.e., indistinguishable under chosen-plaintext attack (IND-CPA). Many real-world applications require the much stronger notion of security against *active* attacks, formally known as indistinguishability under *chosen-ciphertext* attack (IND-CCA).

Fujisaki and Okamoto gave two generic, efficient methods [FO99a, FO99b] for converting any IND-CPA-secure public-key encryption scheme into an IND-CCA-secure one. However, their construction and analysis relies on the random-oracle heuristic [BR93], which is not sound in general (see, e.g., [CGH98]). Recently, an instantiation of the Fujisaki-Okamoto transformation for a particular compact ring-LWE-based cryptosystem was described in [Pei14].

In the rest of this subsection we describe two related paradigms for obtaining actively secure public-key encryption from lattices in the *standard* model, i.e., without the random-oracle heuristic.

5.3.1 From Lossy Trapdoor Functions

The work of Peikert and Waters [PW08] was the first to give a standard-model, IND-CCA-secure cryptosystem from any type of lattice assumption—in this case, LWE. Their construction is based around a concept they called a *lossy trapdoor function* family, or lossy TDF. In a lossy TDF family, the public key pk of a function $f_{pk} \colon X \to Y$ from the family can be generated in one of two ways: in *injective* mode, pk is generated along with a trapdoor sk, and f_{pk} is an injective function that can be efficiently inverted using sk. (In particular, the

[4] If R_q is a field or "nearly" so, then the standard leftover hash lemma is sufficient and yields good parameters.

range Y must be at least as large as the domain X.) In *lossy* mode, pk is generated without any trapdoor, and the function f_{pk} is "lossy" in the sense that its image $f_{pk}(X) \subseteq Y$ is much smaller than the domain X. In other words, f_{pk} has many collisions, and a typical $y = f_{pk}(x)$ has many valid preimages under f_{pk}. Finally, public keys generated in the two modes are *indistinguishable*: no efficient adversary has noticeable advantage in distinguishing between injective and lossy public keys. In particular, this final property can be used to show that inverting an injective function, or finding a collision in a lossy function, are both infeasible.

Peikert and Waters gave a generic construction of IND-CCA-secure encryption from any lossy TDF, and also constructed a lossy TDF from LWE (among other assumptions). An important property of their encryption scheme is that it is *witness recovering*: the decryption algorithm manages to recover the randomness that the encryption algorithm used to create the ciphertext, and then recomputes the ciphertext to verify that it is well formed. This technique departs from prior standard-model IND-CCA-secure constructions, in which ciphertexts essentially carry zero-knowledge proofs of well-formedness. The LWE-based lossy TDF construction of [PW08] has roughly the same asymptotic parameters and efficiency as the passively secure amortized encryption schemes described above in Section 5.2, but with somewhat larger constant and logarithmic factors.

5.3.2 From Injective Trapdoor Functions

Soon after [PW08], Peikert [Pei09] gave a somewhat simpler and more direct construction of IND-CCA-secure encryption from LWE. This construction does not rely on any lossiness properties, but is still witness recovering due to its use of the injective trapdoor functions defined in the work of Gentry, Peikert, and Vaikuntanathan [GPV08] (described below). The public key and ciphertext sizes of Peikert's scheme are roughly a linear factor larger than those in [PW08], due mainly to a more combinatorial (and less algebraic) description of the trapdoor functions.

Later, Micciancio and Peikert [MP12] gave a direct construction of IND-CCA-secure encryption from LWE, which has the best asymptotic and concrete parameters of all such constructions to date. The construction closely follows the one of [Pei09] based on injective TDFs, but with more compact, algebraic "tagged" functions, similar to those originally used in [PW08]. Here the tagged functions are obtained using a technique from [ABB10], which was originally developed in the context of identity-based encryption. See Sections 5.4.3 and 5.5 below for more details.

5.4 Lattice Trapdoors

Informally, a *trapdoor function* is a function that is easy to evaluate and hard to invert on its own, but which can be generated together with some extra "trapdoor" information that makes inversion easy. There are many versions of this basic concept, depending on whether the function in question is injective, surjective, bijective, "lossy," etc. The prototypical candidate trapdoor function is the RSA function [RSA78] $f_{N,e}(x) = x^e \bmod N$, where N is the product of distinct primes p, q, and $\gcd(e, \varphi(N)) = 1$. The RSA function is a bijection on $\mathbb{Z}_N^*$, and the trapdoor is $d = e^{-1} \bmod \varphi(N)$, because $(x^e)^d = x \bmod N$. (Alternatively, the factorization p, q of N is a trapdoor, because one can efficiently compute d from these factors.) There are relatively few trapdoor function candidates, and for the first few decades of modern cryptography, all commonly accepted ones relied on the conjectured hardness of integer factorization [RSA78, Rab79, Pai99].

Inspired by the early ideas of Goldreich, Goldwasser, and Halevi (hereafter GGH) [GGH97] (see Section 3.3), Gentry, Peikert, and Vaikuntanathan (hereafter GPV) [GPV08] showed that certain types of trapdoor functions can be constructed from lattice problems, and in particular (ring-)SIS/LWE. The work

of GPV and several follow-ups used these trapdoor functions to construct many powerful cryptographic applications, including digital signature schemes, (hierarchical) identity-based and attribute-based encryption, and much more.

In the rest of this subsection we give an overview of lattice trapdoors and the precise kinds of trapdoor functions they enable. Then in Section 5.5 we describe several cryptographic applications of these functions.

5.4.1 Trapdoor Functionality and Realizations

At a high level, a trapdoor for a lattice enables two main types of cryptographic functionality:

- The ability to solve the bounded-distance decoding problem (Definition 2.2.5), i.e., to find the unique lattice vector closest to a given target point that is promised to be "rather close" to the lattice. This can be used to construct injective trapdoor functions, as first described by GGH.
- The ability to sample from a discrete Gaussian distribution of "rather small" width, over any desired coset of the lattice. This can be used to construct what GPV called *preimage sampleable* trapdoor functions (PSFs), which yield digital signature schemes and other applications. Informally, a PSF is a regular trapdoor function f whose trapdoor allows one to *randomly sample* a preimage $x \in f^{-1}(y)$ of any range element y under some "canonical" (but not necessarily uniform) distribution. (See Definition 5.4.3 below for details.)

The literature contains two distinct but closely related notions for what constitutes a lattice trapdoor:

- The first notion, which was used in the seminal works [GGH97, GPV08, CHKP10], is a *short basis*: essentially, a lattice basis made up of relatively short lattice vectors. This notion is conceptually very natural and completely generic—it can be applied to any lattice—so we start in Section 5.4.2 by describing the high-level intuition and general-purpose algorithms in terms of short bases.
- The second is a *"gadget"-based* trapdoor, introduced in the work of Micciancio and Peikert [MP12] and described in Section 5.4.3 below. This notion applies only to *q-ary* lattices arising from the (ring-)SIS/LWE problems, so it is somewhat more specialized. But as compared with short bases, gadget trapdoors are technically simpler to work with in SIS/LWE-based applications, computationally more efficient, and at least as powerful. For these reasons, in later sections we describe most cryptographic schemes in terms of gadget trapdoors.

5.4.2 Short Bases

In this subsection we describe how a short basis can serve as a trapdoor for a public lattice specified by a "bad" basis (or other "hard" representation), and how this yields the trapdoor functionalities mentioned above.

Key generation. First we must consider how to jointly generate a pair of good and bad bases for the same lattice. Originally, GGH suggested an ad-hoc method that randomly chooses a set $\mathbf{S}$ of short linearly independent vectors to serve as the good basis of the lattice $\mathcal{L} = \mathcal{L}(\mathbf{S})$, then applies a "random" unimodular transformation $\mathbf{U}$ (having large entries) to obtain a bad basis $\mathbf{B} = \mathbf{S} \cdot \mathbf{U}$ of $\mathcal{L}$. Alternatively, Micciancio [Mic01] suggested using the Hermite normal form basis of $\mathcal{L}$, which is a "hardest possible" basis in a formal sense, as the public bad basis.

It is important to note that the above methods do *not* generate lattices from Ajtai's worst-case-hard family of q-ary SIS lattices (Equation (4.1.2)), so we lack evidence that these methods yield hard-on-average

problems, which is needed for security. Ajtai [Ajt99] addressed this gap by showing how to generate a random lattice from the SIS family along with a relatively short basis. Later works [AP09, MP12] simplified and improved this method to obtain nearly optimal bounds.

Theorem 5.4.1 ([Ajt99, AP09, MP12]). *There is an efficient randomized algorithm that, given positive integers n, q and $m \geq Cn \log q$ for some universal constant C, outputs a (nearly) uniformly random parity-check matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ specifying the integer lattice $\mathcal{L} = \mathcal{L}^\perp(\mathbf{A}) = \{\mathbf{x} \in \mathbb{Z}^m : \mathbf{A}\mathbf{x} = \mathbf{0} \pmod q\} \subseteq \mathbb{Z}^m$, along with a basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\mathcal{L}$ whose vectors have norms bounded by $\mathrm{poly}(n, \log q)$.*

Note that because $\mathbf{S} \subset \mathcal{L}^\perp(\mathbf{A})$, we have $\mathbf{A} \cdot \mathbf{S} = \mathbf{0} \pmod q$. Of course, the theorem is only interesting if the norm bound for $\mathbf{S}$ is significantly smaller than q (which is the case for typical parameters), because the scaled standard basis vectors $q \cdot \mathbf{e}_i$ are trivially in $\mathcal{L}^\perp(\mathbf{A})$. Finally, note that it is not sufficient to simply choose a uniformly random $\mathbf{A}$ and then attempt to compute short vectors in $\mathcal{L}^\perp(\mathbf{A})$, because this is exactly the (hard) SIS problem. Instead, one must somehow generate $\mathbf{S}$ along with $\mathbf{A}$, so that the induced distribution of $\mathbf{A}$ is close to uniform.

Injective trapdoor functions from bounded-distance decoding. At the heart of the GGH encryption scheme is a family of injective trapdoor functions, where inverting a function corresponds to solving the bounded-distance decoding problem (BDD, Definition 2.2.5) on the average. At a high level, the functions work as follows: as already mentioned, the public key and trapdoor are respectively a "hard" representation of a lattice $\mathcal{L}$ (e.g., a bad basis) and a short basis $\mathbf{S}$ of $\mathcal{L}$. Evaluating the function on a random input corresponds to choosing a "random" vector $\mathbf{v} \in \mathcal{L}^*$ in the *dual* lattice (see Section 2.2) and a short "error" vector $\mathbf{e}$ whose norm is significantly smaller than $\lambda_1(\mathcal{L}^*)$, and outputting $\mathbf{t} = \mathbf{v} + \mathbf{e}$.[5] (The reason for using the dual lattice will become clear in the next paragraph.) Because the error is sufficiently short, the inputs $\mathbf{v}, \mathbf{e}$ are uniquely defined by the output $\mathbf{t}$, and recovering them from $\mathbf{t}$ is exactly an average-case BDD problem on $\mathcal{L}^*$, which we may conjecture (or prove) to be hard.

Using a sufficiently short trapdoor basis $\mathbf{S}$ for $\mathcal{L}$, one can efficiently invert the function using a standard lattice decoding algorithm, such as naïve rounding or Babai's nearest-plane algorithm [Bab85]. For example, the rounding algorithm, given $\mathbf{t} = \mathbf{v} + \mathbf{e}$, simply outputs

$$\lfloor \mathbf{t}^t \cdot \mathbf{S} \rceil \cdot \mathbf{S}^{-1} = \lfloor (\mathbf{v}^t + \mathbf{e}^t) \cdot \mathbf{S} \rceil \cdot \mathbf{S}^{-1} = \mathbf{v}^t + \lfloor \mathbf{e}^t \cdot \mathbf{S} \rceil \cdot \mathbf{S}^{-1},$$

where the second equality holds because $\mathbf{v}^t \cdot \mathbf{S}$ is integral, since $\mathbf{v} \in \mathcal{L}^*$. So the algorithm correctly finds $\mathbf{v}$ as long as $\langle \mathbf{e}, \mathbf{s}_i \rangle \in [-\frac{1}{2}, \frac{1}{2})$ for every short basis vector $\mathbf{s}_i \in \mathbf{S}$. This indeed holds as long as both $\mathbf{e}$ and $\mathbf{s}_i$ are short enough, e.g., by the Cauchy-Schwarz inequality, or by tail bounds relying on the distribution of $\mathbf{e}$. (We mention that the nearest-plane algorithm can typically decode from somewhat larger errors using the same basis, but is computationally less efficient.)

The GPV instantiation of the above template, for which inverting the function is equivalent to the search-LWE problem (which, to recall, is an average-case BDD problem), is as follows:

- The public key is a uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, specifying the lattice $\mathcal{L} = \mathcal{L}^\perp(\mathbf{A})$, generated together with a relatively short trapdoor basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\mathcal{L}$, for which $\mathbf{A} \cdot \mathbf{S} = \mathbf{0} \pmod q$.

[5] Alternatively, the output can be (the canonical public representative of) the coset $\mathbf{e} + \mathcal{L}^*$, which eliminates the need for the (somewhat ill-defined) "random" lattice vector $\mathbf{v}$.

- As mentioned in Section 4.2.1, the LWE lattice $\mathcal{L}(\mathbf{A}) := \{\mathbf{A}^t\mathbf{s} : \mathbf{s} \in \mathbb{Z}_q^n\} + q\mathbb{Z}^m$ is the (scaled) dual lattice of $\mathcal{L}$, i.e., $\mathcal{L}(\mathbf{A}) = q \cdot \mathcal{L}^*$. So, following the above template, we define the LWE function

$$g_{\mathbf{A}}(\mathbf{s}, \mathbf{e}) := \mathbf{s}^t\mathbf{A} + \mathbf{e}^t \bmod q,$$

  where one should choose the input $\mathbf{s} \in \mathbb{Z}_q^n$ uniformly at random (corresponding to a "random" lattice vector $\mathbf{v} = \mathbf{A}^t\mathbf{s} \bmod q$) and $\mathbf{e} \in \mathbb{Z}^m$ from the LWE error distribution χ^m. Clearly, inverting this function is syntactically equivalent to the search-LWE$_{n,q,\chi,m}$ problem.

- Using the trapdoor $\mathbf{S}$, we can recover $\mathbf{s}, \mathbf{e}$ from the function output $\mathbf{b}^t = \mathbf{s}^t\mathbf{A} + \mathbf{e}^t \bmod q$ by computing

$$\mathbf{y}^t = \mathbf{b}^t \cdot \mathbf{S} = \mathbf{e}^t \cdot \mathbf{S} \pmod q,$$

  lifting $\mathbf{y}$ to its canonical representative $\bar{\mathbf{y}} \in [-\frac{q}{2}, \frac{q}{2})^m$, and outputting $\bar{\mathbf{y}}^t \cdot \mathbf{S}^{-1}$ (where $\mathbf{S}^{-1}$ is computed over the rationals, not modulo anything) as the desired value of $\mathbf{e}$, from which we can also compute $\mathbf{s}$. Note that as long as $\mathbf{e}^t \cdot \mathbf{S} \in [-\frac{q}{2}, \frac{q}{2})^m$—not modulo anything—which we can ensure by choosing q to be sufficiently large relative to the norms of $\mathbf{e}$ and the $\mathbf{s}_i$, then we do indeed have $\bar{\mathbf{y}}^t = \mathbf{e}^t \cdot \mathbf{S}$, and so we recover $\mathbf{e}$ correctly.

Discrete Gaussian sampling. In the GGH signature scheme, signing a message roughly corresponds to solving an approximate *closest vector problem* (CVP) on the average, which can be accomplished using a short basis. In contrast to the injective BDD-based functions described above, here the target can be an arbitrary point that need not be especially close to the lattice, and it has many valid signatures corresponding to sufficiently nearby lattice vectors. Recall, however, that the GGH signature scheme and some of its derivatives (e.g., [HHGP+03]) turned out to be *insecure* [NR06], because an attacker use a small number of signatures to efficiently reconstruct the secret trapdoor basis. This is because the signing algorithm implicitly leaks information about the geometry of the secret basis it uses.

The work of GPV gave a different, randomized approach to signing that provably leaks *no information* about the secret basis (apart from a bound on its length, which is already public knowledge). A key property is that the probability distribution of a signature is the same *no matter which short basis is used to produce it*, so signatures reveal nothing about the trapdoor basis (other than that it is sufficiently short). This fact facilitates a formal proof of unforgeability in the random-oracle model, assuming the average-case hardness of a CVP-like problem such as SIS.

A key technical ingredient behind the GPV approach is an algorithm for sampling from a discrete Gaussian distribution, given any sufficiently good basis. More precisely, letting $\widetilde{\mathbf{S}} = \{\widetilde{\mathbf{s}}_i\}$ denote the Gram-Schmidt orthogonalization of an ordered set of vectors $\mathbf{S} = \{\mathbf{s}_i\}$, and letting $\|\mathbf{B}\| := \max_i \|\mathbf{b}_i\|$ for any set of vectors $\mathbf{B} = \{\mathbf{b}_i\}$, we have:

Theorem 5.4.2 ([GPV08]). *There is a randomized polynomial-time algorithm that, given any basis $\mathbf{S}$ of a lattice $\mathcal{L} = \mathcal{L}(\mathbf{S}) \subset \mathbb{R}^n$, any coset $\mathbf{c} + \mathcal{L}$, and any Gaussian parameter $s \geq \|\widetilde{\mathbf{S}}\| \cdot \sqrt{\log O(n/\varepsilon)}$, outputs a sample whose distribution is within statistical distance ε of the discrete Gaussian $D_{\mathbf{c}+\mathcal{L},s}$.*

By setting ε to be, e.g., $\varepsilon = n^{-\omega(1)}$ for some slightly super-constant function $\omega(1)$, we obtain a negligible statistical distance from $D_{\mathbf{c}+\mathcal{L},s}$ for a parameter s that exceeds $\|\widetilde{\mathbf{S}}\|$ by only a small $\omega(\sqrt{\log n})$ factor.

For cryptographic purposes, the sampling algorithm from Theorem 5.4.2 can be treated as a black box. However, we briefly mention that the algorithm is simply a randomized variant of Babai's "nearest-plane" algorithm [Bab85], where in each iteration we randomly choose a "plane" according to a one-dimensional

discrete Gaussian over an appropriate coset $c + \mathbb{Z}$, instead of deterministically.[6] Subsequent works gave other sampling algorithms that offer various trade-offs among efficiency, statistical error, and width of the sampled distribution; we discuss some of these at the end of the subsection.

Preimage sampleable functions. With the above sampling algorithm in hand, we can describe GPV's notion of *preimage sampleable* trapdoor functions (PSFs), a generic lattice-based template, and a concrete instantiation from the SIS problem. We start with a semi-formal definition of PSFs.

Definition 5.4.3 (Preimage Sampleable Functions). An efficiently computable function $f\colon X \to Y$, where domain X is endowed with some efficiently sampleable distribution D, is *preimage sampleable* if there exists an efficient randomized algorithm, denoted f^{-1}, such that the following two methods of generating an input-output pair $(x, y = f(y))$ produce the *same joint distribution* (up to negligible statistical error):

- *Forward:* Choose $x \leftarrow D$ from the input distribution and let $y = f(x)$.
- *Reverse:* Choose $y \leftarrow Y$ uniformly from the range, then sample a preimage $x \leftarrow f^{-1}(y)$.

A PSF family is a collection of such functions for which we can efficiently sample (the description of) an f together with an f^{-1}.

As we shall see, the statistical property from the definition is central to the security of applications. We remark that any family of trapdoor *bijections* (e.g., the RSA function family) is also a PSF family: the preimage "sampling" algorithm is actually deterministic, and merely outputs the unique preimage.

The above definition does not include any *security* (or hardness) criteria, which are orthogonal to the sampling functionality and should therefore be considered separately. Typical properties we might ask for include *one-wayness* or *collision resistance*. The former says that it is infeasible, given the description of a sampled f (but not f^{-1}) and a uniformly random $y \leftarrow Y$, to find *any* preimage of y under f. The latter says that it is infeasible, given the description of a sampled f, to find distinct inputs x, x' such that $f(x) = f(x')$.

Lattice-based PSFs. We now describe GPV's generic template for PSFs based on lattices:

- As above, the public description and trapdoor of a function $f_{\mathcal{L}}$ are respectively a "hard" representation (e.g., a bad basis) of a lattice $\mathcal{L} \subset \mathbb{R}^m$, and a short basis $\mathbf{S}$ of $\mathcal{L}$. We also let s denote a public Gaussian parameter that exceeds the smoothing parameter of $\mathcal{L}$ and has an appropriate dependence on $\mathbf{S}$, e.g., $s \geq \|\widetilde{\mathbf{S}}\| \cdot \omega(\sqrt{\log n})$.

- The input distribution is the (continuous) Gaussian D_s over $\mathbb{R}^m$. Evaluating $f_{\mathcal{L}}$ on an input $\mathbf{x} \leftarrow D_s$ merely outputs (the canonical public representative of) the coset $\mathbf{x} + \mathcal{L}$, i.e., $\mathbf{y} = f_{\mathcal{L}}(\mathbf{x}) := \mathbf{x} \bmod \mathcal{L}$.

  We stress that $\mathbf{x}$ is rather *short* (i.e., $\|\mathbf{x}\| \leq s\sqrt{m}$) with overwhelming probability, but $\mathbf{x} \bmod \mathcal{L}$ is almost always represented by a *long* vector, due to the hard public representation of $\mathcal{L}$. Also, because s exceeds the smoothing parameter of $\mathcal{L}$, the output $\mathbf{y} = \mathbf{x} + \mathcal{L}$ is essentially uniform over the range $\mathbb{R}^m/\mathcal{L}$, as required. Lastly, finding a valid preimage for a uniform coset $\mathbf{y} + \mathcal{L}$ (i.e., violating one-wayness) means finding a sufficiently short vector in $\mathbf{y} + \mathcal{L}$, which is essentially an average-case approximate-CVP problem.

[6] Interestingly, Klein [Kle00] proposed essentially the same randomized variant of nearest-plane, but for the very different problem of bounded-distance decoding (BDD), and where the parameter s is smaller than the *minimal* length of the Gram-Schmidt vectors. For such parameters, the algorithm's output distribution is very far from a discrete Gaussian.

- To sample a preimage for a given coset $\mathbf{y} + \mathcal{L}$ using the trapdoor basis $\mathbf{S}$, we simply sample from the discrete Gaussian $D_{\mathbf{y}+\mathcal{L},s}$, e.g., using the algorithm from Theorem 5.4.2 (which requires $s \geq \|\widetilde{\mathbf{S}}\| \cdot \omega(\sqrt{\log n})$). A moment's thought reveals that because the set of preimages is exactly $\mathbf{y} + \mathcal{L}$, and the conditional probability of each $\mathbf{x} \in \mathbf{y} + \mathcal{L}$ under D_s is proportional to $\rho_s(\mathbf{x})$, the input distribution D_s *conditioned on* obtaining output $\mathbf{y} + \mathcal{L}$ is exactly $D_{\mathbf{y}+\mathcal{L},s}$. Combining this with the (near) uniformity of $\mathbf{y} + \mathcal{L} \in \mathbb{R}^m/\mathcal{L}$, we see that the preimage sampling algorithm does indeed satisfy the statistical property from Definition 5.4.3.

A concrete instantiation of the above template, whose one-wayness and collision resistance follow directly from the hardness of the SIS problem, is as follows. It works with integers in place of real numbers, and hence $D_{\mathbb{Z}^m,s}$ in place of D_s.

- As before, the public key is a uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$, specifying the lattice $\mathcal{L} = \mathcal{L}^\perp(\mathbf{A})$ (Equation (4.1.2)), generated together with a relatively short trapdoor basis $\mathbf{S} \in \mathbb{Z}^{m \times m}$ of $\mathcal{L}$.

- The function $f_\mathcal{L}$ is simply the SIS function (Equation (4.1.1))

$$f_\mathbf{A}(\mathbf{x}) := \mathbf{A}\mathbf{x} \bmod q,$$

  where here the input $\mathbf{x} \in \mathbb{Z}^m$ is chosen according to the discrete Gaussian $D_{\mathbb{Z}^m,s}$. Recall from Section 4.1.1 that $\mathbf{y} = \mathbf{A}\mathbf{x} \in \mathbb{Z}_q^n$ can be seen as a canonical representative of the coset $\mathcal{L}_\mathbf{y}^\perp(\mathbf{A}) = \mathbf{x} + \mathcal{L}^\perp(\mathbf{A}) \in \mathbb{Z}^m/\mathcal{L}$. Therefore, finding a short preimage of $\mathbf{y}$ is syntactically identical to the (inhomogeneous) SIS problem. Similarly, a valid collision $\mathbf{x}, \mathbf{x}' \in \mathbb{Z}^m$ immediately yields a short SIS solution $\mathbf{z} = \mathbf{x} - \mathbf{x}' \neq \mathbf{0}$.

- To sample a preimage of $\mathbf{y} \in \mathbb{Z}_q^n$ using the trapdoor basis $\mathbf{S}$, we merely sample from the discrete Gaussian $D_{\mathcal{L}_\mathbf{y}^\perp(\mathbf{A}),s}$ using the algorithm from Theorem 5.4.2 (or any other suitable algorithm).

Extending and randomizing short bases. Cash, Hofheinz, Kiltz, and Peikert [CHKP10] demonstrated additional useful features of lattice trapdoors, namely, that they can be *extended* and *re-randomized*. These properties were used to construct digital signature and IBE schemes without random oracles, as well as *hierarchical* versions of the same, in which secret keys with restricted power can be securely delegated to subordinates. See Section 5.5.3 for further details on these constructions.

A first main idea in [CHKP10] is that a trapdoor short basis $\mathbf{S}$ for a parity-check matrix $\mathbf{A}$ can yield an equally good trapdoor basis for any extension $\mathbf{A}' = [\mathbf{A} \mid \mathbf{A}_1]$, for arbitrary $\mathbf{A}_1$. Specifically, letting $\mathbf{W}$ be any integral solution to $\mathbf{A}\mathbf{W} = -\mathbf{A}_1 \pmod q$, we have

$$[\mathbf{A} \mid \mathbf{A}_1] \cdot \underbrace{\begin{bmatrix} \mathbf{S} & \mathbf{W} \\ \mathbf{0} & \mathbf{I} \end{bmatrix}}_{\mathbf{S}'} = \mathbf{0} \pmod q.$$

It is not hard to show that $\mathbf{S}'$ is indeed a basis of $\mathcal{L}^\perp(\mathbf{A}')$, and that its Gram-Schmidt vectors $\widetilde{\mathbf{S}}$ are no longer than those of $\mathbf{S}$, because $\widetilde{\mathbf{S}}' = \mathrm{diag}(\widetilde{\mathbf{S}}, \mathbf{I})$.

A second main idea is that a trapdoor basis $\mathbf{S}$ for $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ can be re-randomized to a new one $\mathbf{S}' \in \mathbb{Z}^{m \times m}$ that reveals nothing about $\mathbf{S}$ (except for a bound on its quality). This works simply by using $\mathbf{S}$ to sample independent Gaussian solutions to $\mathbf{A} \cdot \mathbf{s}' = \mathbf{0}$ (i.e., lattice vectors in $\mathcal{L}^\perp(\mathbf{A})$) until we have accumulated a set of m linearly independent vectors, which can then be converted to a basis. Note that the quality of the resulting basis $\mathbf{S}'$ is about a $\sqrt{m}$ factor worse than that of the original $\mathbf{S}$, because the norm of each sampled solution $\mathbf{s}'$ is roughly $\|\widetilde{\mathbf{S}}\| \cdot \sqrt{m}$.

Other sampling algorithms. Since the introduction of discrete Gaussian sampling for cryptographic purposes [GPV08], many refinements and alternative algorithms have been proposed; we discuss some of them here. (This material is not needed for the rest of the survey, and may be safely skipped.)

The randomized nearest-plane algorithm for discrete Gaussian sampling described in Theorem 5.4.2 is rather inefficient in general: it requires very high-precision arithmetic to store and operate on (an appropriate representation of) the Gram-Schmidt orthogonalized basis, and it involves n sequential iterations, each of which requires an n-dimensional inner product of high-precision vectors. Also, this super-quadratic runtime persists to the ring setting (e.g., for ring-SIS/LWE), where by contrast all other commonly used ring operations are only quasi-linear time.

Peikert [Pei10] gave a more efficient and parallel discrete Gaussian sampling algorithm, which is also quasi-linear time in the ring setting. As an illustrative first attempt, to sample in parallel from $\mathbf{c} + \mathcal{L}$ using a short basis $\mathbf{S}$ of $\mathcal{L}$, one might try randomly rounding each coefficient of the basis vectors *independently*, i.e., let $\mathbf{x} \leftarrow \mathbf{S} \cdot D_{\mathbf{S}^{-1}\mathbf{c}+\mathbb{Z}^n,r}$ for some appropriate $r \geq \eta_\varepsilon(\mathbb{Z}^n)$. This produces a short random element of $\mathbf{c} + \mathcal{L}$, but unfortunately, the distribution is "skewed" due to the multiplication by $\mathbf{S}$. More formally, it is a *non-spherical* discrete Gaussian with covariance matrix $\mathbb{E}[\mathbf{x} \cdot \mathbf{x}^t] \approx r^2 \cdot \mathbf{S}\mathbf{S}^t$, which reveals a lot about $\mathbf{S}$.

The main idea in [Pei10] is to add an appropriately distributed *perturbation* term to "unskew" the distribution of $\mathbf{x}$, making it spherical.[7] More formally, to sample from $D_{\mathbf{c}+\mathcal{L},s}$ we do the following:

1. In an offline phase (before the desired coset $\mathbf{c} + \mathcal{L}$ is known), choose a Gaussian perturbation $\mathbf{p}$ with covariance $\Sigma_\mathbf{p} = s^2\mathbf{I} - r^2\mathbf{S}\mathbf{S}^t$.

2. In the online phase (when $\mathbf{c} + \mathcal{L}$ is known), output $\mathbf{x} \leftarrow \mathbf{p} + \mathbf{S} \cdot D_{\mathbf{S}^{-1}(\mathbf{c}-\mathbf{p})+\mathbb{Z}^n,r}$.

Notice that the support of $\mathbf{x}$ is $\mathbf{c} + \mathcal{L}$, as desired. Moreover, a "convolution" lemma from [Pei10] essentially says that, just as for *continuous* Gaussians, the covariances of discrete Gaussian addends are *additive*, thus $\mathbf{x}$ is discrete Gaussian with the desired covariance $s^2\mathbf{I}$. Finally, note that the distribution of the perturbation term $\mathbf{p}$ is well-defined exactly when $\Sigma_\mathbf{p}$ is positive definite, which holds if and only if $s > r \cdot s_1(\mathbf{S})$, where $s_1(\mathbf{S})$ denotes the largest singular value (or spectral norm) of $\mathbf{S}$. Therefore, the final Gaussian parameter of the sampler is proportional to $s_1(\mathbf{S})$, rather than $\|\widetilde{\mathbf{S}}\|$ as in Theorem 5.4.2.

Subsequently, Ducas and Nguyen [DN12a] gave a technique that optimizes (up to logarithmic factors) the asymptotic average-case runtime of the randomized nearest-plane algorithm, and the offline phase of Peikert's convolutional sampler, using floating-point arithmetic (FPA). Their method combines low and high-precision FPA via a "laziness" technique that typically remains low precision, and in practice can be implemented using standard IEEE double precision.

In a different direction, Brakerski *et al.* [BLP+13] gave an improved version of the randomized nearest-plane algorithm that uses rejection sampling to sample discrete Gaussians with a parameter as small as $\|\widetilde{\mathbf{S}}\| \cdot O(\sqrt{\log n})$ (saving a slightly super-constant $\omega(1)$ factor over Theorem 5.4.2), and with a statistical error that can be made arbitrarily close to zero. However, these sharper bounds come at the expense of somewhat larger runtime and algorithmic complexity.

Recently, Lyubashevsky and Wichs [LW15] gave algorithms for sampling short vectors from lattice cosets under distributions other than discrete Gaussians, e.g., uniform within a box. Such samplers can be used to provide alternative instantiations of preimage sampleable functions. Compared to prior discrete Gaussian samplers, the primary advantage of these new samplers is their relative simplicity of implementation and avoidance of high-precision arithmetic. Their primary disadvantage is that the vectors they produce are longer

[7] This idea of using a perturbation to hide information about $\mathbf{S}$ is loosely inspired by, but technically very different from, another perturbation technique included as part of NTRUSign [HHGP+03], but which was shown to be insecure in [MPSW09, Wan10, DN12b].

by a factor up to linear in the lattice dimension, which corresponds to worse security (and thus larger keys to compensate).

Finally, Ducas and Prest [DP15] recently investigated a "hybrid" of the nearest-plane and convolutional samplers for lattices defined over rings (e.g., NTRU lattices). They showed that the hybrid approach can simultaneously obtain nearly the best aspects of both previous algorithms, in terms of sample quality and asymptotic efficiency.

5.4.3 Gadget Trapdoors

We now turn from short bases as generic lattice trapdoors, to the "gadget"-based trapdoors for q-ary SIS/LWE lattices as developed in [MP12], building on techniques from several related works [Ajt99, AP09, PW08, ABB10].

Gadgets. The first main idea is that there are special, structured matrices $\mathbf{G}$, called "gadgets," for which solving SIS and LWE is *easy*. The simplest gadget is as follows: for a given a modulus q, define the vector

$$\mathbf{g} = (1, 2, 4, \ldots, 2^{\ell-1}) \in \mathbb{Z}_q^\ell,$$

where $\ell = \lceil \log_2 q \rceil$.[8] Now observe that given any $u \in \mathbb{Z}_q$, we can find a short solution $\mathbf{x} \in \mathbb{Z}^m$ to

$$\langle \mathbf{g}, \mathbf{x} \rangle = \mathbf{g}^t \cdot \mathbf{x} = u \pmod q;$$

for example, let $\mathbf{x}$ correspond to the base-two representation of u's canonical representative in $\{0, \ldots, q-1\}$. In other words, it is easy to solve the SIS problem with respect to the one-row parity-check "matrix" $\mathbf{g}^t$. Moreover, it is easy to randomly sample from a discrete Gaussian over the lattice coset $\mathcal{L}_u^\perp(\mathbf{g}^t)$ of solutions.

A bit more thought reveals that solving LWE with respect to $\mathbf{g}^t$ is also easy. For simplicity, suppose that $q = 2^\ell$; then given any $\mathbf{b}^t \approx s \cdot \mathbf{g}^t \pmod q$ with errors in the interval $[-\frac{q}{4}, \frac{q}{4})$, we can recover the most-significant bit of $s \in \mathbb{Z}_q$ from

$$b_\ell \approx s \cdot 2^{\ell-1} = \mathrm{msb}(s) \cdot \tfrac{q}{2} \pmod q.$$

We can then recover the next-most-significant bit of s from $b_{\ell-1} \approx s \cdot 2^{\ell-2} \pmod q$, etc. (This all generalizes to non-power-of-two moduli as well, but is a bit more technically involved.)

We extend all of the above to n-dimensional SIS and LWE with respect to the block-diagonal *gadget matrix*

$$\mathbf{G} := \mathbf{I}_n \otimes \mathbf{g}^t = \mathrm{diag}(\mathbf{g}^t, \ldots, \mathbf{g}^t) \in \mathbb{Z}_q^{n \times n\ell},$$

just by working blockwise: for LWE, given $\mathbf{b}^t \approx \mathbf{s}^t\mathbf{G} = (s_1\mathbf{g}, \ldots, s_n\mathbf{g})^t$, just recover each s_i from the corresponding block of $\mathbf{b}$. For SIS, to find a short solution to $\mathbf{G}\mathbf{x} = \mathbf{u}$ for a given $\mathbf{u} \in \mathbb{Z}_q^n$, just find short solutions to $\mathbf{g}^t \cdot \mathbf{x}_i = u_i$ for each entry u_i and concatenate the results. It is convenient to express this operation in terms of the "decomposition" function

$$\begin{aligned} \mathbf{G}^{-1} \colon \mathbb{Z}_q^n &\to \mathbb{Z}^{n\ell} \\ \mathbf{u} &\mapsto \text{a short solution } \mathbf{x} \text{ to } \mathbf{G}\mathbf{x} = \mathbf{u} \pmod q, \end{aligned} \tag{5.4.1}$$

[8] This $\mathbf{g}$ can be generalized in a variety of ways, e.g., by using a base other than two, mixed bases, the Chinese Remainder Theorem, etc.

where the choice of notation is explained by the relation

$$\mathbf{G} \cdot \mathbf{G}^{-1}(\mathbf{u}) = \mathbf{u} \pmod{q}.$$

We stress that $\mathbf{G}^{-1}$ is not a *matrix*, but rather a *function* that maps a mod-q input to a short integer vector. Depending on the context, the $\mathbf{G}^{-1}$ operation may even be randomized, e.g., it could sample a Gaussian-distribution solution.

Finally, observe that given any invertible matrix $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$, SIS and LWE remain easy with respect to the matrix $\mathbf{HG} \in \mathbb{Z}_q^{n \times n\ell}$. Specifically, to solve $(\mathbf{HG})\mathbf{x} = \mathbf{u}$, simply output $\mathbf{x} = \mathbf{G}^{-1}(\mathbf{H}^{-1} \cdot \mathbf{u})$, and given $\mathbf{b}^t \approx \mathbf{s}^t\mathbf{HG}$, first recover $\mathbf{s}^t\mathbf{H}$ by solving with respect to $\mathbf{G}$, then multiply the solution by $\mathbf{H}^{-1}$.

Trapdoor definition and usage. The next main idea is that a parity-check matrix $\mathbf{A}$ can "hide" (a multiple of) the public gadget matrix $\mathbf{G}$. A *trapdoor* is merely a short linear transform that unhides the gadget.

Definition 5.4.4. A *trapdoor* for a parity-check matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ is any sufficiently "short" integer matrix $\mathbf{R} \in \mathbb{Z}^{m \times n\ell}$ such that

$$\mathbf{AR} = \mathbf{HG} \pmod{q} \tag{5.4.2}$$

for some invertible $\mathbf{H} \in \mathbb{Z}_q^{n \times n}$, called the *tag* of the trapdoor. The *quality* of the trapdoor is given by its largest singular value $s_1(\mathbf{R}) := \max_{\|\mathbf{u}\|=1} \|\mathbf{Ru}\|$; smaller $s_1(\mathbf{R})$ means higher quality.

Note that the tag $\mathbf{H}$ is easily computable from $\mathbf{A}$ and $\mathbf{R}$, because $\mathbf{G}$ contains the identity $\mathbf{I}_n$ as a submatrix.

Suppose we have a trapdoor $\mathbf{R}$ for $\mathbf{A}$ with tag $\mathbf{H}$, for the moment putting aside the question of how we construct such matrices. Then we can efficiently solve SIS with respect to $\mathbf{A}$, i.e., for any $\mathbf{u} \in \mathbb{Z}_q^n$ we can find a short solution $\mathbf{x} \in \mathbb{Z}^m$ to $\mathbf{Ax} = \mathbf{u} \pmod{q}$. To do this, simply let

$$\mathbf{x} = \mathbf{Rw}, \quad \text{where} \quad \mathbf{w} = \mathbf{G}^{-1}(\mathbf{H}^{-1}\mathbf{u}).$$

Clearly, $\mathbf{x}$ is short because $\mathbf{R}$ and $\mathbf{w}$ are (specifically, $\|\mathbf{x}\| \le s_1(\mathbf{R}) \cdot \|\mathbf{w}\|$), and by Equation (5.4.2),

$$\mathbf{Ax} = \mathbf{ARw} = \mathbf{HGw} = \mathbf{u} \pmod{q}.$$

Sampling a *Gaussian-distributed* SIS solution is a bit more subtle. In the above method we could just sample a Gaussian-distributed solution $\mathbf{w}$, but then the distribution of $\mathbf{x} = \mathbf{Rw}$ would be "skewed" by, and hence leak information about, the trapdoor $\mathbf{R}$. To correct for this skew we can use the "convolution" technique of [Pei10] (described in the final part of Section 5.4.2 above): first choose an appropriately distributed perturbation vector $\mathbf{p} \in \mathbb{Z}^m$, then sample a Gaussian $\mathbf{w} \leftarrow \mathbf{G}^{-1}(\mathbf{H}^{-1} \cdot (\mathbf{u} - \mathbf{Ap}))$, and finally output $\mathbf{x} = \mathbf{p} + \mathbf{Rw}$. By letting $\mathbf{p}$ have the proper distribution, the resulting distribution of $\mathbf{x}$ can be made a discrete Gaussian of any desired parameter exceeding $\approx s_1(\mathbf{R})$.

Similarly, we can use a trapdoor to solve LWE with respect to $\mathbf{A}$. Given $\mathbf{b}^t \approx \mathbf{s}^t\mathbf{A}$, we simply transform it to $\mathbf{b}^t\mathbf{R} \approx \mathbf{s}^t\mathbf{AR} = \mathbf{s}^t\mathbf{HG}$, then find $\mathbf{s}$ by solving LWE with respect to $\mathbf{HG}$, as described above. Note that from the first approximation to the second, the error is expanded by up to $s_1(\mathbf{R})$, so the error we can handle with respect to $\mathbf{A}$ is about an $s_1(\mathbf{R})$ factor smaller than the error we can handle with respect to $\mathbf{G}$.

Trapdoor generation. We now describe how to construct a (nearly) uniformly random parity-check matrix $\mathbf{A}$ together with a trapdoor $\mathbf{R}$ having a desired tag $\mathbf{H}$. We start by choosing a uniformly random

$\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times \bar{m}}$ for sufficiently large $\bar{m}$ (where $\bar{\mathbf{A}}$ can be shared among many matrices and trapdoors), and a short random integer matrix $\bar{\mathbf{R}} \in \mathbb{Z}^{\bar{m} \times n\ell}$, e.g., one having discrete Gaussian entries. We then let

$$\mathbf{A} = [\bar{\mathbf{A}} \mid \mathbf{HG} - \bar{\mathbf{A}}\bar{\mathbf{R}}] \in \mathbb{Z}_q^{n \times m},$$

where $m = \bar{m} + n\ell$.

By construction, it is immediate that $\mathbf{R} = \left[\begin{smallmatrix} \bar{\mathbf{R}} \\ \mathbf{I} \end{smallmatrix}\right]$ is a trapdoor for $\mathbf{A}$ with tag $\mathbf{H}$. Second, by standard bounds from random matrix theory (see, e.g., [Ver12]), with very high probability the spectral norm $s_1(\mathbf{R})$ is small, e.g., it is $O(s \cdot (\sqrt{\bar{m}} + \sqrt{n\ell}))$ when the entries of $\bar{\mathbf{R}}$ are subgaussian with parameter s. Finally, for appropriate choices of $\bar{m} \approx n \log q$ and distributions of $\bar{\mathbf{R}}$, the matrix $\mathbf{A}$ is statistically very close to uniform over the choice of $\bar{\mathbf{A}}, \bar{\mathbf{R}}$. In particular, the choice of tag $\mathbf{H}$ is statistically hidden by $\mathbf{A}$ itself, which is an important fact in the security proofs for many applications described later.

Finally, we mention that it is easy to adapt the above construction to get a smaller parity-check matrix that is *computationally* pseudorandom under the normal-form LWE assumption, by using $[\mathbf{I}_n \mid \bar{\mathbf{A}}]$ for a uniformly random *square* $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times n}$ in place of the rectangular matrix $\bar{\mathbf{A}}$ above.

Puncturing. The above construction admits an algebraic "trapdoor puncturing" technique, which is a main tool in the LWE-based lossy trapdoor functions of [PW08] and the compact (H)IBE of [ABB10], among many other works. Notice that in the above construction, $\mathbf{R} = \left[\begin{smallmatrix} \bar{\mathbf{R}} \\ \mathbf{I} \end{smallmatrix}\right]$ remains a trapdoor with tag $\mathbf{H} - \mathbf{H}'$ for the matrix

$$\mathbf{A}_{\mathbf{H}'} := \mathbf{A} - [\mathbf{0} \mid \mathbf{H}'\mathbf{G}] = [\bar{\mathbf{A}} \mid (\mathbf{H} - \mathbf{H}')\mathbf{G} - \bar{\mathbf{A}}\bar{\mathbf{R}}].$$

By taking $\mathbf{H}, \mathbf{H}'$ from a family of matrices in $\mathbb{Z}_q^{n \times n}$ having *invertible differences*—i.e., the difference between any two distinct matrices is invertible (over $\mathbb{Z}_q$)—we see that the matrices $\mathbf{A}_{\mathbf{H}'}$ are "punctured" in the sense that $\mathbf{R}$ is a trapdoor for *all but one* of them, namely, $\mathbf{A}_{\mathbf{H}} = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\bar{\mathbf{R}}]$.

A family of matrices having the invertible differences property can be constructed, e.g., from the finite field $\mathbb{F}_{q^n}$ where q is prime, as follows. View $\mathbb{F}_{q^n}$ as an n-dimensional vector space over $\mathbb{F}_q = \mathbb{Z}_q$ with some fixed basis, and let each $h \in \mathbb{F}_{q^n}$ correspond to the n-by-n matrix of the $\mathbb{F}_q$-linear transformation representing multiplication by h. Because this correspondence is an injective ring homomorphism from $\mathbb{F}_{q^n}$ to $\mathbb{Z}_q^{n \times n}$, and every nonzero field element is invertible, this family has invertible differences.

Extending and randomizing trapdoors. We conclude this coverage of gadget trapdoors by mentioning that, just as with short-basis trapdoors, there are very simple methods for extending and re-randomizing gadget trapdoors, which are useful in hierarchical IBE and other applications.

Given a trapdoor $\mathbf{R}$ for $\mathbf{A}$, we have the following:

- The matrix $\left[\begin{smallmatrix} \mathbf{R} \\ \mathbf{0} \end{smallmatrix}\right]$ is obviously a trapdoor (having the same tag as $\mathbf{R}$) for any extension $\mathbf{A}' = [\mathbf{A} \mid \mathbf{A}_1]$.
- It is easy to generate a new random trapdoor $\mathbf{R}'$, having any desired tag $\mathbf{H}'$, that reveals nothing about $\mathbf{R}$ (apart from an upper bound on $s_1(\mathbf{R})$), simply by using $\mathbf{R}$ to sample a Gaussian solution to $\mathbf{AR}' = \mathbf{H}'\mathbf{G}$.
- Alternatively, to generate a random trapdoor of the form $\left[\begin{smallmatrix} \star \\ \mathbf{I} \end{smallmatrix}\right]$ (which is needed for the puncturing technique described above), for an arbitrary extension $\mathbf{A}' = [\mathbf{A} \mid \mathbf{A}_1]$ we can sample a Gaussian-distributed solution $\mathbf{R}'$ to

  $$\mathbf{AR}' = \mathbf{H}'\mathbf{G} - \mathbf{A}_1.$$

  Then $\left[\begin{smallmatrix} \mathbf{R}' \\ \mathbf{I} \end{smallmatrix}\right]$ is a trapdoor for $\mathbf{A}'$ with tag $\mathbf{H}'$.

5.5 Trapdoor Applications: Signatures, ID-Based Encryption, and More

Here we describe some cryptographic applications, including signature and identity-based encryption schemes, that immediately arise from the trapdoor functions and techniques described in the previous subsection. (We cover more advanced trapdoor applications in Section 6.)

5.5.1 Signatures

At a high level, the GPV signature scheme is obtained by plugging the lattice-based preimage sampleable functions from Section 5.4 into the classical "hash-and-sign" paradigm, which dates back to the early days of public-key cryptography [DH76, RSA78].[9]

- The public and secret keys are respectively a "hard" public description of, and trapdoor for, an m-dimensional lattice $\mathcal{L}$.
- To sign, one hashes the message to a lattice coset $\mathbf{y} + \mathcal{L}$ using an appropriate public function, then uses the trapdoor to sample a signature $\mathbf{x} \leftarrow D_{\mathbf{y}+\mathcal{L},s}$. Here $s \geq \eta(\mathcal{L})$ is a public parameter of the scheme determined by the concrete algorithms used to generate the trapdoor and sample discrete Gaussians.[10]

  Note that by standard tail bounds, $\|\mathbf{x}\| \leq s\sqrt{m}$ except with exponentially small probability.
- To verify a purported signature $\mathbf{x}$ on a message, one hashes the message to obtain the coset $\mathbf{y} + \mathcal{L}$, and simply checks that $\mathbf{x} \in \mathbf{y} + \mathcal{L}$ and that $\|\mathbf{x}\| \leq s\sqrt{m}$.

A concrete instantiation of the scheme uses the q-ary SIS lattice $\mathcal{L} = \mathcal{L}^{\perp}(\mathbf{A})$ and function $f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x}$ (for uniformly random $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$) as the underlying preimage sampleable function, as described in Sections 5.4.2 and 5.4.3.

By modeling the public hash function as a "programmable" random oracle, as is typical for the hash-and-sign paradigm, one can prove the following informally stated theorem:

Theorem 5.5.1. *The above scheme is unforgeable under chosen-message attack (in the random-oracle model), assuming the average-case hardness of finding a nonzero vector of norm at most $2s\sqrt{m}$ in a random lattice $\mathcal{L}$ as produced by the key generator (i.e., SIS with norm bound $2s\sqrt{m}$ for the concrete instantiation).*

The proof of the theorem works by demonstrating a reduction that uses a hypothetical forger to find a short vector in a given random lattice $\mathcal{L}$. The main idea is that by the statistical property in Definition 5.4.3, the reduction can generate properly distributed signature-hash pairs *without needing a trapdoor for $\mathcal{L}$*. Specifically, it can evaluate the function in the *forward* direction, rather than using the *reverse* direction and trapdoor, as the real scheme does. Moreover, the reduction can obtain a short vector in $\mathcal{L}$ as the difference between a forged signature and its own internally prepared signature for the same message.

In a bit more detail, the reduction works as follows: for each message μ on which the forger queries the hash function H (including all signing queries and the eventual forged message), the reduction internally chooses an $\mathbf{x} \leftarrow D_s$ and "programs" the hash function as $H(\mu) := \mathbf{y} + \mathcal{L} = \mathbf{x} + \mathcal{L}$, where $\mathbf{y}$ is the canonical

[9] We also mention that the properties of preimage sampleable functions and the GPV signature scheme were also used to give *noninteractive statistical zero-knowledge proofs*, with efficient provers, for several (worst-case) lattice problems [PV08].

[10] The very observant reader may notice that when signing the same message multiple times, we should never give out more than one signature for the same coset $\mathbf{y} + \mathcal{L}$, because it would reveal short vectors in $\mathcal{L}$. This issue can be addressed in a few standard ways, either by hashing with randomness to obtain distinct cosets for repeated messages, or by using state or a pseudorandom function to obtain "repeatable randomness," so that we always produce the same signature when signing the same message.

public representative of $\mathbf{x} + \mathcal{L}$. Because $s \geq \eta_\varepsilon(\mathcal{L})$, we know that $\mathbf{y} + \mathcal{L}$ is a (nearly) uniformly random coset of $\mathcal{L}$, thus the programmed oracle H behaves as a random oracle. When the forger asks for a signature on a message μ, the reduction simply returns the corresponding $\mathbf{x}$, which by construction is distributed as $D_{H(\mu)+\mathcal{L},s}$, just as in the real scheme (up to negligible statistical error). Because all queries are answered with the same distributions as in the real system, with noticeable probability the forger must eventually output a forgery $(\mu^*, \mathbf{x}^*)$. This means that $\mathbf{x}^* \in H(\mu^*) + \mathcal{L}$ and is sufficiently short, and the forger never saw the reduction's internally generated signature $\mathbf{x} \in H(\mu^*) + \mathcal{L}$ for μ^*. The reduction simply outputs the short vector $\mathbf{x} - \mathbf{x}^* \in \mathcal{L}$, which with high probability is nonzero because the value of $\mathbf{x}$ is statistically hidden from the forger's view. (Note that this proof strategy yields a very "tight" reduction, i.e., the reduction succeeds with almost exactly the same probability as the forger itself.)

5.5.2 Identity-Based Encryption

Combining GPV signatures with the "dual" LWE cryptosystem (Section 5.2.2) in its multi-user form immediately yields an *identity-based encryption* (IBE) scheme (see the end of Section 2.4 for a description of how IBE operates). Note that the GPV system and its derivatives are the only known IBEs that are conjectured to be secure against quantum attacks; all others are based on the quadratic residuosity assumption (e.g., [Coc01, BGH07]) or on bilinear pairings (e.g., [BF01]).

The main idea behind the GPV IBE is that instead of users generating their own Gaussian-distributed secret keys $\mathbf{x} \in \mathbb{Z}^m$ and corresponding public keys $\mathbf{u} = f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} \in \mathbb{Z}_q^n$, a public key is merely the hash of the user's identity string into $\mathbb{Z}_q^n$, and the corresponding secret key is generated by the authority as a GPV signature on the identity. In more detail:

- The master key is a uniformly random parity-check matrix $\mathbf{A}$, and the master secret key is a trapdoor for $\mathbf{A}$. Also, a public hash function maps each user identity id to a syndrome $\mathbf{u}_{id} \in \mathbb{Z}_q^n$, which serves as part of the user-specific public key $\mathbf{A}_{id} := [\mathbf{A} \mid \mathbf{u}_{id}]$.

- To extract a secret key for identity id, we use the trapdoor for $\mathbf{A}$ to sample a Gaussian-distributed solution to $\mathbf{A}\mathbf{x}_{id} = \mathbf{u}_{id}$, i.e., choose $\mathbf{x}_{id}$ from a discrete Gaussian over the coset $\mathcal{L}^\perp_{\mathbf{u}_{id}}(\mathbf{A})$. (Note that this is equivalent to signing the message id, and at most one distinct key should be revealed for each identity.) Following Equation (5.2.4), this $\mathbf{x}_{id}$ can be interpreted as a secret key for the public key $\mathbf{A}_{id}$, because
$$\mathbf{A}_{id} \cdot (-\mathbf{x}_{id}, 1) = \mathbf{0}.$$

- Encrypting $\mu \in \{0, 1\}$ to an identity id and decrypting are exactly as in the dual LWE system (see Equations (5.2.5) and (5.2.6)), using the user-specific public key $\mathbf{A}_{id}$ and secret key $\mathbf{x}_{id}$:
$$\begin{aligned} \mathbf{c}_{id}^t &\approx \mathbf{s}^t \mathbf{A}_{id} + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil)^t \\ \mathbf{c}_{id}^t \cdot (-\mathbf{x}_{id}, 1) &\approx \mu \cdot \lfloor \tfrac{q}{2} \rceil. \end{aligned}$$

Of course, for ℓ-bit messages the system can be amortized in the usual way, by hashing each identity to multiple syndromes $\mathbf{u}_1, \ldots, \mathbf{u}_\ell$.

By modeling the hash function as a random oracle, and under an appropriate LWE assumption, the above system can be proved semantically secure. (Recall that the attacker can obtain secret keys for any identities of its choice, other than its target identity.) The proof proceeds more or less the same as for GPV signatures, and even admits a tight reduction, though there are some additional technicalities related to the use of LWE.

5.5.3 Removing the Random Oracle, and Hierarchical IBE

The signature and IBE schemes described above depend on the random-oracle heuristic for their security analysis. A work by Cash, Hofheinz, Kiltz, and Peikert [CHKP10] gave lattice-based digital signature and IBE schemes that are provably secure in the *standard* model (i.e., without random oracles), along with *hierarchical* IBE (HIBE) schemes. In a HIBE, any user can securely use its secret key to *delegate* a valid secret key to any subordinate user in a hierarchy (i.e., a tree). We note that the standard-model schemes of Cash *et al.* have substantially larger public keys than their random-oracle counterparts, so they cannot be considered practical.

The HIBE of [CHKP10] can be seen as an extended version of the dual LWE cryptosystem from Section 5.2.2, and works as follows: consider a hierarchy representing a complete binary tree of depth d, where each node in the tree naturally corresponds to a string in $\{0,1\}^{\leq d}$. (Such a tree can simulate a hierarchy of larger arity, by considering blocks of several bits for each level of the hierarchy.)

- The master public key consists of a (nearly) uniformly random matrix $\bar{\mathbf{A}}$, generated with a trapdoor; uniformly random matrices $\mathbf{A}_{i,b}$ for $i \in \{1,\ldots,d\}$ and $b \in \{0,1\}$ having the same dimensions as $\bar{\mathbf{A}}$; and a uniformly random syndrome $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret key is the trapdoor for $\bar{\mathbf{A}}$.
- An identity $id \in \{0,1\}^{\leq d}$ of length $\ell \leq d$ corresponds to the matrix

  $$\mathbf{A}_{id} = [\bar{\mathbf{A}} \mid \mathbf{A}_{1,id_1} \mid \cdots \mid \mathbf{A}_{\ell,id_\ell}].$$

  A secret key for identity id consists of a random trapdoor of sufficient quality for $\mathbf{A}_{id}$, and a Gaussian-distributed solution $\mathbf{x}_{id}$ to $\mathbf{A}_{id} \cdot \mathbf{x}_{id} = \mathbf{u}$. Note that such an $\mathbf{x}_{id}$ is merely a secret key for the public key $[\mathbf{A}_{id} \mid \mathbf{u}]$ in the dual LWE cryptosystem, i.e., $[\mathbf{A}_{id} \mid \mathbf{u}] \cdot (-\mathbf{x}_{id}, 1) = \mathbf{0}$.

  A secret key for id can be generated from *any* trapdoor of sufficient quality for $\mathbf{A}_{id'}$ for *any prefix* id' of id, using the trapdoor sampling, extension, and randomization techniques described in Section 5.4.2 or 5.4.3. Therefore, user id' can delegate a secret key to user id. (Recall, though, that the quality of the secret key degrades by some polynomial factor with each delegation.)
- Encryption and decryption to an identity id are exactly as in the dual LWE cryptosystem, using the public key $[\mathbf{A}_{id} \mid \mathbf{u}]$ and secret key $\mathbf{x}_{id}$, i.e., the ciphertext is $\mathbf{c}^t \approx \mathbf{s}^t \cdot [\mathbf{A}_{id} \mid \mathbf{u}] + (\mathbf{0}, \mu \cdot \lfloor \frac{q}{2} \rceil)^t$.

For an appropriate choice of parameters and LWE assumption, the above HIBE system is semantically secure under a *selective identity* attack. In such an attack, the adversary must *first* name the target identity $id^* \in \{0,1\}^{\leq d}$, is then given the master public key and a challenge ciphertext encrypted to id^*, and may then query secret keys for any identity id that is not a prefix of id^*. (With additional ideas, the system can be made secure under an adaptive-identity attack, but it becomes even less efficient; see [CHKP10] for details.)

The main ideas behind the security proof are as follows: we design a reduction (a "simulator") that is given a source of samples $(\mathbf{a}_i, b_i)$, and aims to determine whether they are LWE-distributed or uniform, using an adversary that attacks the HIBE scheme. The simulator first obtains a target identity $id^* \in \{0,1\}^\ell$ from the adversary, then generates a master public key that is "punctured" at id^*: it assembles $\bar{\mathbf{A}}$, $\mathbf{A}_{i,id_i^*}$ for $1 \leq i \leq \ell$, and $\mathbf{u}$ from the $\mathbf{a}_i$ components of sufficiently many input samples, and also creates a corresponding challenge ciphertext from the b_i components. It generates all the remaining matrices $\mathbf{A}_{i,b}$ together with secret trapdoors, and gives all this to the adversary. Now by construction, the reduction can sample properly distributed secret keys for any queried identity *except* for prefixes of id^* (which are anyway disallowed), by extending and randomizing its secret trapdoors. Finally, notice that if the input samples were LWE-distributed, then this whole simulation exactly matches the real attack (up to negligible statistical error), whereas if they were uniform, the challenge ciphertext hides the message information-theoretically. Therefore, the reduction distinguishes LWE from uniform samples with essentially the same advantage as the HIBE adversary.

5.5.4 Efficiency Improvements via Algebraic Puncturing

Shortly following [CHKP10], Agrawal, Boneh, and Boyen [ABB10] constructed standard-model (H)IBE systems in which the master public key contains many fewer parity-check matrices: just one or two per level of an exponential-arity hierarchy, rather than linearly many as in [CHKP10]. The main idea in [ABB10] is a more algebraic, as opposed to combinatorial, method of "puncturing" a public key and its trapdoor, as described in Section 5.4.3. This allows the security reduction to know a trapdoor for all but one of an *exponentially large* number of related parity-check matrices, which is sufficient for security under selective-identity attacks. (A similar technique was used in a different context in [PW08].)

A version of the scheme that uses gadget-based trapdoors (Section 5.4.3) works as follows: consider a hierarchy representing a complete tree of depth d, where identities correspond to tuples over the tag space $\mathcal{H} = \{\mathbf{H}_i\}$ of the underlying puncturable trapdoor functions. (Recall that we can construct, for example, a tag space corresponding to nonzero finite field elements $\mathbb{F}_{q^n} \setminus \{0\}$ when q is prime.)

- The master public key consists of a (nearly) uniformly random matrix $\bar{\mathbf{A}}$, generated with a trapdoor; uniformly random $\mathbf{A}_1, \ldots, \mathbf{A}_d$ having the same dimensions as the gadget matrix $\mathbf{G}$; and a uniformly random syndrome $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret key is the trapdoor for $\bar{\mathbf{A}}$.
- An identity $id \in \mathcal{H}^{\leq d}$ of length $\ell \leq d$ corresponds to the matrix

  $$\mathbf{A}_{id} = [\bar{\mathbf{A}} \mid \mathbf{A}_1 + \mathbf{H}_{id_1}\mathbf{G} \mid \cdots \mid \mathbf{A}_\ell + \mathbf{H}_{id_\ell}\mathbf{G}].$$

  As above, a secret key for identity id is a random trapdoor of sufficient quality for $\mathbf{A}_{id}$, and a Gaussian-distributed solution $\mathbf{x}_{id}$ to $\mathbf{A}_{id} \cdot \mathbf{x}_{id} = \mathbf{u}$. These can be generated from any trapdoor of corresponding quality for $\mathbf{A}_{id'}$ for any prefix id' of id, using the techniques described in Section 5.4.3.
- Encryption and decryption for an identity id are exactly as above, using $[\mathbf{A}_{id} \mid \mathbf{u}]$ and $\mathbf{x}_{id}$.

Under an appropriate LWE assumption, the scheme is semantically secure under a selective-identity attack. In the proof, the basic setup of the simulator is essentially the same as in the one above. Given a target identity $id^* \in \mathcal{H}^\ell$, the simulator constructs $\bar{\mathbf{A}}$ from its input samples, and for $1 \leq i \leq d$ it constructs

$$\mathbf{A}_i := -\mathbf{H}_{id_i^*}\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_i$$

for a short random $\mathbf{R}_i$, where we define $\mathbf{H}_{id_j^*} = \mathbf{0}$ for $j > \ell$. In other words, the reduction knows a trapdoor $\left[\begin{smallmatrix}\mathbf{R}_i \\ \mathbf{I}\end{smallmatrix}\right]$ with tag $-\mathbf{H}_{id_i^*}$ for $[\bar{\mathbf{A}} \mid \mathbf{A}_i]$. The reduction can therefore answer secret-key queries for any identity id that is not a prefix of id^*: such a query must differ from id^* in some position j, therefore the reduction knows a trapdoor for (any extension of) $[\bar{\mathbf{A}} \mid \mathbf{A}_j + \mathbf{H}_{id_j}\mathbf{G}]$. At the same time, because the master public key is punctured at id^*, i.e., $\mathbf{A}_i + \mathbf{H}_{id_i^*}\mathbf{G} = -\bar{\mathbf{A}}\mathbf{R}_i$, the reduction can use the $\mathbf{b}^t$ part of its input and its $\mathbf{R}_i$ to generate a challenge ciphertext for $[\mathbf{A}_{id} \mid \mathbf{u}] = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\mathbf{R}_1 \mid \cdots \mid -\bar{\mathbf{A}}\mathbf{R}_\ell \mid \mathbf{u}]$, which is either properly distributed or which statistically hides the message, depending on whether the input is LWE-distributed or uniform, respectively. (This can be done using the random self-reduction technique of generating fresh LWE-or-uniform samples from the given input samples, as described at the end of Section 5.2.1.)

Signatures. The above puncturing technique yields a rather compact (H)IBE, but by itself it does not work for signatures due to the "all but one" nature of the puncturing. Boyen [Boy10] adapted the technique to signatures, improving upon [CHKP10] in the bit length of the signatures themselves, though still with a public key that contains a linear number of parity-check matrices. Recently, Ducas and Micciancio [DM14] gave a variant that reduced the number of matrices in the public key to only logarithmic. Shortly thereafter, Alperin-Sheriff [Alp15] reduced the number of matrices to only a constant, but at the cost of a substantially larger norm bound for the underlying SIS problem.

5.6 Signatures Without Trapdoors

A separate line of research on lattice-based digital signatures [LM08, Lyu08, Lyu09, Lyu12, DDLL13] has evolved in parallel with the trapdoor and discrete Gaussian-based methods described in Section 5.5.1 above. The first work in this line [LM08] gave a *one-time* signature based on the ring-SIS problem, which can be converted to a many-time scheme (in the standard model) using standard tree-hashing techniques; it is described in further detail in Section 5.6.1. Using the main idea behind the one-time signature as inspiration, the remaining works adhere to the following template: first design an appropriate kind of public-coin, interactive *identification* protocol, then apply the Fiat-Shamir heuristic [FS86] to convert it into a noninteractive signature scheme in the random-oracle model. Recall that the Fiat-Shamir heuristic is a generic transformation that replaces the verifier's random challenge(s) with the output of random oracle on the transcript of the protocol thus far. More details on this approach are given in Section 5.6.2.

5.6.1 One-Time and Tree-Based Signatures

Lyubashevsky and Micciancio [LM08] constructed an asymptotically quasi-optimal one-time signature from ring-SIS.[11,12] Here "quasi-optimal" means that the key sizes, message length, signature length, and runtimes of key generation, signing, and verification are all quasi-linear $\tilde{O}(\lambda)$ in the security parameter λ, for a conjectured λ bits of security under standard assumptions (i.e., the best known attacks require 2^λ time).[13] Because the scheme is quasi-optimal, and in particular because messages are $\tilde{\Omega}(\lambda)$ bits, the one-time signature can be transformed into a many-time signature (in the standard model) via standard tree-hashing techniques, with only polylogarithmic overhead. However, the resulting many-time scheme requires a *stateful* signer, and the concrete polylogarithmic factors seem to make the scheme unrealistic in practice.

For simplicity, we describe the main idea behind the original one-time signature based on R-SIS for a degree-n ring R over $\mathbb{Z}$. The public parameter, which could be trusted randomness shared by all users, is a parity-check vector $\vec{a} \in R_q^\ell$, for $\ell \approx \log q$. The secret key is two short vectors of ring elements $\vec{x}, \vec{y} \in R^\ell$, and the public key is

$$u = f_{\vec{a}}(\vec{x}) = \vec{a}^t \cdot \vec{x}, \quad v = f_{\vec{a}}(\vec{y}) = \vec{a}^t \cdot \vec{y} \in R_q.$$

To sign a message, it is interpreted as a short ring element $c \in R$, and the signature is $\vec{s} = c \cdot \vec{x} + \vec{y} \in R^\ell$. To verify, one checks that $\vec{a}^t \cdot \vec{s} = c \cdot u + v$, and that $\vec{s}$ is sufficiently short. Note that the verification equation is satisfied for a properly generated signature, because

$$\vec{a}^t \cdot (c \cdot \vec{x} + \vec{y}) = c \cdot (\vec{a}^t \cdot \vec{x}) + (\vec{a}^t \cdot \vec{y}) = c \cdot u + v.$$

The main idea behind one-time security is as follows: the simulator is given $\vec{a} \in R_q^\ell$ as a ring-SIS challenge and wants to find a short solution $\vec{z} \in R^\ell$ to $\vec{a}^t \cdot \vec{z} = 0 \in R_q$. The simulator chooses its own secret key $\vec{x}, \vec{y}$ and gives the corresponding public key $u, v \in R_q$ to the attacker. Because the simulator knows the secret key, it can sign on any (short) $c \in R$ of the attacker's choice, producing the signature $\vec{s} = c \cdot \vec{x} + \vec{y}$. Suppose the attacker then produces a forgery $\vec{t} \in R^\ell$ for a different (short) $c' \neq c$. Then we have

$$\vec{a}^t \cdot (\underbrace{\vec{t} - (c' \cdot \vec{x} + \vec{y})}_{\vec{z}}) = 0 \in R_q,$$

[11] Recall that a one-time signature scheme remains unforgeable if the attacker is given at most one signature on a message of its choice, but may not offer any security if the adversary obtains more than one signature under the same key.

[12] The full version of [LM08] also includes constructions from standard SIS and a coding problem, but these are not quasi-optimal.

[13] We mention that using the quasi-linear trapdoor generation and discrete Gaussian sampling algorithms of [Pei10, MP12], the ring-based variant of the GPV signature scheme (in the random-oracle model) is also quasi-optimal.

so $\vec{z} \in R^\ell$ is a short solution to the ring-SIS challenge, provided that it is nonzero. It can be shown that this is the case with noticeable probability, because some significant information about $\vec{x}, \vec{y}$ is hidden by the one signature $\vec{s} = c \cdot \vec{x} + \vec{y}$. (This would be obviously true if $\vec{x}, \vec{y}$ were *uniform* over R_q, but instead they are short, so the argument is more subtle.) It follows that the forger cannot reliably guess the value of $c' \cdot \vec{x} + \vec{y}$ for *any* short $c' \neq c$, and so $\vec{z}$ is nonzero with noticeable probability.

5.6.2 Identification Protocols and Fiat-Shamir Signatures

Schnorr-like identification protocol. Lyubashevsky [Lyu08] gave the first three-message identification scheme based on (ring-)SIS, which has security against active impersonation attacks. The protocol is inspired by Schnorr's protocol for proving knowledge of a discrete logarithm [Sch89], but involves more technical subtleties due to the use of *short* secrets.

The prover has a secret "very short" uniformly random integer vector $\mathbf{x} \in \{0,1\}^m$, and its public key is $\mathbf{u} = f_{\mathbf{A}}(\mathbf{x}) = \mathbf{A}\mathbf{x} \in \mathbb{Z}_q^n$, where the uniformly random matrix $\mathbf{A} \in \mathbb{Z}_q^{n \times m}$ may be a public parameter generated by a trusted party. One run of the protocol proceeds as follows:

1. The prover first chooses a "somewhat short" $\mathbf{y} \in \{0, \ldots, 5m-1\}^m$ uniformly at random, and sends $\mathbf{v} = \mathbf{A}\mathbf{y}$ to the verifier.
2. The verifier chooses a uniformly random challenge bit $c \leftarrow \{0,1\}$ and sends it to the prover.
3. The prover responds as follows: if $\mathbf{z} = c \cdot \mathbf{x} + \mathbf{y} \in \mathbb{Z}^m$ is "safe," i.e., if $\mathbf{z} \in \{1, \ldots, 5m-1\}^m$, then the prover sends $\mathbf{z}$ to the verifier. Otherwise, it aborts, and the verifier rejects the interaction.
4. If the prover does send some $\mathbf{z}$, the verifier accepts if $\mathbf{A}\mathbf{z} = c \cdot \mathbf{u} + \mathbf{v}$ and $\mathbf{z} \in \{1, \ldots, 5m-1\}^m$, otherwise it rejects. Note that the legitimate prover does convince the verifier, because $\mathbf{A}(c \cdot \mathbf{x} + \mathbf{y}) = c \cdot (\mathbf{A}\mathbf{x}) + (\mathbf{A}\mathbf{y}) = c \cdot \mathbf{u} + \mathbf{v}$, as required.)

In the full protocol, many instances of the protocol are run in parallel, and the verifier accepts the whole interaction only if a sufficiently large fraction of the sub-protocols are accepting.

Notice that if the prover did not check for the safety of $\mathbf{z}$, then a passive eavesdropper could learn the secret key after viewing enough interactions: if an entry of $\mathbf{z}$ is zero (respectively, $5m$), then the corresponding entry of the secret key $\mathbf{x}$ is zero (resp., one). The size of the "somewhat short" domain is taken to be large enough so that $\mathbf{z}$ is safe with some constant probability, so the prover does not abort too often. Using the safety check, Lyubashevsky proves that the protocol is (statistically) *witness indistinguishable* (WI): no matter which secret key $\mathbf{x}$ the prover has for its public key $\mathbf{u}$, the distribution of an interaction with the prover is the same, up to very small statistical error. Since WI is closed under parallel repetition, the whole protocol also remains WI. This is the key fact behind the proof of security under the SIS assumption: a simulator can choose its own secret key and act as the prover, revealing nothing about which of the many valid secret keys it is using. Moreover, if an adversary successfully impersonates the prover, by rewinding it we can extract a valid secret key from it. By the WI property, with good probability the extracted secret key must be different from the simulator's secret key, and their difference is an SIS solution for $\mathbf{A}$.

Efficiency in the ring setting. The full protocol described above is very inefficient in communication and runtime, because many parallel executions are needed to achieve completeness and security. A follow-up work by Lyubashevsky [Lyu09] significantly improved these efficiency measures in the ring setting, using the ring-SIS function $f_{\vec{a}} \colon R^m \to R_q$ defined as $f_{\vec{a}}(\vec{x}) = \vec{a}^t \cdot \vec{x}$ (Equation (4.3.1)). The main idea is to expand the domain of the challenge c from $\{0,1\}$ to all sufficiently short elements in R, of which there

are exponentially many. (Arguably, this version of the protocol is a closer analogue to Schnorr's original identification protocol.) Because $f_{\vec{a}}$ is R-linear, it is easy to check that all the verification equations work out as required, and an analogous security proof can be obtained for appropriate bounds on the various short objects and an appropriate safety test. Using several additional optimizations, the signature scheme obtained from applying the Fiat-Shamir heuristic to Lyubashevsky's protocol comes closer to being practical: for example, signatures are about 60 kilobits, for estimated security levels of about 80 bits.

Back to SIS, and further refinements. A subsequent work by Lyubashevsky [Lyu12] showed that to obtain an identification protocol with good communication complexity, SIS itself is actually sufficient (though ring-SIS is still important for obtaining practical key sizes and runtimes). This work also included various refinements that reduce the norms of the short objects by small polynomial factors (roughly $\sqrt{m}$), which yields better concrete parameters, and milder assumptions via tighter worst-case approximation factors.

The basic idea is that the prover's secret key is now a short integer matrix $\mathbf{X} \in \mathbb{Z}^{m \times \ell}$, and its public key is $\mathbf{U} = \mathbf{AX}$. A run of the protocol proceeds as follows:

1. The prover chooses a "somewhat short" integer vector $\mathbf{y} \in \mathbb{Z}^m$ from a certain distribution, and sends $\mathbf{v} = \mathbf{Ay}$ to the verifier.
2. The verifier chooses a short challenge vector $\mathbf{c} \in \mathbb{Z}^\ell$, and the prover computes $\mathbf{z} = \mathbf{Xc} + \mathbf{y} \in \mathbb{Z}^m$.
3. The prover then applies an appropriate rejection rule to $\mathbf{z}$, and either sends it to the verifier or aborts.
4. If the prover does send some $\mathbf{z}$, the verifier checks that $\mathbf{Az} = \mathbf{Uc} + \mathbf{v}$ and that $\mathbf{z}$ is sufficiently short. Note that the legitimate prover does convince the verifier, because $\mathbf{Az} = \mathbf{A}(\mathbf{Xc} + \mathbf{y}) = \mathbf{Uc} + \mathbf{v}$.

For reasonable parameters, only a small constant numbers of parallel runs of the protocol are necessary to establish adequate completeness and security.

In contrast with the prior works, which used an all-or-nothing "safety check" on $\mathbf{z}$, here the rejection rule is a more refined *probabilistic* test, based on rejection sampling. Essentially, the goal is to ensure that the distribution of $\mathbf{z} = \mathbf{Xc} + \mathbf{y}$ *conditioned on not rejecting* is independent of $\mathbf{X}$. This is done by using rejection sampling to "recenter" the distribution of $\mathbf{z}$ to be a discrete Gaussian centered at zero, rather than at $\mathbf{Xc}$. To ensure that $\mathbf{z}$ is not rejected too often, we let $\mathbf{y}$ have a discrete Gaussian distribution with a parameter proportional to (an upper bound on) $\|\mathbf{Xc}\|$. This ensures that the discrete Gaussians centered at zero and at $\mathbf{Xc}$ have sufficient overlap. A further refinement of this idea uses "bimodal" Gaussians [DDLL13], essentially by letting the prover randomly choose between $\pm\mathbf{Xc}$. This yields slightly more overlap, and allows the Gaussian parameter of $\mathbf{y}$ to be reduced by roughly a $\sqrt{\log(1/\varepsilon)}$ factor, for statistical distance ε.

Practical implementations. Using several additional optimizations and engineering tricks, a programmable hardware (FPGA) implementation of the scheme with unimodal Gaussians was presented in [GLP12]. For an estimated security level of about 80 bits, its public and secret keys are respectively about 12 and 2 kilobits, and signatures are about 9 kilobits. A software implementation of the ring-based bimodal scheme, called BLISS, was also given in [DDLL13]. For estimated security levels of 128 bits, its public and secret keys are respectively about 7 and 2 kilobits, and signatures are about 5.6 kilobits. Signing and verifications times are competitive with (or even better than) those of RSA-2048.

5.7 Pseudorandom Functions

Pseudorandom functions (PRFs), introduced by Goldreich, Goldwasser, and Micali (GGM) [GGM84], are a cornerstone of symmetric-key cryptography. A PRF family is a set $\mathcal{F} = \{F_s \colon D \to R\}$ of functions indexed by an index s, usually called the *secret key* or *seed*, mapping a common finite domain to a common finite range. The pseudorandomness property is that a function $F_s \leftarrow \mathcal{F}$ chosen at random from the family (under a specified distribution) cannot be efficiently distinguished from a uniformly random function $U \colon D \to R$, given oracle access. That is, for some fixed key s chosen at the start of the experiment, the adversary may adaptively query distinct values $x_i \in D$ to receive answers $y_i = F_s(x_i)$, and the values y_i should appear uniformly random and independent (even though they are actually computed determinisically from the seed s and inputs x_i). An example candidate PRF family is AES-128, where $s \in \{0,1\}^{128}$ and $D = R = \{0,1\}^{128}$ as well.[14]

Previous PRF constructions. GGM gave the first formal definition of PRFs and the first provably secure construction, based on any length-doubling pseudorandom generator (PRG) $G \colon \{0,1\}^n \to \{0,1\}^{2n}$. (Such a generator can be constructed from any generator that expands by just a single bit.) Let $G_0, G_1 \colon \{0,1\}^n \to \{0,1\}^n$ respectively denote G with its output restricted to the first and last halves. Then for $s \in \{0,1\}^n$ and any $\ell = \mathrm{poly}(n)$, the GGM PRF $F_s \colon \{0,1\}^\ell \to \{0,1\}^n$ is defined as

$$F_s(x) = G_{x_\ell}(\cdots G_{x_2}(G_{x_1}(s))\cdots),$$

where x_i denotes the ith bit of x. In words: starting with the key s we iteratively apply G, taking either the first or last half of its output as indicated by the bits of the PRF input x. Note that this function is highly *sequential*: its circuit depth is proportional to the PRF input length ℓ.

Naor and Reingold (and Rosen) [NR95, NR97, NRR00] gave highly *parallel* constructions of PRFs based on a generic object they called a *synthesizer*, and even lower-depth PRFs from specific number-theoretic assumptions like the hardness of the decisional Diffie-Hellman (DDH) and factoring problems. The latter constructions are defined as "exponentials of subset-products:" for input domain $D = \{0,1\}^\ell$, the key consists of ℓ elements $s_i \in \mathbb{Z}_p$ and a generator g of some public multiplicative group G of order p. An input $x \in \{0,1\}^\ell$ specifies the subset-product $s_x = \prod_i s_i^{x_i}$, and the output is $g^{s_x} \in G$.

5.7.1 Learning With Rounding and Lattice-Based PRFs

The pseudorandomness inherent in the decision-LWE problem immediately yields a PRG, but it is rather complex and inefficient in its use of input randomness, because it needs to use the input bits to sample errors from a Gaussian-like distribution. And as for PRFs, the GGM construction negates the good parallelism inherent to LWE. This motivates the search for alternative constructions of lattice-based PRGs and PRFs.

Learning With Rounding. Banerjee, Peikert, and Rosen (hereafter BPR) [BPR12] constructed more efficient PRGs and the first non-generic PRFs from lattice problems, namely, LWE. Their first contribution is a problem called *learning with rounding* (LWR), which is essentially a "derandomized" version of LWE in which the errors are deterministic. As with LWE, in LWR there is a secret $\mathbf{s} \in \mathbb{Z}_q^n$, and the goal is to

[14] In fact, each function in the AES family is actually a *bijection* on $\{0,1\}^{128}$, and knowing s allows one to efficiently compute F_s^{-1} as well. In general, a PRF need not have these properties.

distinguish "noisy" random inner products with $\mathbf{s}$ from uniformly random. The difference is that in LWR, the samples are generated as *rounded* inner products

$$(\mathbf{a}_i \ , \ b_i = \lfloor \langle \mathbf{s}, \mathbf{a}_i \rangle \rceil_p) \in \mathbb{Z}_q^n \times \mathbb{Z}_p,$$

where $\mathbf{a}_i \in \mathbb{Z}_q^n$ is uniformly random, and $\lfloor \cdot \rceil_p \colon \mathbb{Z}_q \to \mathbb{Z}_p$ for $p < q$ is the modular "rounding function," defined as $\lfloor x + q\mathbb{Z} \rceil_p := \lfloor x \cdot (p/q) \rceil + p\mathbb{Z}$. Intuitively, in LWE the less-significant information about the inner product is concealed by random small error, whereas in LWR the same effect is achieved by deterministic rounding.

BPR prove that for super-polynomial $q/p \geq (\alpha q) \cdot n^{\omega(1)}$, LWR is at least as hard as LWE with modulus q and error rate $\alpha \leq n^{-\omega(1)}$. Interestingly, it seems likely that LWR is hard for much smaller ratios q/p (where p divides q to avoid rounding bias), though no proof based on worst-case lattice problems is yet known. We note that follow-up work by Alwen *et al.* [AKPW13] and Bogdanov *et al.* [BGM+16] obtained partial results in this direction for a *bounded* number m of LWR samples, depending on the ratio q/p and the underlying LWE error rate. These results suffice for the security of certain applications, but unfortunately not for PRFs.

Pseudorandom functions. BPR use the LWR problem to construct very simple and randomness-efficient PRGs, which yield more practical GGM-like PRFs, as well as very simple log-depth *synthesizers*, which yield polylogarithmic-depth PRFs following the paradigm of Naor and Reingold [NR95]. Finally, BPR also give an analogue of the lowest-depth PRF constructions from [NR97, NRR00]. Instead of an exponential of a subset-product, the construction is a *rounded* subset-product, where the product is over a collection of secret matrices. More specifically, for domain $\{0,1\}^\ell$ the functions are defined as

$$F_{\mathbf{a},\{\mathbf{S}_i\}}(x) = \left\lfloor \mathbf{a}^t \cdot \prod_{i=1}^{\ell} \mathbf{S}_i^{x_i} \right\rceil_p, \tag{5.7.1}$$

where the secret key consists of a uniformly random $\mathbf{a} \in \mathbb{Z}_q^n$ and ℓ short Gaussian-distributed matrices $\mathbf{S}_i \in \mathbb{Z}^{n \times n}$. BPR prove that this function is a secure PRF under the LWE assumption, but for a very small error rate $\alpha = n^{-\Omega(\ell)}$. This induces a rather large modulus $q = n^{\Omega(\ell)}$ and dimension n, and correspondingly large key sizes.

Interestingly, and similarly to the LWR problem, the function in Equation (5.7.1) might be a secure PRF even for much smaller parameters, e.g., a sufficiently large $q = \mathrm{poly}(n)$ that is divisible by p. While no security proof is known for these parameters, neither is any effective attack, beyond generic attacks on LWE. A practical instantiation and software implementation of the above construction's ring-based analogue was given in [BBL+14].

5.7.2 Key-Homomorphic PRFs

Following [BPR12], Boneh *et al.* (hereafter BLMR) [BLMR13] gave the first standard-model constructions of *key homomorphic* PRFs (KH-PRFs), using lattices/LWE. (Previously, the only constructions of KH-PRFs were in the random-oracle model [NPR99].) A KH-PRF family is a PRF family $\{F_k\}$ with the additional property that $F_{k_1+k_2}(x) = F_{k_1}(x) + F_{k_2}(x)$ for all keys k_1, k_2 and inputs x, where both the key space and output range are interpreted as finite additive groups. As shown in [NPR99, BLMR13], KH-PRFs have many useful applications, such as distributing the operation of a key-distribution center, and updatable symmetric-key encryption.

In the BLMR construction, the secret key is just a uniformly random vector $\mathbf{r} \in \mathbb{Z}_q^m$ for $m \approx n \log q$, and there are two short (e.g., binary or Gaussian) random square matrices $\mathbf{B}_0, \mathbf{B}_1 \in \mathbb{Z}^{m \times m}$, which are treated as shared randomness common to all keys. Alternatively, we may see these matrices as $\mathbf{B}_b = \mathbf{G}^{-1}(\mathbf{A}_b)$ for some public uniformly random $\mathbf{A}_b \in \mathbb{Z}_q^{n \times m}$, where the decomposition function $\mathbf{G}^{-1}(\cdot)$ is as defined in Equation (5.4.1). Like the BPR construction, for input domain $\{0,1\}^\ell$ the BLMR function is defined as the rounded subset-product

$$F_{\mathbf{r}}(x) = \left\lfloor \mathbf{r}^t \cdot \prod_{i=1}^{\ell} \mathbf{B}_{x_i} \right\rceil_p .$$

It is easy to verify that this construction is "almost" key-homomorphic, in that $F_{\mathbf{r}_1}(x) + F_{\mathbf{r}_2}(x)$ and $F_{\mathbf{r}_1+\mathbf{r}_2}(x)$ differ by at most one in each coordinate, due to the different order of rounding in the two cases.

BLMR prove that their construction is a secure PRF under the n-dimensional LWE assumption, for error rates $\alpha = n^{-\Omega(\ell)}$, and hence for parameters that are comparable to those of the lowest-depth BPR construction (Equation (5.7.1)). The main idea used in the proof is that LWE with a "large" secret $\mathbf{r} \in \mathbb{Z}_q^m$ and short public $\mathbf{B} = \mathbf{G}^{-1}(\mathbf{A}) \in \mathbb{Z}^{m \times m}$ is as hard as LWE in dimension $n \approx m/\log q$. This is because

$$\mathbf{s}^t \cdot \mathbf{A} = \underbrace{(\mathbf{s}^t \mathbf{G})}_{\mathbf{r}^t} \cdot \mathbf{G}^{-1}(\mathbf{A}),$$

so we can efficiently transform regular LWE samples with secret $\mathbf{s}$ into those of the form described above. (The same idea has been used in many other works and contexts, such as [BV11b, MP12, BLP+13, GSW13].)

Following [BLMR13], Banerjee and Peikert [BP14] gave key-homomorphic PRFs from substantially weaker LWE assumptions, e.g., error rates of only $\alpha = n^{-\Omega(\log \ell)}$ or even $\alpha = n^{-\omega(1)}$, which yields better key sizes and runtimes. For example, the key sizes were reduced from $\tilde{O}(\lambda^3)$ to $\tilde{O}(\lambda)$ bits, and the shared randomness was reduced from $\tilde{O}(\lambda^6)$ bits to $\tilde{O}(\lambda^2)$ bits, with comparable improvements for ring-based constructions. These improvements come at the expense of slightly worse parallelism, specifically, a logarithmic factor in the depth of the "publicly computable" subset-product part of the function. The main idea in the construction from [BP14] is that instead of letting the PRF input x define a subset-product of the short matrices $\mathbf{B}_b = \mathbf{G}^{-1}(\mathbf{A}_b)$, we let it define a matrix $\mathbf{A}_x \in \mathbb{Z}_q^{n \times m}$ via a predefined scheduling of matrix multiplications and $\mathbf{G}^{-1}(\cdot)$ decompositions. In particular, a product of decomposed matrices may itself be decomposed, in a nested fashion. This has the effect of better controlling the expansion of the error terms in the security proof, allowing for the use of smaller parameters. These ideas inherit from recent literature on fully homomorphic and attribute-based encryption [BV14, BGG+14, AP14], as described next in Chapter 6.

Finally, we mention that recent independent works of Banerjee *et al.* [BFP+15] and Brakerski and Vaikuntanathan [BV15] generalized the construction of [BP14] in different ways to give key-homomorphic *constrained* PRFs. Constrained PRFs, introduced in the concurrent and independent works [KPTZ13, BW13, BGI14], allow for delegation of secret keys that allow the PRF to be evaluated on inputs satisfying certain predicates, while at the same time preserving the pseudorandomness of the function outputs on all other inputs.

Chapter 6

Advanced Constructions

In this chapter we survey a selection of very powerful cryptographic objects, namely, *fully homomorphic encryption* and *attribute-based encryption* for arbitrary circuits. To date, the only known constructions of such objects are based on lattice problems of various kinds. Here we mainly restrict our attention to constructions based on the LWE problem.

6.1 Fully Homomorphic Encryption

In 1978, Rivest, Adleman, and Dertouzos [RAD78] proposed a concept which has come to be known as *fully homomorphic encryption*, or FHE. (At the time they called it a "privacy homomorphism.") In brief, an FHE scheme allows *computation on encrypted data*, or more concisely, *homomorphic computation*: given a ciphertext that encrypts some data μ, one can compute a ciphertext that encrypts $f(\mu)$ for any desired (efficiently computable) function f. We emphasize that this is possible without ever needing to decrypt the data or know the decryption key.

Fully homomorphic encryption was known to have abundant applications in cryptography, but for three decades no plausibly secure scheme was known. This changed in 2009, when Gentry proposed a candidate FHE scheme based on ideal lattices [Gen09b, Gen09a]. Gentry's seminal work generated tremendous excitement, and was quickly followed by many works (e.g., [vDGHV10, Gen10b, SV11, BV11a, CMNT11, BV11b, BGV12, CNT12, GHS12b, Bra12, GHPS12, CCK$^+$13, GSW13], among others) that offered various improvements in conceptual and technical simplicity, efficiency, security guarantees, etc. In this section we give an overview of the main ideas behind recent LWE-based FHE schemes, building on the tools described in the previous sections. For additional details, see the earlier surveys by Gentry [Gen10a] and Vaikuntanathan [Vai11].

6.1.1 FHE from LWE

The earliest "first generation" FHE constructions [Gen09b, vDGHV10] were based on ad-hoc average-case assumptions about ideal lattices and the "approximate GCD" problem, respectively. In a sequence of works, Brakerski and Vaikuntanathan (hereafter BV) [BV11a, BV11b] gave a "second generation" of FHE constructions, which were based on standard assumptions supported by worst-case hardness, namely, (ring-)LWE. Here we describe the main ideas behind the LWE-based scheme from [BV11b], with additional improvements from a subsequent work with Gentry [BGV12].

The BV scheme encrypts a single bit per ciphertext, and supports homomorphic *addition* and *multiplication* modulo two.[1] With some important caveats (as explained below), this is sufficient for FHE because by composing such operations, one can homomorphically evaluate any boolean circuit. In the BV system, a secret key is an LWE secret, and an encryption of a bit is simply an LWE sample for an odd modulus q, where the error term e encodes the message as its least-significant bit. More precisely, a ciphertext that encrypts $\mu \in \mathbb{Z}_2$ under a secret key $\mathbf{s} \in \mathbb{Z}^n$ is a vector $\mathbf{c} \in \mathbb{Z}_q^n$ such that

$$\langle \mathbf{s}, \mathbf{c} \rangle = \mathbf{s}^t \cdot \mathbf{c} = e \pmod{q}, \tag{6.1.1}$$

where $e \in \mathbb{Z}$ is some small error such that $e = \mu \pmod 2$, i.e., $e \in \mu + 2\mathbb{Z}$. To decrypt $\mathbf{c}$ using $\mathbf{s}$, one just computes $\langle \mathbf{s}, \mathbf{c} \rangle \in \mathbb{Z}_q$, lifts the result to its unique representative $e \in \mathbb{Z} \cap [-\frac{q}{2}, \frac{q}{2})$, and outputs $\mu = e \bmod 2$.

Notice that by taking $\mathbf{s} = (-\bar{\mathbf{s}}, 1)$, a ciphertext vector $\mathbf{c} = (\bar{\mathbf{c}}, c)$ is just an LWE sample with an $(n-1)$-dimensional secret $\bar{\mathbf{s}}$ and error term e, because $c = \langle \bar{\mathbf{s}}, \bar{\mathbf{c}} \rangle + e \pmod q$. In the symmetric-key setting, such a ciphertext can be produced directly using $\bar{\mathbf{s}}$; in the asymmetric-key setting we can just add a random combination of LWE samples with respect to $\bar{\mathbf{s}}$, which are given in the public key. (Note that this is essentially how all LWE public-key encryption schemes work; see Section 5.2.) Using these observations it is straightforward to show that ciphertexts are pseudorandom, hence the encryption is passively secure, assuming the hardness of decision-LWE.

We remark that the decryption relation expressed in Equation (6.1.1), where $e \in \mu + 2\mathbb{Z}$, is sometimes called the "least significant bit" encoding of the message, as opposed to the "most significant bit" encoding used throughout Chapter 5, where $\langle \mathbf{s}, \mathbf{c} \rangle \approx \mu \cdot \lfloor \frac{q}{2} \rceil \pmod q$. It turns out that the two encodings are equivalent when the plaintext and ciphertext moduli are coprime, because one can easily and losslessly switch between them without knowing the secret key; see [AP13, Appendix A] for a simple proof.

Homomorphic operations. We now describe homomorphic addition and multiplication. For $i = 1, 2$, let $\mathbf{c}_i \in \mathbb{Z}_q^n$ be a ciphertext that encrypts $\mu_i \in \mathbb{Z}_2$ under secret key $\mathbf{s}$, with small error term $e_i \in \mu_i + 2\mathbb{Z}$. Homomorphic addition is simple: $\mathbf{c}_1 + \mathbf{c}_2 \in \mathbb{Z}_q^n$ encrypts $\mu_1 + \mu_2 \in \mathbb{Z}_2$, because

$$\langle \mathbf{s}, \mathbf{c}_1 + \mathbf{c}_2 \rangle = \langle \mathbf{s}, \mathbf{c}_1 \rangle + \langle \mathbf{s}, \mathbf{c}_2 \rangle = e_1 + e_2 \pmod q,$$

and of course $e_1 + e_2 \in (\mu_1 + \mu_2) + 2\mathbb{Z}$ is still small. Notice, however, that we cannot add an unbounded number of ciphertexts, because eventually the magnitude of the error will grow larger than $q/2$, in which case decryption may fail; we return to this issue shortly.

Homomorphic multiplication is a bit trickier. We start with the observation that the *tensor* (or Kronecker) product $\mathbf{c}_1 \otimes \mathbf{c}_2 = (c_{1,i} \cdot c_{2,j})_{i,j} \in \mathbb{Z}_q^{n^2}$ is a valid encryption of $\mu_1 \mu_2 \in \mathbb{Z}_2$ under an *alternative secret key* $\mathbf{s} \otimes \mathbf{s} \in \mathbb{Z}_q^{n^2}$, i.e., the secret key tensored with itself. This is because by the mixed-product property of tensor products,

$$\langle \mathbf{s} \otimes \mathbf{s}, \mathbf{c}_1 \otimes \mathbf{c}_2 \rangle = \langle \mathbf{s}, \mathbf{c}_1 \rangle \cdot \langle \mathbf{s}, \mathbf{c}_2 \rangle = e_1 \cdot e_2 \pmod q,$$

and $e_1 \cdot e_2 \in \mu_1 \mu_2 + 2\mathbb{Z}$ is still rather small, as long as the original errors were small enough to begin with. So just as with homomorphic addition, the number of homomorphic multiplications is bounded a priori.

Key switching. Homomorphic multiplication as described above has an even more significant problem than the error growth: the *dimension* of the ciphertext also grows extremely fast, i.e., exponentially with the

[1] The scheme is easily generalizable to a message space of $\mathbb{Z}_p$ for any small integer p.

number of multiplied ciphertexts, due to the use of the tensor product. To resolve this issue, BV introduced a clever *dimension reduction*—also called *key switching*—technique. Suppose we have an n_{in}-dimensional ciphertext $\mathbf{c}_{\text{in}}$ (e.g., $\mathbf{c}_{\text{in}} = \mathbf{c}_1 \otimes \mathbf{c}_2$ as above) that encrypts some message μ under a secret key $\mathbf{s}_{\text{in}}$ (e.g., $\mathbf{s}_{\text{in}} = \mathbf{s} \otimes \mathbf{s}$ as above), under the "most-significant bit" encoding:

$$\langle \mathbf{s}_{\text{in}}, \mathbf{c}_{\text{in}} \rangle = \mathbf{s}_{\text{in}}^t \cdot \mathbf{c}_{\text{in}} \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil \pmod q.$$

We wish to convert $\mathbf{c}_{\text{in}}$ to an n_{out}-dimensional ciphertext $\mathbf{c}_{\text{out}}$ that still encrypts μ, but with respect to some possibly different secret key $\mathbf{s}_{\text{out}}$. The first main insight is that

$$\langle \mathbf{s}_{\text{in}}, \mathbf{c}_{\text{in}} \rangle = \mathbf{s}_{\text{in}}^t \cdot \mathbf{c}_{\text{in}} = (\mathbf{s}_{\text{in}}^t \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{c}_{\text{in}}) \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil \pmod q, \tag{6.1.2}$$

where $\mathbf{G}$ is the gadget matrix with n_{in} rows as defined in Section 5.4.3; also recall that $\mathbf{G}^{-1}(\mathbf{c}_{\text{in}})$ is a *short* integer vector. Key-switching is made possible by publishing a suitable "encryption" of $\mathbf{s}_{\text{in}}$ under $\mathbf{s}_{\text{out}}$, namely, a matrix $\mathbf{K}$ over $\mathbb{Z}_q$ such that

$$\mathbf{s}_{\text{out}}^t \mathbf{K} \approx \mathbf{s}_{\text{in}}^t \mathbf{G} \pmod q, \tag{6.1.3}$$

where the approximation hides small errors. Essentially, the columns of $\mathbf{K}$ are LWE samples with respect to $\mathbf{s}_{\text{out}}$, with $\mathbf{s}_{\text{in}}^t \mathbf{G}$ added to the last row. Assuming the hardness of LWE, it is easy to prove that such a $\mathbf{K}$ is pseudorandom and hence safe to publish, as long as $\mathbf{s}_{\text{in}}$ and $\mathbf{s}_{\text{out}}$ are independent.[2]

To key-switch the ciphertext $\mathbf{c}_{\text{in}}$ using $\mathbf{K}$, we simply output $\mathbf{c}_{\text{out}} = \mathbf{K} \cdot \mathbf{G}^{-1}(\mathbf{c}_{\text{in}})$. Combining this with Equations (6.1.2) and (6.1.3), we see that

$$\mathbf{s}_{\text{out}}^t \cdot \mathbf{c}_{\text{out}} = (\mathbf{s}_{\text{out}}^t \mathbf{K}) \cdot \mathbf{G}^{-1}(\mathbf{c}_{\text{in}}) \approx (\mathbf{s}_{\text{in}}^t \mathbf{G}) \cdot \mathbf{G}^{-1}(\mathbf{c}_{\text{in}}) \approx \mu \cdot \lfloor \tfrac{q}{2} \rceil \pmod q,$$

where the first approximation holds by Equation (6.1.3) and because $\mathbf{G}^{-1}(\mathbf{c}_{\text{in}})$ is a *short* integer vector. Therefore, $\mathbf{c}_{\text{out}}$ encrypts μ under $\mathbf{s}_{\text{out}}$, as desired.

Error management and modulus switching. Recall from above that homomorphic addition and multiplication increase the magnitude of the error terms in the resulting ciphertexts; in particular, the error in a homomorphic product of ciphertexts is the product of the individual errors (plus a little more, due to key-switching). This severely limits the homomorphic capacity and the hardness/efficiency tradeoff of the scheme: the modulus q must be larger than the error in the final ciphertext, so freshly encrypted ciphertexts must have very small error rate relative to q. More specifically, the scheme as described so far can homomorphically evaluate circuits of only a fixed *logarithmic* depth $d = O(\log \lambda)$ in the security parameter λ, because the modulus must be $q = \lambda^{\Omega(2^d)}$.

A very simple but powerful *modulus reduction* technique, first used in [BV11b] and then exploited to its full potential in [BGV12], greatly extends the homomorphic capacity to circuits of any a-priori bounded *polynomial* depth $d = \text{poly}(\lambda)$ in the security parameter. The idea is that by strategically *scaling down* the modulus by some $\text{poly}(n)$ factor (typically, before homomorphic multiplication), we can decrease the *absolute* error $|e|$ to some small fixed polynomial, even though the relative error *rate* $|e|/q$ remains essentially unchanged. Because the absolute error is what determines the error growth in homomorphic multiplication, modulus reduction yields an arbitrage that allows us to evaluate a depth-d circuit with an (original) modulus of only $q = \lambda^{\Omega(d)}$. More specifically, after evaluating d layers of a circuit, our original modulus q shrinks to

[2] Note that if we want to switch from key $\mathbf{s}_{\text{in}} = \mathbf{s} \otimes \mathbf{s}$ back to the *original* $\mathbf{s}_{\text{out}} = \mathbf{s}$, then the keys are not independent. In such a case we can simply make the "circular security" assumption that publishing $\mathbf{K}$ is safe, though this assumption is still not very well understood.

$q/n^{O(d)}$ while the absolute error remains $\mathrm{poly}(n)$. So it suffices to set the original modulus as $q = n^{\Theta(d)}$ in order to ensure correct decryption after a depth-d computation.

The modulus reduction technique relies on the normal form of the LWE problem, where the secret $\mathbf{s} \in \mathbb{Z}^n$ is a rather short integer vector drawn from the error distribution (see Section 4.2.1). The main idea is that *rounding* an LWE sample (having a short secret) from $\mathbb{Z}_q$ to $\mathbb{Z}_{q'}$ scales the absolute error by a q'/q factor, plus a small additive term:

$$\langle \mathbf{s}, \mathbf{c} \rangle \in e + q\mathbb{Z} \quad \Longrightarrow \quad \langle \mathbf{s}, \lfloor \mathbf{c} \rceil_{q'} \rangle \in \left\langle \mathbf{s}, \tfrac{q'}{q} \cdot \mathbf{c} + [-\tfrac{1}{2}, \tfrac{1}{2}]^n \right\rangle \subseteq (\tfrac{q'}{q} \cdot e + \|\mathbf{s}\|\sqrt{n} \cdot [-\tfrac{1}{2}, \tfrac{1}{2}]) + q'\mathbb{Z}.$$

In the FHE context, one can verify that the above rounding also preserves the encrypted message, when the ciphertext is in most-significant bit form (i.e., $\langle \mathbf{s}, \mathbf{c} \rangle \approx \mu \cdot \lfloor \frac{q}{2} \rceil \pmod q$).

We also mention that Brakerski [Bra12] gave an alternative "scale invariant" method of homomorphic multiplication that increases the error rate by only a fixed $\mathrm{poly}(n)$ factor, regardless of the absolute error of the input ciphertexts. Using this method, modulus reduction becomes optional. However, it can still be useful because the ciphertext sizes and computation times become smaller as the modulus shrinks.

6.1.2 Bootstrapping

Even with all of the above techniques, homomorphic operations still always increase the error *rate* of a ciphertext, by as much as a polynomial factor per operation. Therefore, the schemes described so far can only homomorphically evaluate circuits of an a-priori bounded depth; such systems are frequently called "somewhat homomorphic" or "leveled" (we ignore the precise technical distinction between these terms).

A beautiful idea from Gentry's original work [Gen09b, Gen09a], called *bootstrapping* or sometimes *refreshing*, makes it possible to reduce the error *rate* of a ciphertext, thus enabling unbounded homomorphic computation. Suppose we have a ciphertext c that encrypts some (unknown) message μ under a secret key s, where the error rate of c is too large to perform further homomorphic operations on it. The idea behind bootstrapping is to *homomorphically evaluate the decryption function* on a low-error encryption $c_s = \mathsf{Enc}(s)$ of the secret key s, which is included as part of the public key. More specifically, we homomorphically evaluate the function $f_c(\cdot) = \mathsf{Dec}(\cdot, c)$ on the ciphertext c_s. (Note that the ciphertext c to be refreshed is "hard-coded" into the function $f_c(\cdot)$, whereas the secret key is treated as the function argument.) Because $f_c(s) = \mathsf{Dec}(s, c) = \mu$, it follows that homomorphic evaluation of f_c on an *encryption* of s (namely, c_s) produces an *encryption* of μ. Moreover, as long as the circuit depth of f_c and the error rate of c_s are small enough, the error rate of the output ciphertext will be substantially smaller than that of the input ciphertext c, as desired. In particular, decryption can be performed in $O(\log \lambda)$ depth, so it suffices for c_s to have some $\lambda^{-O(\log \lambda)}$ error rate.

Because bootstrapping involves the homomorphic evaluation of a somewhat complex function, it is not very efficient (see, e.g., [GH11b]). However, bootstrapping has been intensively studied and improved in various ways [GH11a, BGV12], culminating in ring-LWE-based methods that run in only polylogarithmic $\tilde{O}(1)$ time per encrypted bit [GHS12a, AP13], where the hidden $\log^{O(1)}(\lambda)$ factors are not exceedingly large.

We conclude this discussion by noting that, in order to yield *unbounded* homomorphic operations, bootstrapping requires a "circular" encryption of the secret key *under itself*. It is unknown whether it is provably secure (under a standard assumption) to reveal such an encryption, but nor are any attacks known. So to date, all unbounded FHE schemes require an additional assumption of circular security for the secret key.

6.1.3 Third-Generation FHE

In 2013, Gentry, Sahai, and Waters (hereafter GSW) [GSW13] proposed an interesting LWE-based FHE scheme that has some unique and advantageous properties. For example, homomorphic multiplication does not require any key-switching step, and the scheme can be made identity-based. Moreover, as shown in [BV14, AP14], the GSW scheme can be used to bootstrap with only a small *polynomial*-factor growth in the error rate, as opposed to quasi-polynomial $\lambda^{\Theta(\log \lambda)}$ growth for the system described above. This yields unbounded FHE based on LWE with just an inverse-polynomial $n^{-O(1)}$ error rate (plus a circular-security assumption). We describe these results in some detail next.

Homomorphic trapdoors. The GSW scheme is presented most simply in terms of the gadget-based trapdoors described in Section 5.4.3. (The presentation here has evolved through [BGG+14, AP14, GVW15b].) At the heart of GSW are the following additive and multiplicative homomorphisms for *tags* and *trapdoors*. Let $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times \bar{m}}$ be arbitrary, and for $i = 1, 2$ let

$$\mathbf{A}_i = x_i \mathbf{G} - \bar{\mathbf{A}} \mathbf{R}_i \tag{6.1.4}$$

for some integer x_i and short integer matrix $\mathbf{R}_i$. In other words, $\left[\begin{smallmatrix} \mathbf{R}_i \\ \mathbf{I} \end{smallmatrix}\right]$ is a trapdoor with tag x_i (or more accurately, tag $x_i \mathbf{I}$) for the matrix $[\bar{\mathbf{A}} \mid \mathbf{A}_i]$.

Applying Equation (6.1.4), it is easy to verify that

$$\mathbf{A}_1 + \mathbf{A}_2 = (x_1 + x_2)\mathbf{G} - \bar{\mathbf{A}}(\mathbf{R}_1 + \mathbf{R}_2), \tag{6.1.5}$$

and

$$\begin{aligned} \mathbf{A}_1 \cdot \mathbf{G}^{-1}(\mathbf{A}_2) &= (x_1 \mathbf{G} - \bar{\mathbf{A}} \mathbf{R}_1) \cdot \mathbf{G}^{-1}(\mathbf{A}_2) \\ &= x_1 \mathbf{A}_2 - \bar{\mathbf{A}} \cdot \mathbf{R}_1 \mathbf{G}^{-1}(\mathbf{A}_2) \\ &= x_1 x_2 \mathbf{G} - \bar{\mathbf{A}} \underbrace{\left(\mathbf{R}_1 \mathbf{G}^{-1}(\mathbf{A}_2) + x_1 \mathbf{R}_2\right)}_{\mathbf{R}}. \end{aligned} \tag{6.1.6}$$

In other words, $\left[\begin{smallmatrix} \mathbf{R}_1 + \mathbf{R}_2 \\ \mathbf{I} \end{smallmatrix}\right]$ is a trapdoor with tag $x_1 + x_2$ for the matrix $[\bar{\mathbf{A}} \mid \mathbf{A}_1 + \mathbf{A}_2]$, and $\left[\begin{smallmatrix} \mathbf{R} \\ \mathbf{I} \end{smallmatrix}\right]$ is a trapdoor with tag $x_1 x_2$ for the matrix $[\bar{\mathbf{A}} \mid \mathbf{A}_1 \cdot \mathbf{G}^{-1}(\mathbf{A}_2)]$. Note that in the latter case, we need x_1 to be a *small* integer in order to get a good-quality trapdoor.

One very important property of homomorphic multiplication is that the growth of the resulting trapdoor matrix $\mathbf{R}$ is *asymmetric* and *quasi-additive*: while $\mathbf{R}_1$ is expanded by some polynomial factor due to the multiplication by $\mathbf{G}^{-1}(\mathbf{A}_2)$, the $\mathbf{R}_2$ term is only multiplied by x_1. We discuss the implications of this below.

Fully homomorphic encryption. The GSW FHE scheme works as follows. As in prior systems, the public key is a matrix $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n \times \bar{m}}$ whose columns are LWE samples with some secret $\bar{\mathbf{s}} \in \mathbb{Z}^{n-1}$, i.e.,

$$\mathbf{s}^t \bar{\mathbf{A}} \approx \mathbf{0} \tag{6.1.7}$$

where $\mathbf{s} = (-\bar{\mathbf{s}}, 1)$ is the secret key. An encryption of an integer x is a *matrix*

$$\mathbf{A} = x\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}$$

for some sufficiently short random $\mathbf{R}$. As usual, semantic security follows by a lossiness argument: encrypting under a uniformly random ("malformed") public key $\bar{\mathbf{A}}$ statistically hides the message, because $[\bar{\mathbf{A}} \mid \bar{\mathbf{A}}\mathbf{R}]$ is very close to uniformly distributed.

To decrypt a ciphertext $\mathbf{A}$ using the secret key $\mathbf{s}$, one just computes

$$\mathbf{s}^t\mathbf{A} \approx x \cdot \mathbf{s}^t\mathbf{G},$$

where the approximation holds by Equation (6.1.7), and because $\mathbf{R}$ is short. Since it is easy to solve LWE with respect to $\mathbf{G}$, we can recover $x \cdot \mathbf{s}^t$ and hence x (modulo q). In fact, if x is known to be just a bit, then because one of the rightmost entries of $\mathbf{s}^t\mathbf{G}$ is far from zero modulo q, we can recover x simply by testing whether $\mathbf{s}^t \cdot \mathbf{a} \approx 0$ for an appropriate column $\mathbf{a}$ of $\mathbf{A}$.

To homomorphically add or multiply ciphertexts $\mathbf{A}_1, \mathbf{A}_2$ that encrypt small messages, one just computes $\mathbf{A}_1 + \mathbf{A}_2$ or $\mathbf{A}_1 \cdot \mathbf{G}^{-1}(\mathbf{A}_2)$, respectively. As shown above in Equations (6.1.5) and (6.1.6), the resulting ciphertext is a valid encryption for the sum or product of the messages, respectively. Because it is important for noise growth that messages remain small integers, we typically only use operations that maintain the messages as $\{0, 1\}$-values. For example, the binary NAND operation, which is sufficient for expressing arbitrary computations, can be written as $(x_1 \text{ NAND } x_2) = 1 - x_1x_2$.

Bootstrapping. As first shown by Brakerski and Vaikuntanathan [BV14], the asymmetric and quasi-additive growth of the trapdoor matrices under homomorphic multiplication allows certain computations—in particular, decryption for the purpose of *bootstrapping*—to be performed with rather small error growth. The first main observation is that by Equation (6.1.6), any polynomial-length chain of homomorphic bit-multiplications on fresh ciphertexts, if done in a *right-associative* manner, incurs only polynomial error growth. The same also holds when multiplying a sequence of *permutation* matrices (or more generally, orthonormal integer matrices), where each matrix is encrypted entry-wise.

The second main idea is that by using Barrington's Theorem [Bar86], any depth-d circuit can be converted into a length-4^d branching program of permutation matrices. In particular, the $O(\log \lambda)$-depth decryption circuit can be computed homomorphically in polynomial time and with polynomial error growth, albeit for rather large polynomials. The subsequent work of Alperin-Sheriff and Peikert [AP14] significantly improved the runtime and growth to small polynomials, by avoiding Barrington's Theorem and instead expressing decryption as an *arithmetic* function that can be embedded directly into permutation matrices. Ducas and Micciancio [DM15] devised and implemented a version of this method incorporating additional ideas, yielding a system that evaluates a complete "bootstrapped NAND gate" in less than a second on standard desktop hardware.

Fully homomorphic signatures. Gorbunov, Vaikuntanathan, and Wichs [GVW15b] showed how homomorphic trapdoors can also be used to obtain fully homomorphic *signatures* (FHS). The precise model and security goals for FHS are beyond the scope of this survey, but the basic idea is the following: the signer signs some initial data $x \in \{0, 1\}^\ell$ under its public key, producing a signature σ_x. Then, given only the public key, x, and σ_x, an untrusted worker can apply an arbitrary function f to x and compute a corresponding signature $\sigma_{f,y}$ for the value $y = f(x)$. A verifier, given only the public key, f, y, and $\sigma_{f,y}$ (but not x itself!), can verify that y is indeed the value of f on some data x that the signer originally signed.

Fully homomorphic signatures arise quite naturally from the above homomorphic trapdoors.[3] The public key is a uniformly random $\bar{\mathbf{A}}$, along with ℓ more uniformly random matrices $\mathbf{A}_i$, one for each bit of the

[3] Indeed, this construction can be seen as a direct analogue of the GSW encryption scheme and a related attribute-based encryption scheme of Boneh *et al.* [BGG+14], described below in Section 6.2.2.

original data to be signed. The secret key is a trapdoor for $\bar{\mathbf{A}}$. To sign data $x \in \{0,1\}^\ell$, the signer uses the trapdoor to sample a Gaussian-distributed $\mathbf{R}_i$ satisfying $\mathbf{A}_i = x_i\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_i$, and the signature is the collection of all these $\mathbf{R}_i$.

Homomorphic operations on signatures, and signature verification, are defined as follows. For any predicate $f\colon \{0,1\}^\ell \to \{0,1\}$, expressed without loss of generality as an arithmetic circuit of addition and multiplication gates, we define a matrix $\mathbf{A}_f$ which is computed recursively as follows:

- If $f(x) = x_i$ for some i, define $\mathbf{A}_f = \mathbf{A}_i$.
- If $f(x) = f_1(x) + f_2(x)$ for some predicates f_1, f_2, let $\mathbf{A}_f = \mathbf{A}_{f_1} + \mathbf{A}_{f_2}$.
- Otherwise, $f(x) = f_1(x) \cdot f_2(x)$ for some predicates f_1, f_2; let $\mathbf{A}_f = \mathbf{A}_{f_1} \cdot \mathbf{G}^{-1}(\mathbf{A}_{f_2})$.

Given some x and the corresponding signature components $\mathbf{R}_i$, to homomorphically derive a signature attesting that $f(x) = y$ for some predicate f, one just computes $\mathbf{A}_f$ along with, via Equations (6.1.5) and (6.1.6), a short $\mathbf{R}_{f,y}$ that satisfies $\mathbf{A}_f = y\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_{f,y}$. The verifier likewise computes $\mathbf{A}_f$ from the $\mathbf{A}_i$ and f alone (without needing to know x), and can verify that the presented $\mathbf{R}_{f,y}$ is sufficiently short and satisfies the above relation.

6.2 Attribute-Based Encryption

The concept of *attribute-based encryption* (ABE), introduced in [SW05, GPSW06], is a generalization of identity-based encryption. In one main variant of ABE, a ciphertext is encrypted under a set of *attributes*, and the master secret key can be used to generate secret keys for any *predicate* in a particular class. If a ciphertext's attributes satisfy a secret key's predicate, then the secret key can decrypt the ciphertext; otherwise the underlying message should remain hidden. (Identity-based encryption is the special case in which attributes correspond to the bits of an identity, and a predicate is an equality test with a particular identity.) Many ABE schemes have been devised based on cryptographic groups admitting bilinear pairings, starting from [GPSW06]. Soon after the identity-based lattice encryption schemes of [GPV08, CHKP10, ABB10], generalizations of these systems to the attribute-based setting emerged.

6.2.1 ABE for Inner-Product Predicates

A first example of ABE from lattices is a system for inner-product predicates over (large) finite fields $\mathbb{F}$, due to Agrawal, Freeman, and Vaikuntanathan [AFV11], with improvements by Xagawa [Xag13]. The construction inherits from the HIBE of [ABB10] described in Section 5.5.3, but key generation and decryption for a predicate involve linearly homomorphic operations on the master public key and ciphertext. Here we describe a version of these constructions.

- Recall from Section 5.4.3 that for prime q, the finite field $\mathbb{F} = \mathbb{F}_{q^n}$ is isomorphic to a certain matrix subring $\mathcal{H} \subseteq \mathbb{Z}_q^{n\times n}$. In particular, the difference between any two distinct matrices in $\mathcal{H}$ is invertible.
- The master public key consists of a (nearly) uniformly random $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n\times\bar{m}}$ that is generated with a trapdoor; a uniformly random
$$\mathbf{A} = [\mathbf{A}_1 \mid \cdots \mid \mathbf{A}_\ell] \in \mathbb{Z}_q^{n\times w\ell}$$
where each $\mathbf{A}_i \in \mathbb{Z}_q^{n\times w}$ has the same dimensions as the gadget matrix $\mathbf{G}$, and ℓ is the length of the attribute vectors over $\mathbb{F}$; and a uniformly random syndrome $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret key is the trapdoor for $\bar{\mathbf{A}} \in \mathbb{Z}_q^{n\times\bar{m}}$. (We describe secret keys for predicates after describing encryption.)

- For an attribute vector $\vec{h} = (\mathbf{H}_1, \ldots, \mathbf{H}_\ell) \in \mathcal{H}^\ell$, define the matrices $\mathbf{G}_{\vec{h}} = [\mathbf{H}_1\mathbf{G} \mid \cdots \mid \mathbf{H}_\ell\mathbf{G}]$ and

$$\mathbf{A}_{\vec{h}} = \mathbf{A} + \mathbf{G}_{\vec{h}}.$$

  To encrypt a bit μ under $\vec{h}$, we simply encrypt as in the dual LWE cryptosystem to the key $[\bar{\mathbf{A}} \mid \mathbf{A}_{\vec{h}} \mid \mathbf{u}]$:

$$[\bar{\mathbf{c}}^t \mid \mathbf{c}_{\vec{h}}^t \mid c] \approx \mathbf{s}^t \cdot [\bar{\mathbf{A}} \mid \mathbf{A}_{\vec{h}} \mid \mathbf{u}] + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil)^t, \tag{6.2.1}$$

  where $\mathbf{s}$ is a random LWE secret, the approximation hides appropriate small errors, and the components $\bar{\mathbf{c}}, \mathbf{c}_{\vec{h}}, c$ correspond to $\bar{\mathbf{A}}, \mathbf{A}_{\vec{h}}, \mathbf{u}$ as expected.

- To generate a secret key $\mathbf{x}_{\vec{p}}$ for a predicate vector $\vec{p} = (\mathbf{P}_1, \ldots, \mathbf{P}_\ell) \in \mathcal{H}^\ell$ using the trapdoor for $\bar{\mathbf{A}}$, define the short integer matrix

$$\mathbf{S}_{\vec{p}} = \begin{bmatrix} \mathbf{G}^{-1}(\mathbf{P}_1\mathbf{G}) \\ \vdots \\ \mathbf{G}^{-1}(\mathbf{P}_\ell\mathbf{G}) \end{bmatrix},$$

  define $\mathbf{B}_{\vec{p}} = \mathbf{A} \cdot \mathbf{S}_{\vec{p}}$, and generate a Gaussian-distributed solution $\mathbf{x}_{\vec{p}}$ to

$$[\bar{\mathbf{A}} \mid \mathbf{B}_{\vec{p}}] \cdot \mathbf{x}_{\vec{p}} = \mathbf{u}.$$

- For decryption, observe that for any $\vec{h}, \vec{p} \in \mathcal{H}^\ell$, we have

$$\mathbf{A}_{\vec{h}} \cdot \mathbf{S}_{\vec{p}} = (\mathbf{A} + \mathbf{G}_{\vec{h}}) \cdot \mathbf{S}_{\vec{p}} = \mathbf{B}_{\vec{p}} + \langle \vec{h}, \vec{p} \rangle \cdot \mathbf{G}.$$

  Therefore, to decrypt a ciphertext $(\bar{\mathbf{c}}, \mathbf{c}_{\vec{h}}, c)$ having attribute vector $\vec{h}$ using a secret key $\mathbf{x}_{\vec{p}}$ for some predicate vector $\vec{p}$ such that $\langle \vec{h}, \vec{p} \rangle = \mathbf{0}$, we compute

$$[\bar{\mathbf{c}}^t \mid \mathbf{c}_{\vec{h}}^t \cdot \mathbf{S}_{\vec{p}}] \cdot \mathbf{x}_{\vec{p}} \approx \mathbf{s}^t \cdot [\bar{\mathbf{A}} \mid \mathbf{B}_{\vec{p}}] \cdot \mathbf{x}_{\vec{p}} = \mathbf{s}^t \cdot \mathbf{u} \approx c - \mu \cdot \lfloor \tfrac{q}{2} \rceil$$

  and recover μ, where the approximations hold by Equation (6.2.1) and because $\mathbf{S}_{\vec{p}}$ and $\mathbf{x}_{\vec{p}}$ are short.

  (Notice also that if $\langle \vec{h}, \vec{p} \rangle \neq 0$, then $\mathbf{A}_{\vec{h}} \cdot \mathbf{S}_{\vec{p}}$ includes a nonzero multiple of $\mathbf{G}$ with $\mathbf{B}_{\vec{p}}$, which prevents decryption using $\mathbf{x}_{\vec{p}}$.)

Under an appropriate LWE assumption, the above scheme is semantically secure under a *selective attribute* attack, in which the adversary must name the target attribute vector $\vec{h}^* \in \mathcal{H}^\ell$ before seeing the master public key, and may request a secret key for any predicate vector $\vec{p} \in \mathcal{H}^\ell$ such that $\langle \vec{h}^*, \vec{p} \rangle \neq \mathbf{0}$. The security proof closely mirrors the one described in Section 5.5.4: the simulator constructs the matrix $\bar{\mathbf{A}}$ to come from its input samples, and constructs the remaining master public key matrices as $\mathbf{A}_i = -\mathbf{H}_i^*\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_i$, thus statistically hiding the tags of $\vec{h}^*$ in these matrices. This setup "punctures" the public key so that for any legal predicate query $\vec{p} \in \mathbb{F}^\ell$ (for which $\langle \vec{h}^*, \vec{p} \rangle \neq \mathbf{0}$), the reduction knows a short trapdoor

$$\mathbf{R}_{\vec{p}} = \sum_i \mathbf{R}_i \cdot \mathbf{G}^{-1}(\mathbf{P}_i\mathbf{G}) \quad \text{for} \quad [\bar{\mathbf{A}} \mid \mathbf{B}_{\vec{p}}] = [\bar{\mathbf{A}} \mid -\langle \vec{h}^*, \vec{p} \rangle \mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_{\vec{p}}],$$

so the reduction can generate a properly distributed secret key $\mathbf{x}_{\vec{p}}$. Moreover, because the master public key is punctured at $\vec{h}^*$, the reduction can use its LWE challenge and the $\mathbf{R}_i$ to generate a challenge ciphertext for $[\bar{\mathbf{A}} \mid \mathbf{A}_{\vec{h}} \mid \mathbf{u}] = [\bar{\mathbf{A}} \mid -\bar{\mathbf{A}}\mathbf{R}_1 \mid \cdots \mid -\bar{\mathbf{A}}\mathbf{R}_\ell \mid \mathbf{u}]$ which is either properly distributed or which statistically hides the message, depending on whether the challenge is LWE-distributed or uniform, respectively.

6.2.2 ABE for Arbitrary Circuits

In prior works on ABE from bilinear pairings, the class of supported predicates was somewhat limited, to boolean *formulas*. A work of Gorbunov, Vaikuntanathan, and Wee [GVW13] constructed from LWE an ABE for arbitrary predicates expressed as *circuits* of any *a priori* bounded depth, which is fixed at system setup. In this system, the secret key for a predicate grows proportionally to the size of the circuit, and corresponds roughly to a "garbled circuit" for the predicate, for which a ciphertext and its attributes provides the "garbled input."

Soon after [GVW13], Boneh *et al.* [BGG$^+$14] combined ideas from GSW fully homomorphic encryption (Section 6.1.3) and the ABE for inner-product predicates (Section 6.2.1) to construct an ABE for arbitrary circuits of any *a priori* bounded depth. In contrast to [GVW13], here the secret key size grows only with the *depth* (not the size) of its predicate. Here we describe a version of the construction from [BGG$^+$14].

- Setup is exactly as in the system from Section 6.2.1: the master public key consists of a (nearly) uniformly random matrix $\bar{\mathbf{A}}$, generated with a trapdoor; uniformly random matrices $\mathbf{A}_1, \ldots, \mathbf{A}_\ell$ having the same dimensions as $\mathbf{G}$, where ℓ is the number of attribute bits; and a uniformly random syndrome $\mathbf{u} \in \mathbb{Z}_q^n$. The master secret key is a trapdoor for $\bar{\mathbf{A}}$. (We discuss secret keys for predicates after describing encryption.)

- Encryption is essentially the same as in the ABE from Section 6.2.1, except that here we use only binary attributes/tags. Specifically, to encrypt a message bit $\mu \in \{0,1\}$ under an attribute vector $x \in \{0,1\}^\ell$, output a ciphertext of the form

$$[\bar{\mathbf{c}}^t \mid \mathbf{c}_1^t \mid \cdots \mid \mathbf{c}_\ell^t \mid c] \approx \mathbf{s}^t \cdot [\bar{\mathbf{A}} \mid \mathbf{A}_1 - x_1\mathbf{G} \mid \cdots \mid \mathbf{A}_\ell - x_\ell\mathbf{G} \mid \mathbf{u}] + (\mathbf{0}, \mu \cdot \lfloor \tfrac{q}{2} \rceil)^t,$$

  where $\mathbf{s} \in \mathbb{Z}_q^n$ is a random LWE secret, the approximation hides appropriate small errors, and the ciphertext components $\bar{\mathbf{c}}, \mathbf{c}_1, \ldots, \mathbf{c}_\ell, c$ correspond to $\bar{\mathbf{A}}, \mathbf{A}_1, \ldots, \mathbf{A}_\ell, \mathbf{u}$ in the expected way.

- For a predicate $f\colon \{0,1\}^\ell \to \{0,1\}$, viewed without loss of generality as a binary circuit of XOR (addition) and AND (multiplication) gates, we recursively define a matrix $\mathbf{A}_f$ exactly as in the fully homomorphic signature scheme of Section 6.1.3:

  - In the base case where $f(x) = x_i$ for some i, let $\mathbf{A}_f = \mathbf{A}_i$. Clearly, for any $x \in \{0,1\}^\ell$ we have

$$\mathbf{A}_i - x_i\mathbf{G} = \mathbf{A}_f - f(x)\mathbf{G}. \tag{6.2.2}$$

  - If $f = f_1 + f_2$ for some predicates f_1, f_2, let $\mathbf{A}_f = \mathbf{A}_{f_1} + \mathbf{A}_{f_2}$. Observe that for any $x \in \{0,1\}^\ell$, we have

$$[\mathbf{A}_{f_1} - f_1(x)\mathbf{G} \mid \mathbf{A}_{f_2} - f_2(x)\mathbf{G}] \cdot \underbrace{\begin{bmatrix}\mathbf{I}\\ \mathbf{I}\end{bmatrix}}_{\text{short}} = \mathbf{A}_f - f(x)\mathbf{G}. \tag{6.2.3}$$

  - Otherwise, $f = f_1 \cdot f_2$ for some predicates f_1, f_2, and we let $\mathbf{A}_f = \mathbf{A}_{f_1} \cdot \mathbf{G}^{-1}(\mathbf{A}_{f_2})$. Observe that for any $x \in \{0,1\}^\ell$, we have

$$[\mathbf{A}_{f_1} - f_1(x)\mathbf{G} \mid \mathbf{A}_{f_2} - f_2(x)\mathbf{G}] \cdot \underbrace{\begin{bmatrix}\mathbf{G}^{-1}(\mathbf{A}_{f_2})\\ f_1(x)\mathbf{I}\end{bmatrix}}_{\text{short}} = \mathbf{A}_f - f(x)\mathbf{G}. \tag{6.2.4}$$

To generate a secret key for a predicate f using the trapdoor for $\bar{\mathbf{A}}$, we sample a Gaussian-distributed integer solution $\mathbf{k}_f$ to

$$[\bar{\mathbf{A}} \mid \mathbf{A}_f] \cdot \mathbf{k}_f = \mathbf{u}. \tag{6.2.5}$$

- To decrypt a ciphertext $(\bar{\mathbf{c}}, \mathbf{c}_1, \ldots, \mathbf{c}_l, c)$ having known attributes $x \in \{0,1\}^\ell$ using a secret key $\mathbf{k}_f$ for some predicate f where $f(x) = 0$, we first operate on the $\mathbf{c}_i^t \approx \mathbf{s}^t(\mathbf{A}_i - x_i\mathbf{G})$ components to compute

  $$\mathbf{c}_{f,x}^t \approx \mathbf{s}^t(\mathbf{A}_f - f(x)\mathbf{G}) = \mathbf{s}^t\mathbf{A}_f,$$

  in a manner described below. We then recover the message from

  $$(\bar{\mathbf{c}}, \mathbf{c}_{f,x})^t \cdot \mathbf{k}_f \approx \mathbf{s}^t[\bar{\mathbf{A}} \mid \mathbf{A}_f] \cdot \mathbf{k}_f = \mathbf{s}^t \cdot \mathbf{u} \approx c - \mu \cdot \lfloor \tfrac{q}{2} \rceil,$$

  where the approximation holds because $\mathbf{k}_f$ is short. (Note that if $f(x) \neq 0$, the nonzero $\mathbf{G}$-multiple appearing in $\mathbf{c}_{f,x}$ prevents decryption using $\mathbf{k}_f$.)

  To obtain $\mathbf{c}_{f,x}$, we just recursively apply Equations (6.2.2), (6.2.3), and (6.2.4) to obtain

  $$\mathbf{c}_{g,x}^t \approx \mathbf{s}^t(\mathbf{A}_g - g(x)\mathbf{G})$$

  for every sub-predicate g involved in the computation of f. For example, if $g = g_1 \cdot g_2$ then we define

  $$\mathbf{c}_{g,x}^t = [\mathbf{c}_{g_1,x}^t \mid \mathbf{c}_{g_2,x}^t] \cdot \begin{bmatrix} \mathbf{G}^{-1}(\mathbf{A}_{g_2}) \\ g_1(x)\mathbf{I} \end{bmatrix}.$$

  Note that the approximations are maintained because the matrices we multiply by in Equations (6.2.3) and (6.2.4) are short. Note also that to apply Equation (6.2.4), we need to know the value of $g_1(x)$, which is why the attribute vector x needs to be known to the decryption algorithm.

Under an appropriate LWE assumption, the above scheme is semantically secure under a selective-attribute attack, in which the adversary must name the target attribute vector x^* before seeing the master public key, and may request a secret key for any predicate f such that $f(x^*) \neq 0$. The security proof proceeds very similarly to the ones described in Sections 5.5.4 and 6.2.1, but here the simulator can do arbitrary (rather than just linear) operations on the tags and hidden trapdoors. More specifically, the matrix $\bar{\mathbf{A}}$ is constructed from the simulator's input samples, and the simulator sets up the remaining master public key as $\mathbf{A}_i = x_i^*\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_i$ for some short matrices $\mathbf{R}_i$. This setup "punctures" the master public key so that for any legal predicate query f (for which $f(x^*) = 1 \neq 0$), by using Equations (6.1.5) and (6.1.6) with its $\mathbf{R}_i$ the simulator can compute a short $\mathbf{R}_f$ for which $\mathbf{A}_f = f(x^*)\mathbf{G} - \bar{\mathbf{A}}\mathbf{R}_f$. Therefore, the simulator knows a short trapdoor for $[\bar{\mathbf{A}} \mid \mathbf{A}_f]$, which allows it to answer the query by sampling a Gaussian solution to Equation (6.2.5). Moreover, because the master public key is punctured at x^*, the simulator can use the $\mathbf{b}^t$ component of its input and the $\mathbf{R}_i$ to generate a challenge ciphertext which is either properly distributed or which statistically hides the message, depending on whether the challenge is LWE-distributed or uniform, respectively.

Predicate encryption. We conclude this chapter by mentioning that a recent work of Gorbunov, Vaikuntanathan, and Wee [GVW15a] constructs *predicate encryption* (PE) based on the LWE assumption. In brief, predicate encryption is a strengthening of attribute-based encryption in which the attributes associated with a ciphertext remain hidden from (any coalition of) users who are not authorized to decrypt the ciphertext. In the

construction of Gorbunov *et al.*, the encryption algorithm uses an FHE to *encrypt* the attribute vector x (under a fresh key, chosen by the encrypter), while the decryption algorithm performs a *homomorphic evaluation* of the predicate f on the encrypted x. Of course, the resulting value $f(x) \in \{0, 1\}$ remains encrypted, yet recovery of the message should be conditioned on this bit. This issue is addressed as follows: recall that FHE decryption is a relatively lightweight process, consisting of a rounded mod-q inner product of the ciphertext with the secret key. Also recall that we have attribute-hiding ABE schemes for inner-product predicates (see Section 6.2.1). Therefore, we augment the PE encryption algorithm to also include the FHE secret key coordinates as *hidden* attributes, and correspondingly extend the PE decryption algorithm to compute an inner product of the secret key with the final FHE ciphertext encrypting $f(x)$. Finally, *rounding* the inner product is done by shifting it by each of the roughly $q/2$ values in $\mathbb{Z}_q$ that correspond to an encryption of zero. Note that to do this, the decrypter needs a separate secret key for each shift, and learns the exact value of the inner product when decryption succeeds; this is why the PE scheme hides attributes from unauthorized users, but not authorized ones.

Chapter 7

Open Questions

There are many fascinating questions and open problems relating to (ring-)SIS/LWE and their applications. In this chapter we describe a small selection.

7.1 Foundations

Learning with errors or rounding. Recall from Section 4.2 that LWE with Gaussian error of rate $\alpha < 1$ and any modulus $q \geq 2\sqrt{n}/\alpha$ is *quantumly* at least as hard as the GapSVP_γ and SIVP_γ problems, for $\gamma = \tilde{O}(n/\alpha)$ [Reg05]. By contrast, the known *classical* reduction from [Pei09] works only for GapSVP, not SIVP, and requires an exponentially large modulus q (or a dimension-modulus tradeoff that does not decrease the bit length of the samples [BLP+13]). This state of affairs leaves open the following important problem.

Question 1. *Is there a* classical *worst-case hardness reduction for LWE that fully subsumes the known* quantum *reduction?*

Recall from Section 5.7.1 that for appropriate parameters, the learning with rounding (LWR) problem is at least as hard as LWE. However, the known reductions [BPR12, AKPW13] involve either a super-polynomial modulus q and inverse error rate $1/\alpha$ for LWE (corresponding to super-polynomial approximation factors for lattice problems), or an *a priori* bound on the number of LWR samples (which affects the underlying LWE parameters). However, LWR appears to be hard for much more aggressive parameters, e.g., sufficiently large and integral $q/p = \mathrm{poly}(n)$. The situation is similar for all the LWE-based pseudorandom function (PRF) constructions of [BPR12, BLMR13, BP14].

Question 2. *Do LWR, and more generally, existing PRF constructions from LWE, have worst-case hardness for polynomially bounded q and an unbounded number of samples?*

Ideal lattices and ring-LWE. Turning now to ring-based cryptography, perhaps the most important question for security is whether the additional algebraic structure of ideals and modules over rings opens the door to any new kinds of attacks on the relevant lattice problems. For ring-SIS/LWE cryptography in particular, a central question is the following:

Question 3. *For worst-case problems on* ideal *lattices, especially in cyclotomic rings, are there (possibly quantum) algorithms that substantially outperform the known ones for general lattices? If so, do these attacks also extend to work against the ring-SIS and ring-LWE problems themselves?*

For ring-LWE, no meaningful analogue of the classical LWE hardness reduction of [Pei09] is known, because GapSVP for ideal lattices is *easy* for the relevant polynomial approximation factors. This leads to the following very important question.

Question 4. *Is there a meaningful* classical *worst-case hardness reduction for ring-LWE?*

While the known worst-case reduction for the *search* version of ring-LWE works in *arbitrary* number fields (see [LPR10, Theorem 4.1]), the search-decision equivalence from [LPR10, Section 5] relies centrally on *cyclotomic* number fields—specifically, the fact that they are Galois (they have many automorphisms), which is a rather rare property.

Question 5. *Is there a search-to-decision reduction for ring-LWE, or some other hardness reduction for decisional ring-LWE, in number fields that do not have many automorphisms?*

The known worst-case hardness reduction for the search version of ring-LWE involves a family of elliptical error distributions. For this reason, the search-decision reduction (for cyclotomic rings) involves a decision problem where the error distribution either has n *random* and *secret* parameters, or is spherical with a parameter that depends on the number of samples available to the distinguisher (see [LPR10, Theorems 5.1 and 5.2]). It would be preferable to demonstrate that the search problem is hard for a fixed *spherical* error distribution, because, in addition to its simplicity, for spherical error the search and decision problems are equivalent, with no degradation in the width of the error (see [LPR10, Theorem 5.3]).

Question 6. *Is there a worst-case hardness reduction for ring-LWE (either search or decision) with a narrow* spherical *error distribution, where the amount of error does not need to grow with the number of samples?*

(We mention that a standard "noise flooding" technique can prove hardness for spherical error having *super-polynomial* width, but using this technique comes at an unsatisfactory cost in efficiency and worst-case approximation factors.)

The known search-decision reductions for ring-LWE from [LPR10, Section 5] convert a distinguisher that solves the decision problem with some non-negligible advantage into an algorithm that solves the search problem with high probability, but using many more samples than the distinguisher needs. This increase is not a problem as far as worst-case hardness is concerned, but it is undesirable for relating the pseudorandomness of ring-LWE to its one-wayness as a cryptographic function. For plain LWE, a work by Micciancio and Mol [MM11] (building on [IN96] for the subset-sum function) gave a *sample-preserving* search-decision reduction, in which the solver of the search problem uses the same number of samples as the distinguisher (and has success probability polynomially related to the distinguisher's advantage).

Question 7. *Is there a* sample-preserving *search-decision equivalence for ring-LWE?*

NTRU. Recall from Section 5.2.4 that Stehlé and Steinfeld [SS11] proposed a variant of the NTRU cryptosystem in which the public key $g/f \in R_q$ is statistically close to uniformly random (owing to the relatively large sizes of f, g), and proved it to be passively secure under a ring-LWE assumption. However, NTRU is typically defined so that the public key is statistically very far from uniform, but is conjectured to be pseudorandom—yet we currently have little theoretical evidence to support this conjecture. Indeed, we do not even know of a search-decision equivalence for problems associated with NTRU.

Question 8. *Is there a worst-case hardness reduction, or a search-to-decision reduction, for an NTRU-like problem?*

7.2 Cryptographic Applications

Adaptive security. The identity-based, attribute-based, and predicate encryption schemes (in the standard model) described in Chapters 5 and 6 are analyzed under somewhat artificial model of *selective* attacks, in which the adversary must name the identity or attributes it intends to attack before seeing any of the public parameters. A more realistic model is *adaptive* security, wherein the adversary can name its target after seeing the master public key and making queries. While some lattice-based IBE schemes can be adapted to obtain adaptive security (without using random oracles; see, e.g., [CHKP10]), this comes at a rather high cost in the size of the master public key and the tightness of the security reduction, and it is unclear whether similar techniques work for ABE/PE. In the literature on bilinear pairings, the "dual system" methodology of Waters [Wat09] (further developed in many follow-up works) yields adaptively secure IBE/ABE systems that are comparably efficient to their selectively secure precursors. However, a lattice analogue of the dual-system methodology has so far remained elusive.

Question 9. *Are there standard-model, adaptively secure lattice-based IBE/ABE/PE schemes that have comparable efficiency and concrete security to the existing selectively secure ones? Is there an analogue of the dual-system methodology for lattices?*

Unbounded FHE. Recall from Section 6.1.2 that all known fully homomorphic encryption (FHE) schemes follow the same basic template from Gentry's initial work [Gen09b, Gen09a]: first, one constructs a "somewhat homomorphic" scheme that supports only a bounded amount of homomorphic computation on fresh ciphertexts, then one applies a "bootstrapping" transformation to convert the scheme into one that can handle unbounded homomorphic computation. Despite significant advances, bootstrapping is computationally quite expensive, because it involves homomorphically evaluating the entire decryption function. In addition, bootstrapping for unbounded FHE requires one to make a "circular security" assumption, i.e., that it is secure to reveal an encryption of (a suitable encoding of) the secret key under itself. To date, such assumptions are poorly understood, and we have little theoretical evidence to support them (in particular, no worst-case hardness).

Question 10. *Is there an unbounded FHE scheme that does not rely on bootstrapping? Is there a version of bootstrapping that uses a lighter-weight computation than full decryption?*

Question 11. *Is there an unbounded FHE scheme that can be proved secure solely under a worst-case complexity assumption?*

Unbounded attribute-based encryption and fully homomorphic signatures. Recall from Section 6.2 that the known attribute-based and predicate encryption schemes for arbitrary circuits are "leveled," i.e., for every circuit depth, there is a scheme that supports any access policy computable in that depth. The fully homomorphic signature scheme described in Section 6.1.3 is subject to a similar caveat. As a matter of theoretical feasibility, it would be preferable to have a *single* scheme that can handle *any* efficiently computable access policy. In the context of fully homomorphic encryption, bootstrapping provides a way to "reverse the quantifiers" for a leveled scheme. But despite the strong similarities between FHE and ABE/PE/FHS schemes, to date no bootstrapping technique is known for the latter.

Question 12. *Is there an ABE/PE/FHS scheme for all efficiently computable functions? Can such schemes be "bootstrapped?"*

Bibliography

[ABB10] S. Agrawal, D. Boneh, and X. Boyen. Efficient lattice (H)IBE in the standard model. In *EUROCRYPT*, pages 553–572. 2010.

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

[AD97] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *STOC*, pages 284–293. 1997.

[AD07] M. Ajtai and C. Dwork. The first and fourth public-key cryptosystems with worst-case/average-case equivalence. *Electronic Colloquium on Computational Complexity (ECCC)*, 14(97), 2007.

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

[AFV11] S. Agrawal, D. M. Freeman, and V. Vaikuntanathan. Functional encryption for inner product predicates from learning with errors. In *ASIACRYPT*. 2011.

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

[Ajt96] M. Ajtai. Generating hard instances of lattice problems. *Quaderni di Matematica*, 13:1–32, 2004. Preliminary version in STOC 1996.

[Ajt98] M. Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions (extended abstract). In *STOC*, pages 10–19. 1998.

[Ajt99] M. Ajtai. Generating hard instances of the short basis problem. In *ICALP*, pages 1–9. 1999.

[Ajt05] M. Ajtai. Representing hard lattices with $O(n \log n)$ bits. In *STOC*, pages 94–103. 2005.

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

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

[Ale03] M. Alekhnovich. More on average case vs approximation complexity. In *FOCS*, pages 298–307. 2003.

[Alp15] J. Alperin-Sheriff. Short signatures with short public keys from homomorphic trapdoor functions. In *PKC*, pages 236–255. 2015.

[AP09] J. Alwen and C. Peikert. Generating shorter bases for hard random lattices. *Theory of Computing Systems*, 48(3):535–553, April 2011. Preliminary version in STACS 2009.

[AP12] J. Alperin-Sheriff and C. Peikert. Circular and KDM security for identity-based encryption. In *Public Key Cryptography*, pages 334–352. 2012.

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

[AP14] J. Alperin-Sheriff and C. Peikert. Faster bootstrapping with polynomial error. In *CRYPTO*, pages 297–314. 2014.

[AR04] D. Aharonov and O. Regev. Lattice problems in NP $\cap$ coNP. *J. ACM*, 52(5):749–765, 2005. Preliminary version in FOCS 2004.

[Bab85] L. Babai. On Lovász' lattice reduction and the nearest lattice point problem. *Combinatorica*, 6(1):1–13, 1986. Preliminary version in STACS 1985.

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

[Ban95] W. Banaszczyk. Inequalites for convex bodies and polar reciprocal lattices in R^n. *Discrete & Computational Geometry*, 13:217–231, 1995.

[Bar86] D. A. M. Barrington. Bounded-width polynomial-size branching programs recognize exactly those languages in NC^1. In *STOC*, pages 1–5. 1986.

[Bar12] Website for the Bar-Ilan winter school on lattice-based cryptography and applications, 2012. `http://crypto.biu.ac.il/winterschool2012/`.

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

[BF01] D. Boneh and M. K. Franklin. Identity-based encryption from the Weil pairing. *SIAM J. Comput.*, 32(3):586–615, 2003. Preliminary version in CRYPTO 2001.

[BF11] D. Boneh and D. M. Freeman. Linearly homomorphic signatures over binary fields and new tools for lattice-based signatures. In *Public Key Cryptography*, pages 1–16. 2011.

[BFKL93] A. Blum, M. L. Furst, M. J. Kearns, and R. J. Lipton. Cryptographic primitives based on hard learning problems. In *CRYPTO*, pages 278–291. 1993.

[BFP+15] A. Banerjee, G. Fuchsbauer, C. Peikert, K. Pietrzak, and S. Stevens. Key-homomorphic constrained pseudorandom functions. In *TCC*, pages 31–60. 2015.

[BGG+14] D. Boneh, C. Gentry, S. Gorbunov, S. Halevi, V. Nikolaenko, G. Segev, V. Vaikuntanathan, and D. Vinayagamurthy. Fully key-homomorphic encryption, arithmetic circuit ABE and compact garbled circuits. In *EUROCRYPT*, pages 533–556. 2014.

[BGH07] D. Boneh, C. Gentry, and M. Hamburg. Space-efficient identity based encryption without pairings. In *FOCS*, pages 647–657. 2007.

[BGI14] E. Boyle, S. Goldwasser, and I. Ivan. Functional signatures and pseudorandom functions. In *PKC*, pages 501–519. 2014.

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

[BGV12] Z. Brakerski, C. Gentry, and V. Vaikuntanathan. (Leveled) fully homomorphic encryption without bootstrapping. *TOCT*, 6(3):13, 2014. Preliminary version in ITCS 2012.

[BKPW12] M. Bellare, E. Kiltz, C. Peikert, and B. Waters. Identity-based (lossy) trapdoor functions and applications. In *EUROCRYPT*, pages 228–245. 2012.

[BLMR13] D. Boneh, K. Lewi, H. W. Montgomery, and A. Raghunathan. Key homomorphic PRFs and their applications. In *CRYPTO*, pages 410–428. 2013.

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

[Boy10] X. Boyen. Lattice mixing and vanishing trapdoors: A framework for fully secure short signatures and more. In *Public Key Cryptography*, pages 499–517. 2010.

[BP14] A. Banerjee and C. Peikert. New and improved key-homomorphic pseudorandom functions. In *CRYPTO*, pages 353–370. 2014.

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

[BR93] M. Bellare and P. Rogaway. Random oracles are practical: A paradigm for designing efficient protocols. In *ACM Conference on Computer and Communications Security*, pages 62–73. 1993.

[Bra12] Z. Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In *CRYPTO*, pages 868–886. 2012.

[BV11a] Z. Brakerski and V. Vaikuntanathan. Fully homomorphic encryption from ring-LWE and security for key dependent messages. In *CRYPTO*, pages 505–524. 2011.

[BV11b] Z. Brakerski and V. Vaikuntanathan. Efficient fully homomorphic encryption from (standard) LWE. *SIAM J. Comput.*, 43(2):831–871, 2014. Preliminary version in FOCS 2011.

[BV14] Z. Brakerski and V. Vaikuntanathan. Lattice-based FHE as secure as PKE. In *ITCS*, pages 1–12. 2014.

[BV15] Z. Brakerski and V. Vaikuntanathan. Constrained key-homomorphic PRFs from standard lattice assumptions - or: How to secretly embed a circuit in your PRF. In *TCC*, pages 1–30. 2015.

[BW13] D. Boneh and B. Waters. Constrained pseudorandom functions and their applications. In *ASIACRYPT*, pages 280–300. 2013.

[CCK+13] J. H. Cheon, J.-S. Coron, J. Kim, M. S. Lee, T. Lepoint, M. Tibouchi, and A. Yun. Batch fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 315–335. 2013.

[CGH98] R. Canetti, O. Goldreich, and S. Halevi. The random oracle methodology, revisited. *J. ACM*, 51(4):557–594, 2004. Preliminary version in STOC 1998.

[CGH+15] J. Coron, C. Gentry, S. Halevi, T. Lepoint, H. K. Maji, E. Miles, M. Raykova, A. Sahai, and M. Tibouchi. Zeroizing without low-level zeroes: New MMAP attacks and their limitations. In *CRYPTO*, pages 247–266. 2015.

[CHKP10] D. Cash, D. Hofheinz, E. Kiltz, and C. Peikert. Bonsai trees, or how to delegate a lattice basis. *J. Cryptology*, 25(4):601–639, 2012. Preliminary version in Eurocrypt 2010.

[CHL+15] J. H. Cheon, K. Han, C. Lee, H. Ryu, and D. Stehlé. Cryptanalysis of the multilinear map over the integers. In *EUROCRYPT*, pages 3–12. 2015.

[CL15] J. H. Cheon and C. Lee. Cryptanalysis of the multilinear map on the ideal lattices. Cryptology ePrint Archive, Report 2015/461, 2015. `http://eprint.iacr.org/`.

[CLT13] J. Coron, T. Lepoint, and M. Tibouchi. Practical multilinear maps over the integers. In *CRYPTO*, pages 476–493. 2013.

[CLT15] J. Coron, T. Lepoint, and M. Tibouchi. New multilinear maps over the integers. In *CRYPTO*, pages 267–286. 2015.

[CMNT11] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi. Fully homomorphic encryption over the integers with shorter public keys. In *CRYPTO*, pages 487–504. 2011.

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

[CNT12] J.-S. Coron, D. Naccache, and M. Tibouchi. Public key compression and modulus switching for fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 446–464. 2012.

[Coc01] C. Cocks. An identity based encryption scheme based on quadratic residues. In *IMA Int. Conf.*, pages 360–363. 2001.

[CS97] D. Coppersmith and A. Shamir. Lattice attacks on NTRU. In *EUROCRYPT*, pages 52–61. 1997.

[DD12] L. Ducas and A. Durmus. Ring-LWE in polynomial rings. In *Public Key Cryptography*, pages 34–51. 2012.

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

[DGK+10] Y. Dodis, S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Public-key encryption schemes with auxiliary inputs. In *TCC*, pages 361–381. 2010.

[DH76] W. Diffie and M. E. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, IT-22(6):644–654, 1976.

[DM13] N. Döttling and J. Müller-Quade. Lossy codes and a new variant of the learning-with-errors problem. In *EUROCRYPT*, pages 18–34. 2013.

[DM14] L. Ducas and D. Micciancio. Improved short lattice signatures in the standard model. In *CRYPTO*, pages 335–352. 2014.

[DM15] L. Ducas and D. Micciancio. FHEW: Bootstrapping homomorphic encryption in less than a second. In *EUROCRYPT*, pages 617–640. 2015.

[DN12a] L. Ducas and P. Q. Nguyen. Faster Gaussian lattice sampling using lazy floating-point arithmetic. In *ASIACRYPT*, pages 415–432. 2012.

[DN12b] L. Ducas and P. Q. Nguyen. Learning a zonotope and more: Cryptanalysis of NTRUSign countermeasures. In *ASIACRYPT*, pages 433–450. 2012.

[DP15] L. Ducas and T. Prest. A hybrid Gaussian sampler for lattices over rings. Cryptology ePrint Archive, Report 2015/660, 2015. `http://eprint.iacr.org/`.

[FO99a] E. Fujisaki and T. Okamoto. How to enhance the security of public-key encryption at minimum cost. In *Public Key Cryptography*, pages 53–68. 1999.

[FO99b] E. Fujisaki and T. Okamoto. Secure integration of asymmetric and symmetric encryption schemes. In *CRYPTO*, pages 537–554. 1999.

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

[Gen09a] C. Gentry. *A fully homomorphic encryption scheme*. Ph.D. thesis, Stanford University, 2009. `http://crypto.stanford.edu/craig`.

[Gen09b] C. Gentry. Fully homomorphic encryption using ideal lattices. In *STOC*, pages 169–178. 2009.

[Gen10a] C. Gentry. Computing arbitrary functions of encrypted data. *Commun. ACM*, 53(3):97–105, 2010.

[Gen10b] C. Gentry. Toward basing fully homomorphic encryption on worst-case hardness. In *CRYPTO*, pages 116–137. 2010.

[GG98] O. Goldreich and S. Goldwasser. On the limits of nonapproximability of lattice problems. *J. Comput. Syst. Sci.*, 60(3):540–563, 2000. Preliminary version in STOC 1998.

[GGH97] O. Goldreich, S. Goldwasser, and S. Halevi. Public-key cryptosystems from lattice reduction problems. In *CRYPTO*, pages 112–131. 1997.

[GGH13a] S. Garg, C. Gentry, and S. Halevi. Candidate multilinear maps from ideal lattices. In *EUROCRYPT*, pages 1–17. 2013.

[GGH+13b] S. Garg, C. Gentry, S. Halevi, M. Raykova, A. Sahai, and B. Waters. Candidate indistinguishability obfuscation and functional encryption for all circuits. In *FOCS*, pages 40–49. 2013.

[GGH15] C. Gentry, S. Gorbunov, and S. Halevi. Graph-induced multilinear maps from lattices. In *TCC*, pages 498–527. 2015.

[GGM84] O. Goldreich, S. Goldwasser, and S. Micali. How to construct random functions. *J. ACM*, 33(4):792–807, 1986. Preliminary version in FOCS 1984.

[GH11a] C. Gentry and S. Halevi. Fully homomorphic encryption without squashing using depth-3 arithmetic circuits. In *FOCS*, pages 107–109. 2011.

[GH11b] C. Gentry and S. Halevi. Implementing Gentry's fully-homomorphic encryption scheme. In *EUROCRYPT*, pages 129–148. 2011.

[GHPS12] C. Gentry, S. Halevi, C. Peikert, and N. P. Smart. Field switching in BGV-style homomorphic encryption. *Journal of Computer Security*, 21(5):663–684, 2013. Preliminary version in SCN 2012.

[GHS12a] C. Gentry, S. Halevi, and N. P. Smart. Better bootstrapping in fully homomorphic encryption. In *Public Key Cryptography*, pages 1–16. 2012.

[GHS12b] C. Gentry, S. Halevi, and N. P. Smart. Fully homomorphic encryption with polylog overhead. In *EUROCRYPT*, pages 465–482. 2012.

[GKPV10] S. Goldwasser, Y. T. Kalai, C. Peikert, and V. Vaikuntanathan. Robustness of the learning with errors assumption. In *ICS*, pages 230–240. 2010.

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

[GM82] S. Goldwasser and S. Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. Preliminary version in STOC 1982.

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

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

[Gol01] O. Goldreich. *Foundations of Cryptography*, volume I. Cambridge University Press, 2001.

[GPSW06] V. Goyal, O. Pandey, A. Sahai, and B. Waters. Attribute-based encryption for fine-grained access control of encrypted data. In *CCS*, pages 89–98. 2006.

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

[GSW13] C. Gentry, A. Sahai, and B. Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In *CRYPTO*, pages 75–92. 2013.

[GVW13] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Attribute-based encryption for circuits. In *STOC*, pages 545–554. 2013.

[GVW15a] S. Gorbunov, V. Vaikuntanathan, and H. Wee. Predicate encryption for circuits from LWE. In *CRYPTO*, pages 503–523. 2015.

[GVW15b] S. Gorbunov, V. Vaikuntanathan, and D. Wichs. Leveled fully homomorphic signatures from standard lattices. In *STOC*, pages 469–477. 2015.

[HHGP+03] J. Hoffstein, N. Howgrave-Graham, J. Pipher, J. H. Silverman, and W. Whyte. NTRUSIGN: Digital signatures using the NTRU lattice. In *CT-RSA*, pages 122–140. 2003.

[HILL99] J. Håstad, R. Impagliazzo, L. A. Levin, and M. Luby. A pseudorandom generator from any one-way function. *SIAM J. Comput.*, 28(4):1364–1396, 1999.

[HJ15] Y. Hu and H. Jia. Cryptanalysis of GGH map. Cryptology ePrint Archive, Report 2015/301, 2015. `http://eprint.iacr.org/`.

[HPS98] J. Hoffstein, J. Pipher, and J. H. Silverman. NTRU: A ring-based public key cryptosystem. In *ANTS*, pages 267–288. 1998.

[HPS01] J. Hoffstein, J. Pipher, and J. H. Silverman. NSS: an NTRU lattice-based signature scheme. In *EUROCRYPT*, pages 211–228. 2001.

[HR07] I. Haviv and O. Regev. Tensor-based hardness of the shortest vector problem to within almost polynomial factors. In *STOC*, pages 469–477. 2007.

[IN96] R. Impagliazzo and M. Naor. Efficient cryptographic schemes provably as secure as subset sum. *J. Cryptology*, 9(4):199–216, 1996.

[Kan83] R. Kannan. Improved algorithms for integer programming and related lattice problems. In *STOC*, pages 193–206. 1983.

[Kho03] S. Khot. Hardness of approximating the shortest vector problem in lattices. *J. ACM*, 52(5):789–808, 2005. Preliminary version in FOCS 2003.

[KL14] J. Katz and Y. Lindell. *Introduction to Modern Cryptography*. Chapman and Hall/CRC Press, second edition, November 2014.

[Kle00] P. N. Klein. Finding the closest lattice vector when it's unusually close. In *SODA*, pages 937–941. 2000.

[KPTZ13] A. Kiayias, S. Papadopoulos, N. Triandopoulos, and T. Zacharias. Delegatable pseudorandom functions and applications. In *CCS*, pages 669–684. 2013.

[KTX08] A. Kawachi, K. Tanaka, and K. Xagawa. Concurrently secure identification schemes based on the worst-case hardness of lattice problems. In *ASIACRYPT*, pages 372–389. 2008.

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

[LM06] V. Lyubashevsky and D. Micciancio. Generalized compact knapsacks are collision resistant. In *ICALP (2)*, pages 144–155. 2006.

[LM08] V. Lyubashevsky and D. Micciancio. Asymptotically efficient lattice-based digital signatures. In *TCC*, pages 37–54. 2008.

[LM09] V. Lyubashevsky and D. Micciancio. On bounded distance decoding, unique shortest vectors, and the minimum distance problem. In *CRYPTO*, pages 577–594. 2009.

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

[LMvdP13] T. Laarhoven, M. Mosca, and J. van de Pol. Finding shortest lattice vectors faster using quantum search. *Designs, Codes and Cryptography*, 2015. To appear; preliminary version in PQCrypto 2013.

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

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

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

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

[LPS10] V. Lyubashevsky, A. Palacio, and G. Segev. Public-key cryptographic primitives provably as secure as subset sum. In *TCC*, pages 382–400. 2010.

[LPSS14] S. Ling, D. H. Phan, D. Stehlé, and R. Steinfeld. Hardness of k-LWE and applications in traitor tracing. In *CRYPTO*, pages 315–334. 2014.

[LS15] A. Langlois and D. Stehlé. Worst-case to average-case reductions for module lattices. *Designs, Codes and Cryptography*, 75(3):565–599, 2015.

[LTV12] A. López-Alt, E. Tromer, and V. Vaikuntanathan. On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption. In *STOC*, pages 1219–1234. 2012.

[LW15] V. Lyubashevsky and D. Wichs. Simple lattice trapdoor sampling from a broad class of distributions. In *PKC*, pages 716–730. 2015.

[Lyu08] V. Lyubashevsky. Lattice-based identification schemes secure under active attacks. In *Public Key Cryptography*, pages 162–179. 2008.

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

[Lyu12] V. Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, pages 738–755. 2012.

[McE78] R. J. McEliece. A public-key cryptosystem based on algebraic coding theory. DSN Progress Report 42-44, Jet Propulsion Laboratory, 1978.

[MG02] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, 2002.

[Mic98] D. Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM J. Comput.*, 30(6):2008–2035, 2000. Preliminary version in FOCS 1998.

[Mic01] D. Micciancio. Improving lattice based cryptosystems using the Hermite normal form. In *CaLC*, pages 126–145. 2001.

[Mic02] D. Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, 16(4):365–411, 2007. Preliminary version in FOCS 2002.

[Mic07] D. Micciancio. *The LLL Algorithm: Survey and Applications*, chapter Cryptographic functions from worst-case complexity assumptions, pages 427–452. Information Security and Cryptography. Springer, December 2009. Preliminary version in Proceedings of LLL25, 2007.

[Mic10] D. Micciancio. Duality in lattice cryptography. In *Public Key Cryptography*. 2010. Invited talk.

[Mic14] D. Micciancio. Lecture notes on lattice algorithms and applications, 2014. Available at `http://cseweb.ucsd.edu/~daniele/classes.html`, last accessed 17 Oct 2014.

[MM11] D. Micciancio and P. Mol. Pseudorandom knapsacks and the sample complexity of LWE search-to-decision reductions. In *CRYPTO*, pages 465–484. 2011.

[MP12] D. Micciancio and C. Peikert. Trapdoors for lattices: Simpler, tighter, faster, smaller. In *EUROCRYPT*, pages 700–718. 2012.

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

[MPSW09] T. Malkin, C. Peikert, R. A. Servedio, and A. Wan. Learning an overcomplete basis: Analysis of lattice-based signatures with perturbations, 2009. Manuscript.

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

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

[MV10] D. Micciancio and P. Voulgaris. A deterministic single exponential time algorithm for most lattice problems based on Voronoi cell computations. In *STOC*, pages 351–358. 2010.

[Ngu99] P. Q. Nguyen. Cryptanalysis of the Goldreich-Goldwasser-Halevi cryptosystem from Crypto '97. In *CRYPTO*, pages 288–304. 1999.

[NPR99] M. Naor, B. Pinkas, and O. Reingold. Distributed pseudo-random functions and KDCs. In *EUROCRYPT*, pages 327–346. 1999.

[NR95] M. Naor and O. Reingold. Synthesizers and their application to the parallel construction of pseudo-random functions. *J. Comput. Syst. Sci.*, 58(2):336–375, 1999. Preliminary version in FOCS 1995.

[NR97] M. Naor and O. Reingold. Number-theoretic constructions of efficient pseudo-random functions. *J. ACM*, 51(2):231–262, 2004. Preliminary version in FOCS 1997.

[NR06] P. Q. Nguyen and O. Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. *J. Cryptology*, 22(2):139–160, 2009. Preliminary version in Eurocrypt 2006.

[NRR00] M. Naor, O. Reingold, and A. Rosen. Pseudorandom functions and factoring. *SIAM J. Comput.*, 31(5):1383–1404, 2002. Preliminary version in STOC 2000.

[NS98] P. Q. Nguyen and J. Stern. Cryptanalysis of the Ajtai-Dwork cryptosystem. In *CRYPTO*, pages 223–242. 1998.

[NS01] P. Q. Nguyen and J. Stern. The two faces of lattices in cryptology. In *CaLC*, pages 146–180. 2001.

[OPW11] A. O'Neill, C. Peikert, and B. Waters. Bi-deniable public-key encryption. In *CRYPTO*, pages 525–542. 2011.

[Pai99] P. Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *EUROCRYPT '99*, pages 223–238. 1999.

[Pei07] C. Peikert. Limits on the hardness of lattice problems in ℓ_p norms. *Computational Complexity*, 17(2):300–351, May 2008. Preliminary version in CCC 2007.

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

[Pei10] C. Peikert. An efficient and parallel Gaussian sampler for lattices. In *CRYPTO*, pages 80–97. 2010.

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

[PR06] C. Peikert and A. Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *TCC*, pages 145–166. 2006.

[PR07] C. Peikert and A. Rosen. Lattices that admit logarithmic worst-case to average-case connection factors. In *STOC*, pages 478–487. 2007.

[PV08] C. Peikert and V. Vaikuntanathan. Noninteractive statistical zero-knowledge proofs for lattice problems. In *CRYPTO*, pages 536–553. 2008.

[PVW08] C. Peikert, V. Vaikuntanathan, and B. Waters. A framework for efficient and composable oblivious transfer. In *CRYPTO*, pages 554–571. 2008.

[PW08] C. Peikert and B. Waters. Lossy trapdoor functions and their applications. *SIAM J. Comput.*, 40(6):1803–1844, 2011. Preliminary version in STOC 2008.

[Rab79] M. O. Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical Report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

[RAD78] R. L. Rivest, L. Adleman, and M. L. Dertouzos. On data banks and privacy homomorphisms. *Foundations of secure computation*, 4(11):169–180, 1978.

[Reg03] O. Regev. New lattice-based cryptographic constructions. *J. ACM*, 51(6):899–942, 2004. Preliminary version in STOC 2003.

[Reg04] O. Regev. Lecture notes on lattices in computer science, 2004. Available at http://www.cs.tau.ac.il/~odedr/teaching/lattices_fall_2004/index.html, last accessed 28 Feb 2008.

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

[Reg10] O. Regev. The learning with errors problem (invited survey). In *IEEE Conference on Computational Complexity*, pages 191–204. 2010.

[RSA78] R. L. Rivest, A. Shamir, and L. M. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Commun. ACM*, 21(2):120–126, 1978.

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

[Sch89] C. Schnorr. Efficient signature generation by smart cards. *J. Cryptology*, 4(3):161–174, 1991. Preliminary version in CRYPTO 1989.

[Sho97] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26(5):1484–1509, 1997.

[SS11] D. Stehlé and R. Steinfeld. Making NTRU as secure as worst-case problems over ideal lattices. In *EUROCRYPT*, pages 27–47. 2011.

[SSTX09] D. Stehlé, R. Steinfeld, K. Tanaka, and K. Xagawa. Efficient public key encryption based on ideal lattices. In *ASIACRYPT*, pages 617–635. 2009.

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

[SW05] A. Sahai and B. Waters. Fuzzy identity-based encryption. In *EUROCRYPT*, pages 457–473. 2005.

[SW14] A. Sahai and B. Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *STOC*, pages 475–484. 2014.

[Vai11] V. Vaikuntanathan. Computing blindfolded: New developments in fully homomorphic encryption. In *FOCS*, pages 5–16. 2011.

[vDGHV10] M. van Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan. Fully homomorphic encryption over the integers. In *EUROCRYPT*, pages 24–43. 2010.

[Ver12] R. Vershynin. *Compressed Sensing, Theory and Applications*, chapter 5, pages 210–268. Cambridge University Press, 2012. Available at http://www-personal.umich.edu/~romanv/papers/non-asymptotic-rmt-plain.pdf.

[Wan10] A. Wan. *Learning, Cryptography and the Average Case*. Ph.D. thesis, Columbia University, 2010.

[Wat09] B. Waters. Dual system encryption: Realizing fully secure IBE and HIBE under simple assumptions. In *CRYPTO*, pages 619–636. 2009.

[Xag13] K. Xagawa. Improved (hierarchical) inner-product encryption from lattices. In *PKC*, pages 235–252. 2013.