Idk what my thesis title is

Bachelor's Project Mathematics

March 2023

Student: K. J. Pudowski

First supervisor: <add titles> Killicer

Second assessor: <add titles> Seri

Contents

1 Introduction **2**
1.1 Actual introduction here with some funny name 2
1.2 Motivation 2
1.3 Main results 3
1.4 Outline of the paper 3

2 Background **3**
2.1 Notation 3
2.2 Lattices 3
2.3 Algebraic Number Theory 6
2.4 Complexity Theory and hard problems 8

3 Homomorphic Encryption **8**
3.1 Somewhat Homomorphic Encryption 9
3.2 Fully Homomorphic Encryption 11

4 Lattice based cryptography **11**
4.1 The GGH public key cryptosystem 12
4.2 Learning With Errors 13
4.2.1 Background on Lattices II 15
4.2.2 Pseudorandomness of LWE 19
4.2.3 LWE cryptosystem 20
4.3 Ring-LWE 23
4.3.1 Background on Lattices II 24
4.3.2 Hardness of RLWE 24
4.3.3 Pseudorandomness of Ring-LWE 24
4.4 Fully Homomorphic Encryption Using Ideal Lattices 24

5 Conclusions **25**

Bibliography **28**

1 Introduction

1.1 Actual introduction here with some funny name

1.2 Motivation

include the table from page 16 of **bernstein** of systems broken by quantum computers quote from **bernstein**: "As it turns out, number theoretic problems are also the main place where quantum computers have been shown to have exponential speedups.

Examples of such problems include factoring and discrete log [38], Pell's equation [18], and computing the unit group and class group of a number field [17, 37]. The existence of these algorithms implies that a quantum computer could break RSA, Diffie-Hellman and elliptic curve cryptography, which are currently used".
On the 18^{th} of November 2022, a document was issued on migrating to post-quantum cryptography.

1.3 Main results

I imagine this will be filled last.

1.4 Outline of the paper

2 Background

2.1 Notation

Most of the notation is "standard" in the field of number theory and cryptography. Nonetheless, to avoid any misunderstandings and simplify some statements, we will present the notation used throughout the text. Anything that is not mentioned here shall be defined "on the go" with definitions and such.

Any scalars $a, t, \beta, \dots$ are represented by non-bold, latin or greek letters.
Bold symbols $\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{s}, \dots$ will denote vectors.
Algorithms will be represented by lmodern font names such as Encrypt or Evaluate.
Traditionally, the symbols $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ shall represent the sets of integer, rational, real and complex numbers, respectivelly.
Additionally, for any real $m \geq 0$, $\lfloor m \rfloor$ shall denote greatest integer not exceeding m, $\lfloor m \rceil = \lfloor m + \frac{1}{2} \rfloor$ and $[m]$ is the set $\{1, 2, \dots, \lfloor m \rfloor\}$.

2.2 Lattices

Basic Definitions

We define a *lattice* as a discrete additive subgroup of $\mathbb{R}^n$. Once we fix a basis $\boldsymbol{B} = (\boldsymbol{b}_1, \dots, \boldsymbol{b}_n) \in \mathbb{R}^n$ we can then describe the lattice as

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

There are many bases for a lattice (actually, for $n \geq 2$i, there are infinitely many as can be proven using a diagonalization argument), some "better" than others. This will be the foundation for some of the problems like `SVP` or `CVP`.

Example 2.1. The simplest example of a lattice is the $\mathbb{Z}^n$ itself. Taking the standard basis $\boldsymbol{B}_1 = (\boldsymbol{e}_1, \ldots, \boldsymbol{e}_n)$ we obtain

$$\mathcal{L}(\boldsymbol{B}_1) = \Big\{\sum_i z_i \boldsymbol{e}_i : z_i \in \mathbb{Z}\Big\} = \mathbb{Z}^n.$$

More generally, Λ is a lattice of rank m in $\mathbb{R}^n$ if it is a rank m free abelian group. Recall that we call a group *free abelian group* of rank m if it can be written as $\Lambda = \mathbb{Z}\beta_1 \oplus \cdots \oplus \mathbb{Z}\beta_m$ with $\beta_1, \ldots, \beta_m$ linearly independent over $\mathbb{R}$ where $\oplus$ represents the direct sum. In this paper we will only consider lattices of full rank n.

Remark 2.2. We can also view the vectors $\boldsymbol{b}_i$ as the columns of the matrix $\boldsymbol{B} \in \mathbb{R}^n \times \mathbb{R}^n$ in which case, our definition becomes:

$$\Lambda = \mathcal{L}(B) = \{\boldsymbol{B}\boldsymbol{z} : \boldsymbol{z} \in \mathbb{Z}^n\}.$$

Reciprocally, any matrix $\boldsymbol{B} \in GL_n(\mathbb{R})$ spans a lattice: the set of all integer linear combinations of its rows.

Example 2.3.

1. $\mathcal{L} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$ in which case $\boldsymbol{b}_1 = \left(\begin{smallmatrix} 1 \\ 0 \end{smallmatrix}\right)$ and $\boldsymbol{b}_2 = \left(\begin{smallmatrix} 0 \\ 1 \end{smallmatrix}\right)$
2. $\mathcal{L} = \{(z_1, z_2) : z_1 + z_2 \text{ is even}\}$
3. $\mathcal{L} = \begin{pmatrix} 13 & 21 \\ 21 & 34 \end{pmatrix}$

As noted before, the basis of a lattice is not unique. There is one that is particularly interesting to us, namely, the *Hermite Normal Form* (HNF). A basis $\boldsymbol{B}$ is in HNF if it is upper triangular (or lower triangular - does not matter as long as one is consistent), all elements on the diagonal are strictly positive and any other element $\boldsymbol{b}_{i,j}$ satisfies $0 \leq \boldsymbol{b}_{i,j} < \boldsymbol{b}_{i,i}$.

Fundamental Domain

Definition 2.1 (Fundamental Domain)**.** Let $\mathcal{L}$ be a lattice of dimension n and let $(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n)$ be a basis for $\mathcal{L}$. The *fundamental domain* (or *fundamental parallelepiped*) for $\mathcal{L}$ corresponding to this basis is the set

$$\mathcal{F}(\boldsymbol{b}_1, \ldots, \boldsymbol{b}_n) = \{t_1\boldsymbol{b}_1 + \cdots + t_n\boldsymbol{b}_n : 0 \leq t_i < 1\}.$$

We define the *volume* of $\mathcal{F}(\boldsymbol{B})$ as the volume of the corresponding parallelepiped in $\mathbb{R}^n$. The *volume* - closely connected to the determinant - plays a very important role in our study which will become evident in later chapters. One of the advantages, of defining the fundamental domain, is that we can formalize the notion of area (or the determinant) of any given lattice. Recall that a lattice is just a countable collection of points and therefore has no volume by itself. This, however, is resolved by introducing the following.

Definition 2.2. Let $\mathcal{L}$ be a lattice of dimension n and let $\mathcal{F}(\boldsymbol{B})$ be a fundamental domain for $\mathcal{L}$ over some basis $\boldsymbol{B}$. We define the *determinant* of that lattice as

$$\det(\mathcal{L}) = \mathrm{Vol}(\mathcal{F}(\boldsymbol{B})) = |\det(\boldsymbol{B})|$$

The next two propositions are *de facto* foundation for lattice based cryptography. The first one states that the $\det(\mathcal{L})$ does not depend on the choice of the basis for that lattice. The second, that our whole ambient space $\mathbb{R}^n$ can be described using only vectors from the lattice and the fundamental domain. We will only give an outline of the proofs for the sake of keeping this section compact. Full proofs, however, can be found in **book**, chapter 6.4.

Proposition 2.3. *The* $\det(\mathcal{L})$ *of an* n*-dimensional lattice is invariant under the choice of the basis.*

Outline of the proof. Let $\boldsymbol{B}_1, \boldsymbol{B}_2$ be two bases for a lattice $\mathcal{L}$. The crucial part of the proof is to note that any two bases are related by some unimodular matrix U (i.e. a matrix with the determinant of ± 1) s.t. $\boldsymbol{B}_1 = U\boldsymbol{B}_2$. It now easily follows to compute $|\det(\boldsymbol{B}_1)| = \det(\mathcal{L}) = |\det(U \cdot \boldsymbol{B}_2)| = |\det(U)| \cdot |\det(\boldsymbol{B}_2)| = |\det(\boldsymbol{B}_2)|$ □

From now on we will write $\mathcal{F}$ to denote the fundamental domain of the lattice without specifying the basis.

Proposition 2.4. *Let* $\mathcal{L} \subset \mathbb{R}^n$ *be a lattice of dimension* n *and let* $\mathcal{F}$ *be a fundamental domain for* $\mathcal{L}$*. Then every vector* $\boldsymbol{v} \in \mathbb{R}^n$ *can be written in the form*

$$\boldsymbol{v} = \boldsymbol{f} + \boldsymbol{t}$$

for $\boldsymbol{f} \in \mathcal{F}$ *and* $\boldsymbol{t} \in \mathcal{L}$ *both unique and associated to the original* $\boldsymbol{v}$*.*

Equivalently, the space $\mathbb{R}^n$ is spanned exactly (without overlap) by shifting the fundamental domain by the vectors from our lattice.

$$\mathbb{R}^n = \bigcup_{\boldsymbol{t} \in \mathcal{L}} \{\boldsymbol{f} : \boldsymbol{f} \in \mathcal{F}\}$$

Remark 2.4. Sometimes the *fundamental domain* is refered to as a parallelepiped or parallelotope and denoted by caligraphic $\mathcal{P}$. If we take a matrix $\boldsymbol{B}$ to represent our lattice $\mathcal{L}$, then $\mathcal{P}_{1/2}(\boldsymbol{B}) = \{\boldsymbol{x}\boldsymbol{B}, \boldsymbol{x} \in [-1/2, 1/2]^n\}$ can also represent the (shifted by a half) fundamental domain of $\mathcal{L}$ (like for example in **gentry**).

We will now present two results that give us an upper bound on the length of the shortest vector in a lattice. This will later on be useful to determine the security and/or correctness of our schemes. These theorems are due to Hermite (1822 - 1901) and Minkowski (1864 - 1909).

Theorem 2.5 (Hermite's Theorem)**.** *Every lattice $\mathcal{L}$ of dimension n contains a nonzero vector $v \in \mathcal{L}$ satisfying*

$$\|v\| \leq \sqrt{n} \det(\mathcal{L})^{\frac{1}{n}}.$$

[Krzys: add minkowski's theorem]
[Krzys: add shortest vector $\lambda_1(n)$]
[Krzys: add dual]
[Krzys: add smoothing parameter]

2.3 Algebraic Number Theory

Algebraic number theory is the study of *number fields*, *rings of integers* and *finite fields*. In this section we will provide all the necessary background needed to understand and verify the results presented in the cryptographic schemes later in the text. Most results will be stated without proof however all of them can be found in the book **Number Fields** by **Daniel A. Marcus** algebra after which this sections is modelled.

Number Fields

A *number field* is defined as a subfield of $\overline{\mathbb{Q}}$ having finite dimension as a vector space over the rationals $\mathbb{Q}$. The *degree* of a number field K is defined as the dimension of K over $\mathbb{Q}$. There exists a monic irreducible polynomial[1] $f \in \mathbb{Q}[x]$ such that $K \cong \mathbb{Q}[x]/\langle f \rangle$. In fact, every monic and irreducible polynomial in $\mathbb{Q}[x]$ defines a number field via such isomorphism.

Definition 2.6 (Algebraic integer)**.** An element $\alpha \in \mathbb{C}$ is an *algebraic integer* iff it is a root of some monic polynomial in $\mathbb{Z}[x]$.

In fact, the set of *algebraic integers* forms a ring.

Definition 2.7 (Ring of Integers)**.** We define the *ring of integers* (sometimes also called *maximal order*) $\mathcal{O}_K$ of a number field K as the intersection:

$$\mathcal{O}_K = K \cap \overline{\mathbb{Z}} = \{x \in K : \ x \text{ is an algebraic integer}\}.$$

Example 2.5. The field $K = \mathbb{Q}$ is a number field of degree 1. Its ring of integers is, as one can guess, the ordinary integers $\mathbb{Z}$.

Example 2.6. The ring of Gaussian integers $\mathbb{Z}[\sqrt{-1}]$ is the ring of integers of $K = \mathbb{Q}(\sqrt{-1}) = \{a + b\sqrt{-1} : a, b \in \mathbb{Q}\}$ which has degree 2 since $x^2 + 1$ is the minimal (and irreducible) polynomial of $\sqrt{-1}$ over $\mathbb{Q}$.

[1] Recall that we call polynomial monic if its leading coefficient is 1. It is an irreducible polynomial if it is irreducible as an element of the polynomial ring $\mathbb{Q}[x]$.

As another example, in generality, we can make a following statement about the ring of integers of a quadratic extension of rationals (real quadratic field).

Lemma 2.8. *Let $d \in \mathbb{Z}$ be a square-free integer. For the field $K = \mathbb{Q}(\sqrt{d})$, its ring of integers is*

$$\mathcal{O}_K = \begin{cases} \mathbb{Z}[\sqrt{d}] & \text{if } d \equiv 2, 3 \mod 4, \\ \mathbb{Z}[(1+\sqrt{d})/2] & \text{otherwise.} \end{cases}$$

Proof. Take $d \equiv 1 \mod 4$ square-free. □

Example 2.7. For $K = \mathbb{Q}(\sqrt{5})$ the ring of integers is $\mathcal{O}_K = \mathbb{Z}[(1+\sqrt{5})/2]$.

Cyclotomic fields

Definition 2.9 (Roots of unity)**.** Given a field K and a positive integer n, an element $\zeta \in K$ is called *primitive n-th root of unity* if ζ has order n in the multiplicative group $K^\times$. (In other words, $\zeta^n = 1$ and $\zeta^m \neq 1$ for $1 \leq m < n$).

The minimal polynomial Φ_n of ζ over $\mathbb{Q}$ is called the n-th cyclotomic polynomial. It can be shown that every such ζ is equal to precisely $e^{2\pi\sqrt{-1}/n}$ and $\zeta^i = \zeta^j$ iff $i = j$.

Remark 2.8. What is worth noting here, is that for a number field $\mathbb{Q}(\alpha)$ for some $\alpha \in \mathbb{C}$, the ring of integers is not necessarily the $\mathbb{Z}[\alpha]$. Instead, $\mathbb{Z}[\alpha]$ is what's called an *order* in $\mathcal{O}_K$. We will not consider them in general here because they are not relevant for our study. However, one very useful feature of cyclotomic fields is that their ring of integers is actually just $\mathbb{Z}[\zeta]$. That is - $\mathcal{O}_K = \mathbb{Z}[\zeta]$ for $K = \mathbb{Q}(\zeta)$ and ζ is some n-th root of unity. This greatly simplifies the approach in proving some of the results later in this paper. In general, a field $K = \mathbb{Q}(\alpha)$ such that $\mathcal{O}_K = \mathbb{Z}[\alpha]$ is called a *monogenic* field. For more details on orders, look at for example Chapter 5 of **stein**. [Krzys: I need more good resources]

Proposition 2.10. *The ring of integers of a*

Embeddings in $\mathbb{C}$

Let $K = \mathbb{Q}(\alpha)$ be a number field of degree n for some α. Then there are exactly n canonical embeddings (injective ring homomorphisms) of K in $\mathbb{C}$. These are easily described by observing that α can be sent to any one of its n conjugates over $\mathbb{Q}$. Each conjugate β determines a unique embedding ($\sigma_i : K \to \mathbb{C}$ and every embedding must arise in this way since α must be sent to one of its conjugates).

Example 2.9. The quadratic field $\mathbb{Q}[\sqrt{d}]$, d squarefree, has two embeddings in $\mathbb{C}$: The identity mapping, and also the one which sends $a + b\sqrt{d}$ to $a - b\sqrt{d}$ (a, $b \in \mathbb{Q}$), since $\sqrt{d}$ and $-\sqrt{d}$ are the two conjugates of $\sqrt{d}$. The n-th cyclotomic field has $\varphi(n)$ embeddings in $\mathbb{C}$, the $\varphi(n)$ automorphisms where $\sigma_i(\zeta) = \zeta^i$.

Consider now

[Pinar: what do we want to consider as a cyclotomic poly? is $n = 2^a$? is n a prime? and why?]

[Pinar: Maximal orders (ring of integers) are dedekind domains, embedding of $\mathbb{Q}(\alpha)$ to $\mathbb{C}$ hence embedding of the ideals. Properties of ideals in dedekind domains, operations, unique factorization and so on. all the necessary info.]

2.4 Complexity Theory and hard problems

[Pinar: will wait] mention "efficient", "negligible", "hard" and define reductions. show some problems like svp, cvp, bdd In this section we will briefly introduce what it means for a problem to be considered *hard* and provide couple of examples We define a `negligible` amount in n as an amount that is asymptotically smaller than n^{-c} for any constant $c > 0$. More precisely,

Definition 2.11. $f(n)$ is a `negligible` function in n if $\lim_{n\to\infty} n^{-c} f(n) = 0$ for any $c > 0$.

The best know examples `FACTORIZE`

Quantum computations

I'm not sure if that is supposed to be in a section about preliminaries but I also don't want to include it in the introduction coz its a bit long

3 Homomorphic Encryption

Fully Homomorphic Encryption (FHE) has been referred as the "holy grail" of modern cryptography as it was one of the most sought goals for the past couple of decades. First formally introduced by Rivest, Adleman and Dertouzos in **primal** (at the time called "privacy homomorphism"), shortly after the discovery of public key cryptography, it has been an open and elusive problem. Only "recently", in 2009, Craig Gentry proposed first FHE in his PhD thesis **gentry_phd**. Since then, there has been a lot of development in the area like for example [Krzys: TODO: finish developments of fhe].

Simply stated, in homomorphic encryption we want our data to be secure but we also want to perform calculations on it. This is useful when you need a third party (e.g. someone with more computational power) to perform operations on your data while still retaining privacy. Alice can store her data somewhere on external server (the cloud) and ask to perform computations on it. For example query searches without the engine knowing what is actually being searched for.

In other words, we would like our encryption scheme to satisfy the following. Say the ciphertexts c_i's decrypt to messages m_i's. Then we want

$$\mathsf{Decrypt}_{\mathcal{E}}(c_1 + c_2) = m_1 + m_2, \qquad \mathsf{Decrypt}_{\mathcal{E}}(c_1 * c_2) = m_1 * m_2$$

Equivalently, we want $\mathsf{Decrypt}$ to be a ring homomorphism with respect to both addition and multiplication (of the given ring). $\mathcal{E}$ is *fully homomorphic* means that whenever f is a composition of **arbitrily many** additions and multiplications, then $\mathsf{Decrypt}_{\mathcal{E}}(f(c_1, \ldots, c_n)) = f(m_1, \ldots, m_n)$[2] which is also refered to as the *correctness* of the scheme.

Remark 3.1. Typically, an encryption scheme $\mathcal{E}$ is a tuple $\mathsf{KeyGen}_{\mathcal{E}}$, $\mathsf{Encrypt}_{\mathcal{E}}$ and $\mathsf{Decrypt}_{\mathcal{E}}$ (representing the key-generation, encryption and decryption respectively), all of which we require to be *efficient* - i.e. run in time poly(λ) - polynomial in the security parameter λ that represents the bit-length of the keys (see for example **katz** or **book** for more details). A homomorphic encryption scheme has a fourth algorithm - $\mathsf{Evaluate}_{\mathcal{E}}$ which we associate with some set of *permitted functions*. In our case this will simply be $\mathsf{Add}_{\mathcal{E}}$ and $\mathsf{Mult}_{\mathcal{E}}$ which we will introduce in further sections. Adopting the notation from **easy_fhe** we will denote by $\mathcal{F}_{\mathcal{E}}$, the generalized set of such functions.

3.1 Somewhat Homomorphic Encryption

Before we introduce the solution on to how to construct such FHE presented by Gentry, we will start with something slightly simpler, introduced in **int_scheme** by van Dijk et al. Their scheme works over the integers rather than lattices but relies on a similar assumption. Namely, that finding the greatest common divisor of many "noisy" multiples of a number is computationally difficult. We will come back to this problem later. To keep the exposition compact, we will avoid specifying most parameter choices.

Symmetric Key Scheme

We begin with the symmetric key scheme. We take our message to be a bit $m \in \{0, 1\}$. The private key is an odd integer chosen from some interval $p \in [2^{\eta-1}, 2^{\eta})$. To encrypt our message m, we choose integers q and r at random (from some other interval and such that the magnitude of $2r$ is smaller than $p/2$). We obtain the ciphertext c by computing:

$$c = pq + 2r + m. \tag{3.1}$$

[2]There are two more technical requitements, namely *compactness of the ciphertexts* and *efficiency* but we will not consider them in this paper.

If we now want to decrypt our message, simply compute $(c \mod p) \mod 2$.
Let's say we have two messages c_1 and c_2. Then we can compute:

$$c_1 + c_2 = m_1 + m_2 + 2(r_1 + r_2) + p(q_1 + q_2),$$

$$c_1 * c_2 = m_1 * m_2 + 2(m_1 r_2 + m_2 r_1 + 2 r_1 r_2) + p(m_1 q_2 + m_2 q_1 + 2(r_1 q_2 + r_2 q_1) + p q_1 q_2)$$

where we can see that the noise grows with each operation and the message becomes impossible to decrypt after we do too many of them. If we can assure that $2(m_1 r_2 + m_2 r_1 + 2 r_1 r_2)$ is small enough - i.e. smaller than p[3] - then we can assure that $\mathsf{Decrypt}(c_1 * c_2)$ evaluates correctly to the starting $m_1 * m_2$. Notice that $\mathsf{Decrypt}$ removes all the noise. This will be useful later for bootstrapping.

This simple encryption scheme is thus somewhat homomorphic as per definition by Gentry in **gentry_phd** – namely, it can be used to evaluate low-degree polynomials over encrypted data. Further on in the Section 6 of **int_scheme**, van Dijk et al. use the techniques (called bootstrapping and squashing) to lift it to a Fully Homomorphic Scheme.

Public Key Scheme

The public key scheme is build very similarly. The private key p stays the same. For the public key, sample $x_i = pq_i + 2r_i$ for $i = 0, 1, \ldots, t$ where the q_i and r_i stay as before. The x_i may be viewed as encryption of 0 under the symmetric key scheme. The x_i are now taken s.t. x_0 is the largest, odd and $x_0 \mod p$ is even.
To now encrypt a message $m \in \{0, 1\}$, chose a random subset $S \subseteq \{1, 2, \ldots, t\}$ and a random integer r, and output

$$c = (m + 2r + 2\sum_{i \in S} x_i) \mod x_0. \tag{3.2}$$

To decrypt, we again output $m = (c \mod p) \mod 2$.

The security of this preliminary SH scheme relies on the *Approximate GCD Problem*[4]. In the simplest case, Euclid has shown us, that given two integers c_1 and c_2, it is easy to compute their gcd. However, suppose now that $c_1 = p \cdot q_1 + r_1$ and $c_2 = p \cdot q_2 + r_2$ are "near" multiples of p, where r_1 and r_2 is some small noise sampled at random. This turns out to be much more difficult. In fact, if we pick our values appropriately (see **easy_fhe** §3.4 and **int_scheme** §3 for details) we do not know any efficient (running in polynomial time) algorithm even if we are given arbitrarily many samples $c_i = r_i + p \cdot q_i$.

[3] When $2r > p$ then it might be the case that $2r = 1 \mod p$ and so $pq + 2r + m \mod p = 1 + m \neq m$.

[4] Later in **revisited**, a reduction was constructed to LWE. This means, that under few more assumptions, this problem (and by extension any scheme based on it) is as secure as one based on LWE.

However, this comfortable security comes at great cost because, as shown in **int_scheme**, the parameters chosen to assure the secrecy, yield a scheme that has complexity of $\tilde{O}(\lambda^{10})$ where λ is our security parameter (the greater it is the more secure message). As a small example, consider $\lambda = 10$ as the (small) key size. To now encrypt a single(!) bit, it will take approximately 10^{10} operations. On a modern laptop this would take a little less than 5 seconds. To send the message 'hello', we need to use 5 letters $*$ 16-bits per letter $= 5 * 16 * 5 = 650$ seconds which is almost 11 minutes! As one can imagine, this is completely impractical for most applications.

3.2 Fully Homomorphic Encryption

We will now present the main idea introduced in Gentry's PhD thesis **gentry_phd**. Namely, the initial "bootstrapping" result, our SHE from previous section and a technique to "squash the decryption circuit" to allow bootstrapping. At the end, we will finally be left with *Fully Homomorphic Encryption* scheme. As a basis, we will be using the SHE scheme from previous section.

Boostrapping

We are faced with a problem. Because our method relies on some error being added to the message, it builds up after we perform operations on our data. The scheme $\mathcal{E}$ can handle functions in a limited set $\mathcal{F}_{\mathcal{E}}$ until the noise becomes too large.

The basic idea behind bootstrapping is to use a homomorphic decryption circuit to "refresh" the ciphertext, so that it can be used for further homomorphic operations. The process involves encrypting the decryption circuit itself, evaluating it on the ciphertext, and then re-encrypting the result to obtain a new ciphertext that can be used for further homomorphic operations.

Remark 3.2. In cryptography, instead of general functions, one often considers a *circuit* instead.

To simplify our construction assume that our scheme $\mathcal{E}$ is "circularly secure". This means that we can encrypt the secret key using itself under the same scheme and no information is leaked to third parties. This is in practice very difficult to prove and has to be explicitly assumed.

4 Lattice based cryptography

Gentry's work was a true breakthrough. It not only presented the first, fully homomorphic encryption scheme, but also gave researchers a very powerful tool, the *bootstrapping*. From now on, all we need to construct another FHE scheme, is some suitable (one requirement would be to use a scheme based on ring rather than a group) SHE method, apply appropriate "squashing" to obtain the bootstrapping and

we are done. In the following years this is exactly what happened in academia and the industry.

This section will mostly serve as a survey of the main developments towards more efficient fully homomorphic encryption using (ideal) lattices and their security based on computational hardness of the underlying problems. We adopt chronological narrative of the sections, starting with the oldest, the GGH algorithm, progressing through works on (ring-)LWE and eventually arriving at the work of Gentry **gentry_phd** on ideal lattices and FHE. For a good survey on the lattice based cryptography, see for example **two_faces**, **book** chapter 6 or **lattice-survey**.

4.1 The GGH public key cryptosystem

We will start this section with a somewhat simpler cryptosystem that was developed by Goldreich, Goldwasser and Halevi and presented in 1997 **ggh**, called the GGH cryptosystem. This scheme, rather than using ideal lattices (i.e. lattices that are also ideals in the ring of integers), relies on general properties of lattices. Namely, the hardness of the SVP and CVP (see section 2.4).

Idea behind the scheme

The basic GGH cryptosystem, as mentioned before, is based on the problem of finding the closest vector in the lattice $\mathcal{L}$ to a given point in the ambient space $\mathbb{R}^n$. We are given two bases, call them $\boldsymbol{B}_{good}$ and $\boldsymbol{B}_{bad}$. The $\boldsymbol{B}_{bad}$ will be our public key and $\boldsymbol{B}_{good}$ the secret key. The $\boldsymbol{B}_{bad}$ consists of long and highly non-orthogonal vectors, as opposed to $\boldsymbol{B}_{good}$. Our secret message $\boldsymbol{m}$ is represented as a binary vector which we will use to form a linear combination $\boldsymbol{s} = \sum m_i \boldsymbol{v}_i^{bad} \in \mathcal{L}$ of the vectors in $\boldsymbol{B}_{bad}$. We now add some small and random[5] error $\boldsymbol{e} \in \mathbb{R}^n$ to obtain the ciphertext $\boldsymbol{c} = \boldsymbol{s} + \boldsymbol{e} = \sum m_i \boldsymbol{v}_i^{bad} + \boldsymbol{e} \in \mathbb{R}^n$ - some point that is not in the lattice, but rather, very close to a point in it.
To decrypt, we can use our good basis $\boldsymbol{B}_{good}$ to represent $\boldsymbol{c}$ and, for example Babai's algorithm[6] to find $\boldsymbol{v}$ and represent it in terms of the basis $\boldsymbol{B}_{good}$ to recover $\boldsymbol{m}$. On the other hand, any eavesdropping adversary that is trying to learn our secret, is left with some bad basis that will be of no help in solving the CVP.

[5] some small note about the "randomness" of this e

[6] Simply stated, if the vectors of the basis are sufficiently orthogonal to one another, then this algorithm solves `approxCVP`. However, if the Hadamard ratio is too small, the algorithm fails to find the closest vector - **book**.

GGH construction - concretely

KeyGen:

- Pick a basis $(\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n) \subset \mathbb{Z}^n$ such that they are reasonably orthogonal to one another - i.e. with small Hadamard ratio. We will associatie the vectors $\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_n$ as the n-by-n matrix $\boldsymbol{V}$ and let $\mathcal{L}$ be the lattice generated by these vectors. This is our good basis $\boldsymbol{B}_{good}$ - the **private key**.
- Pick an n-by-n matrix $\boldsymbol{U}$ with integer coefficients and determinant ± 1 and compute $\boldsymbol{W} = \boldsymbol{UV}$. The column vectors $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n$ of $\boldsymbol{W}$ are the bad basis $\boldsymbol{B}_{bad}$ of $\mathcal{L}$ - the **public key**[7].

Encrypt: Decrypt:

The greatest drawback of GGH is that there were no proofs of security presented along the algorithm, only heuristic assumptions. This motivated researchers to look for possible exploits beased on the choice of parameters. Indeed, this scheme turned out to be insecure for most practical choices of the security parameter only 2 years later, in **break1** and broken completely in **break2**. Nonetheless, the ideas presented there have served as a basis for many schemes that are proven to be secure, like for example LWE, and has led to a plethora of applications.

4.2 Learning With Errors

Let us now begin with what went wrong in GGH. Namely, first prove the hardness of a problem, then use it to construct a secure and efficient cryptosystem. In this section we introduce *Learning With Errors* (LWE) problem and the cryptosystem introduced by Oded Regev in **regev** (he won the 2018 Gödel Prize for this work). This very important work in the field of lattice based cryptography is, up to the date [Krzys: im not sure if this statement is true, i need to look more into it], one of the most efficient schemes with an actual proof of security. It has served as a foundation for countless subsequent works in the field. [Krzys: provide validation for this statement]

LWE problem

There are multiple equivalent definitions of this problem. We adopt the notation and approach introduced in the original paper by Regev. In this section, we will mainly focus on the parts that are the most relevant for our study of ring-LWE such as[Krzys: finish the lemmas most influenced by switch to ring].

The problem is parametrized by positive integers n, m[8] and prime q, as well as an error distibution χ over $\mathbb{Z}_q$. It is now defined as follows. We are given m equations of

[8] As will be seen later, m plays virtually no role in the problem definition and is usually omitted.

the form $(\boldsymbol{a}_i, b_i = \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i)$ and are asked to find the vector $\boldsymbol{s} \in \mathbb{Z}_q^n$. Here, $\boldsymbol{a}_i$ are chosen uniformly and independently from $\mathbb{Z}_q^n$, $b_i \in \mathbb{Z}_q$ and $\langle \cdot, \cdot \rangle$ denotes the usual dot product. The errors e_i are obtained by sampling independently from the probability distribution $\chi : \mathbb{Z}_q \to \mathbb{R}^+$ on $\mathbb{Z}_q$. We will denote the problem of recovering $\boldsymbol{s}$ from such equations, by $\mathrm{LWE}_{q,\chi}$ (learning with errors).

Example 4.1 (From the introduction to **regev_survey**). Given as input this set of equations, LWE asks us to recover the vector $\boldsymbol{s} = (s_1, s_2, s_3, s_4) \in \mathbb{Z}_{17}^4$. In this case $n = 4$, $q = 17$ and the error distribution is giving us $e_i = \{-1, 1\}$ with equal probability.

$$
\begin{aligned}
14s_1 + 15s_2 + 5s_3 + 2s_4 &\approx 8 \,(\mod 17)\\
13s_1 + 14s_2 + 14s_3 + 6s_4 &\approx 16 \,(\mod 17)\\
6s_1 + 10s_2 + 13s_3 + 1s_4 &\approx 3 \,(\mod 17)\\
10s_1 + 4s_2 + 12s_3 + 16s_4 &\approx 12 \,(\mod 17)\\
9s_1 + 5s_2 + 9s_3 + 6s_4 &\approx 9 \,(\mod 17)\\
3s_1 + 6s_2 + 4s_3 + 5s_4 &\approx 16 \,(\mod 17)\\
&\vdots\\
6s_1 + 7s_2 + 16s_3 + 2s_4 &\approx 3 \,(\mod 17)
\end{aligned}
$$

In this case, $\boldsymbol{s} = (0, 13, 9, 11)$. Note that if not for the error, the secret would be very easy to find. Given about n equations, we could recover $\boldsymbol{s}$ in an efficient way using Gaussian elimination. Inducing the error is what seems to render the problem untraceable for modern day algorithms.

The central part of **regev** revolves around proving the hardness of LWE. Specifically, that for appropriately chosen q and χ, a *quantum*[9] reduction algorithm exists that approximates worst-case lattice problems. The following is the main result presented in the paper.

Theorem 4.1 (**regev**, Theorem 1.1)**.** *Let n, q be positive integers and $\alpha \in (0, 1)$ be such that $\alpha q > 2\sqrt{n}$. If there exists an efficient algorithm that solves $\mathit{LWE}_{q,\Psi_\alpha}$, then there exists an efficient quantum algorithm that approximates the decision version of the shortest vector problem ($\mathtt{GapSVP}_\gamma$) and the shortest independent vectors problem ($\mathtt{SIVP}_\gamma$) to within $\gamma = \tilde{O}(n/\alpha)$ in the worst case on any lattice of dimension n.*

Let us unwrap this statement. As said before, we need an appropriate choice of paramenters to obtain our results and $\alpha > 2\sqrt{n}/q$ is one of those choices (and requirements). It specifies the shape of the Ψ_α distribution. This one is almost identical to the discrete Gaussian distribution over $\mathbb{Z}_q$ that is centered around 0

[9] In fact, quantum reduction is used only in the small part of the whole proof.

with standard deviation αq[10]. The theorem can be rephrased as follows. Imagine that we have an efficient algorithm that solves the $\mathrm{LWE}_{q,\bar{\Psi}_\alpha}$. Then, there exists a quantum solution to worst-case lattice problems, namely `GapSVP` and `SIVP`. Since we strongly believe that `GapSVP` and `SIVP` are difficult to solve (**svp-hard**, **reductions**, **cvp-hard**) we are left with a difficult, yet efficient way to share secrets. Oded Regev proceeds to prove this using various lemmas and results from few areas of mathematics like probability, lattice theory and quantum computing. We will now present selected proofs and outline of whole the approach.

4.2.1 Background on Lattices II

Gaussians

Definition 4.2 (Statistical distance)**.**

Problems

Definition 4.3 (Search-LWE)**.**

Definition 4.4 (Decision-LWE)**.**

Definition 4.5 (LWE distribution)**.** Let $q \geq 2$ be some integer, and let $\chi : \mathbb{Z}_q \to \mathbb{R}^+$ be some probability distribution on $\mathbb{Z}_q$. Let n be an integer and let $\boldsymbol{s} \in \mathbb{Z}_n^q$ be a vector. We define $A_{\boldsymbol{s},\chi}$ as the distribution on $\mathbb{Z}_n^q \times \mathbb{Z}_q$ obtained by choosing a vector $\boldsymbol{a} \in \mathbb{Z}_q^n$ uniformly at random, choosing $e \in \mathbb{Z}_q$ according to χ, and outputting $(\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e)$, where additions are performed in $\mathbb{Z}_q$, i.e., modulo q. In parallel, we define U as the uniform distribution on $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

Definition 4.6.

Smoothing parameter

Proof of Theorem 4.1

In this section we will focus on proving the main statement of **regev** following the steps presented in the paper. We will put an emphasis on the parts that require the most adjustment when trying to prove the same results for ring-LWE in the next section.

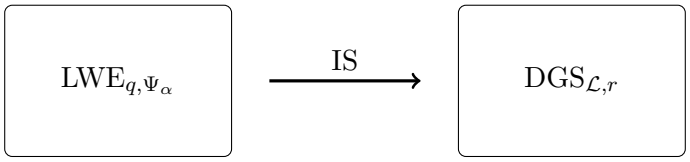
Recall that we want to prove, that being able to solve $\mathrm{LWE}_{q,\chi}$ implies that we are able to solve standard worst-case lattice problems like `SIVP`. To achieve this, we need to proceed in steps, in other words, we perform reductions from one problem to

[10] A comment from **lattice-survey**: Originally, Regev considered the continuous Gaussian and rounded the result to the nearest integer. This does not exactly yield the discrete distribution but thanks to **discr** we know how the problem can be fixed.

another. From this point on, a probabilistic algorithm which solves a given $\mathrm{LWE}_{q,\chi}$ instance, will be called an LWE-*oracle* (or, when there is no confusion, simply an oracle).

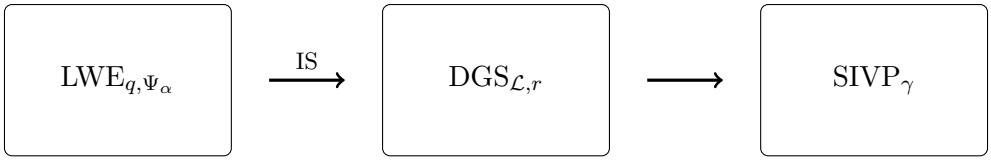
The high-level version of the proof is as follows. Let us assume that we have such an oracle that solves $\mathrm{LWE}_{q,\chi}$ on a lattice $\mathcal{L}$ just like in the assumption. The procedure is based on the so called *iterative step* (IS). It is repeatedly used to reduce our LWE problem to the problem of sampling from the discrete Gaussian distribution on $\mathcal{L}$. As it turns out, it is indeed sufficient to solve the `DGS` problem (Definition 4.6). Intuitively, if we have enough samples from the Gaussian distribution of some small radius r around our lattice, we can use it to obtain short lattice vectors. Indeed, if we call the `DGS` oracle enough times we can prove, that with very high probability, there are n linearly independent vectors among the samples returned by oracle - thus, a solution to `SIVP`.

On each of the iterations, the IS is using the oracle to produce discrete Gaussian samples of smaller and smaller radius around our desired *closest vector*. Once we have a sample with radius small enough, we can use that to solve `DGS` and use one more step which is the reduction from `DGS` to the `GapSVP` and `SIVP` as required in Theorem 4.1.



The algorithm can be divided into two basic steps, the *classical* step and the *quantum* step. First, given some polynomial number n^c (where c is some positive integer) of samples from a discrete Gaussian on $\mathcal{L}$ with some (large enough) parameter r, we us the first step to obtain a solution to a `CVP` on the dual lattice $\mathcal{L}^\vee$ to within some (smaller than the initial r) distance. We then use this solution along with the second step - the quantum algorithm - to obtain n^c samples from a dicrete Gaussian with parameter $r' < r$. Iterating these steps polynomial amount of times, we finally get n^c samples from $\mathrm{D}_{\mathcal{L},r''}$ where $r'' \ll r$. We simply pick one of those samples to solve the $\mathrm{DGS}_{\mathcal{L},r}$ and we are done. [Krzys: input a schematic as nice as the one in regev]

The full picture now stands as follows. We begin with a assumed solution to LWE and we end up with a solution to some (conjured) hard lattice problems like SIVP to within some tolerance γ.



We will now focus on the first step of the reduction from LWE to DGS as it is what is altered the most in the ring-LWE hardness proof. This, however, will follow in the next section.

The following statement (Theorem 3.1 in **regev**) is the core of the hardness results of (search) LWE. Given an oracle form LWE, there exists an algorithm that gives us samples from discrete Gaussian distribution which in turn can yield us a solution to the CVP. For example, later we will show the equivalence between search-LWE and decision-LWE (Theorem 4.12). This is an important result as it is usually much easier to construct cryptographic schemes based on some *decision* version of a problem rather than the *search*.

Theorem 4.7. *Let $\epsilon = \epsilon(n)$ be some negligible function of n. Also, let $q = q(n)$ be some integer and $\alpha = \alpha(n) \in (0,1)$ be such that $\alpha q > 2\sqrt{n}$. Assume that we have access to an oracle W that solves LWE_{q,Ψ_α} given a polynomial number of samples. Then there exists an efficient quantum algorithm for $DGS_{\sqrt{2}n\cdot\eta_\epsilon(L)/\alpha}$.*

Proof. The proof follows in a straight-forward manner by the Lemma 4.8. We sample a polynomial amount of elements from the $\mathrm{D}_{\mathcal{L},r}$ for r large enough[11]. We then apply the IS to obtain $\mathrm{D}_{\mathcal{L},r'}$ for $r' \leq r/2$. Applying this procedure enough times (turns out only about $3n$ iterations are needed). At the end of the loop, we are left with the same amount of samples but each within sufficient distance $d \leq \sqrt{2}n \cdot \eta_\epsilon(\mathcal{L})/\alpha$ and we complete the algorithm by simply outputting the first one. □

The iterative step The proof of Theorem 4.7 relies heavily on the following algorithm.

Lemma 4.8 (**regev**, 3.3 - IS)**.** *Let $\epsilon = \epsilon(n)$ be a negligible function, $\alpha = \alpha(n) \in (0,1)$ be a real number, and $q = q(n) \geq 2$ be an integer. Assume that we have access to an oracle W that solves LWE_{q,Ψ_α} given a polynomial number of samples. Then, there exists a constant $c > 0$ and an efficient quantum algorithm that, given any n-dimensional lattice L, a number $r > \sqrt{2}q\eta_\epsilon(L)$, and n^c samples from $D_{L,r}$, produces a sample from $D_{L,r\sqrt{n}/(\alpha q)}$.*

Proof. As mentioned before, the algorithm consists of two parts. The first part is presented in 4.9, where, given an oracle W and samples from $\mathrm{D}_{L,r}$, solves $\mathtt{CVP}_{L^\vee,\alpha q/(\sqrt{2}r)}$. The second part - the quantum algorithm 4.10 - when given an oracle that solves $\mathtt{CVP}_{L^\vee,\alpha q/(\sqrt{2}r)}$, yields us a sample from $\mathrm{D}_{L,r\sqrt{n}/\alpha q)}$. □

Pictographically, here are two iterations of the algorithm: [Krzys: include pictoraphic like in Regev page 8]

[11] By Claim 2.13 and Lemma 3.2 of **regev**, we can efficiently draw samples that are exponentially close to $\mathrm{D}_{\mathcal{L},r}$ if $r > 2^{2n}\lambda_n(\mathcal{L})$.

We now present heavily simplified version of the algorithm that, when given a polynomial amount of samples from the discrete Gaussian, gives us a solution to the CVP to within error of $\alpha q/(\sqrt{2}r)$ to the dual lattice $\mathcal{L}^\vee$.

Lemma 4.9 (Step 1 - classical)**.** *Let $\epsilon > 0$ be some small constant (possibly depending on n), $q \geq 2$ be an integer, and $\alpha \in (0,1)$ be a real number. Assume that we have access to an oracle W that finds $\boldsymbol{s}$ given a polynomial number of samples from $A_{\boldsymbol{s},\Psi_\alpha}$. Then, there exists an efficient algorithm that given an n-dimensional lattice $\mathcal{L}$, a number $r > \sqrt{2}q\eta_\epsilon(\mathcal{L})$, and a polynomial number of samples from $D_{\mathcal{L},r}$, solves $\mathit{CVP}_{\mathcal{L}^\vee,\alpha q/(\sqrt{2}r)}$.*

Our goal is, given a point $\boldsymbol{x}$ close enough (precisely within $\alpha q/(\sqrt{2}r)$) to $\mathcal{L}^\vee$, construct an instance of a $A_{\boldsymbol{s},\Psi_\beta}$ (for some $\beta \leq \alpha$) where the $\boldsymbol{s}$ will depend on $\boldsymbol{x}$. After polynomial amount of such samples, we can then use our oracle to recover $\boldsymbol{s}$ and consecutively x. We say an algorithm solves $\mathtt{CVP}_{\mathcal{L},d}$ if, given any point $\boldsymbol{x} \in \mathbb{R}^n$ within distance d of $\mathcal{L}$ it gives $\boldsymbol{y} \mod q \in \mathbb{Z}_q^n$, the coefficient vector of the closest vector to $\boldsymbol{x}$ reduced modulo q[12].

The following proof is a simplified version of the proof of Lemma 3.11 from **regev** where we skip few technical details. For all of them, the reader is redirected to the original.

Proof. We start with a sample vector $\boldsymbol{v} \in \mathcal{L}$ from $\mathrm{D}_{\mathcal{L},r}$ and set $\boldsymbol{a} = \mathcal{L}^{-1}\boldsymbol{v} \mod q$ - its coefficient vector (notice that we are slightly abusing the notation here using $\mathcal{L}$ as the matrix representing the lattice). We note at this point that it is in fact enough to find solution modulo q as there exists an algorithm iterating over the coefficients along with (for example) Babai's nearest plane algorithm (see **babai**) to obtain the full, unreduced solution. For the details, see **regev** Lemma 3.5. We now output

$$(\boldsymbol{a}, \langle \boldsymbol{x}, \boldsymbol{v} \rangle + e \mod q) \tag{4.1}$$

where the $e \in \mathbb{R}^n$ is chosen from a normal distribution with deviation $\alpha/(\sqrt{2}\pi)$. We claim now that our output is within negligible statistical distance of $A_{\boldsymbol{s},\Psi_\beta}$ for some $\beta \leq \alpha$ and $\boldsymbol{s} = (\mathcal{L}^\vee)^{-1}\boldsymbol{y} \mod q$.

To see this, first note that $\boldsymbol{a}$ is a uniform sample (**regev**, Claim 3.8). Let us now fix $\boldsymbol{a}$ and consider $\boldsymbol{y} = \boldsymbol{x} - \epsilon$. Then

$$\langle \boldsymbol{x}, \boldsymbol{v} \rangle + e \mod q = \langle \epsilon, \boldsymbol{v} \rangle + \langle \boldsymbol{y}, \boldsymbol{v} \rangle + e \mod q.$$

Focusing now on the second term, we observe that

$$\langle \boldsymbol{y}, \boldsymbol{v} \rangle = (\mathcal{L}^\vee)^{-1}\langle \boldsymbol{y}, \boldsymbol{v} \rangle \mathcal{L}^\vee = \langle (\mathcal{L}^\vee)^{-1}\boldsymbol{y}, \mathcal{L}^{-1}\boldsymbol{v} \rangle = \langle \boldsymbol{s}, \boldsymbol{a} \rangle$$

[12]This is technically solution to $\mathtt{CVP}^{(q)}_{\mathcal{L},d}$, however as we will see later in the proof, there is a reduction from one to another.

since $\mathcal{L}^{-1} = (\mathcal{L}^\vee)^T$.

To finish the proof we need to show that the remaining term $\langle \epsilon, \boldsymbol{v} \rangle + e$ is distributed within negligible statistical distance of Ψ_β. This is obtained by noting that summing two normal distributions with the standard deviation within some bound, gives us a distribution that is indeed close enough to Ψ_β as desired. This is Claim 3.10 in the original paper by Regev. □

Proposition 4.10 (Step 2 - quantum)**.** *There exists an efficient quantum algorithm that, given any n-dimensional lattice $\mathcal{L}$, a number $d < \lambda_n(\mathcal{L}^\vee)/2$, and an oracle that solves $\mathtt{CVP}_{\mathcal{L}^\vee,d}$, outputs a sample from $D_{\mathcal{L},\sqrt{n}/(\sqrt{2}d)}$.*

Lastly, we can finish the proof of Theorem 4.1 by presenting the following proposition that provides reduction from `SIVP` and `GapSVP` to `DGS`.

Proposition 4.11. *Let $\mathcal{L}$ be an n-dimensional lattice and let r be such that $r \geq \sqrt{2}\eta_\epsilon(\mathcal{L})$ where $\epsilon \leq \frac{1}{10}$. Then, the probability that a set of n^2 vectors chosen independently from $D_{\mathcal{L},r}$ contains no n linearly independent vectors is exponentially small.*

4.2.2 Pseudorandomness of LWE

In this section we will prove that the search version of the LWE (given sample from the LWE distribution, find the secret $\boldsymbol{s}$) to the decision version - given a sample from $\mathbb{Z}_q^n \times Z_q$ decide weather it is a LWE sample or a uniform one. More precisely, if our sample is of the form $(\boldsymbol{a}, b)$, there is no way for us to tell if b is uniform or actually $b = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e$ for some secret $\boldsymbol{s}$ and small error e.

In general, the decision version is considered more suitable for cryptographic purposes because it is often easier to analyze and prove security guarantees for. For many cryptographic problems, it is computationally infeasible to find the solution to the search version, but it is possible to determine the correct answer to the decision version with high probability. Thus, cryptographic schemes are often designed based on decision versions of hard computational problems

The statement can be phrased as follows.

Theorem 4.12 (**regev**, 4.2 - Decision to Search)**.** *Let $n \geq 1$ be some integer, $2 \leq q \leq \mathsf{poly}(n)$ be a prime, and χ be some distribution on $\mathbb{Z}_q$. Assume that we have access to procedure $\mathtt{W}$ that for all $\boldsymbol{s}$ accepts with probability exponentially close to 1 on inputs from $A_{\boldsymbol{s},\chi}$ and rejects with probability exponentially close to 1 on inputs from U. Then, there exists an efficient algorithm $\mathtt{W'}$ that, given samples from $A_{\boldsymbol{s},\chi}$ for some s, outputs $\boldsymbol{s}$ with probability exponentially close to 1.*

In other words, if we have an oracle (called procedure in this theorem) that solves the decision-LWE, then there is an efficient (running in polynomial time) algorithm

that outputs us the secret $\boldsymbol{s}$ used for the generation of the sample (if the sample was indeed taken from $A_{\boldsymbol{s},\chi}$).

Proof. We are given a sample $(\boldsymbol{a}, b)$ and an oracle that tells us whether this element was chosen uniformly at random, or whether b is related to $\boldsymbol{a}$ via $b = \langle \boldsymbol{a}, \boldsymbol{s} + e$. Note that by definition in both cases $\boldsymbol{a}$ is chosen uniformly from $\mathbb{Z}_q^n$. The idea to obtain the secret $\boldsymbol{s}$ is relatively simple. We pick some element $k \in \mathbb{Z}_q^n$ and check coordinate by coordinate if this is the respective coordinate of our key $\boldsymbol{s}$. The procedure is identical for each coordinate hence we will only consider the case of the first one. Pick any k as before and consider the transformation $(\boldsymbol{a} + (l, 0, \cdots, 0), b + k \cdot l)$ for $l \in Z_q^n$ chosen uniformly at random. We have now three cases to consider.

1. Our original sample was in fact taken from uniform distribution. Then it is easy to see that the transformation maps it again to uniform and nothing is changed. In this case, there is no $\boldsymbol{s}$ to be found and we terminate.

2. The sample was taken from the $A_{\boldsymbol{s},\chi}$ distribution. Then either:

   (a) $k = s_1$. In this case, the sample is mapped back to $A_{\boldsymbol{s},\chi}$ because

   $$\langle (a_1 + l, a_2, \cdots, a_n), \boldsymbol{s} \rangle + e = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + l \cdot s_1 + e = b + k \cdot l.$$

   We can now proceed with s_2 in a similar manner.

   (b) Or $k \neq s_1$ in which case the sample is mapped to a uniform distribution and oracle tells us we picked wrong. Note that this requires our q to be prime because otherwise it might happen that $a_1 \cdot k = a_1 \cdot s_1$ as either of the three could be a zero divisor. We have therefore picked wrong k and we need to pick another one.

Note that by the choice of our $q \leq \mathsf{poly}(n)$, we can try all of them. □

4.2.3 LWE cryptosystem

Now that we have a solid hardness assumptions, we can attempt to construct a cryptosystem that employs those results. The following public key cryptosystem was presented in the same paper. To keep the notation consistent with previous section, we will slightly deviate from the original.

We begin by specifying our parameters. Let us denote by n our security parameter. As before, the scheme is characterized by two integers m and q and a probability distribution χ over $\mathbb{Z}_q$. To now make the scheme secure and correct, we should choose q prime between n^2 and $2n^2$, $m = (1 + \epsilon)(n + 1) \log q$ for some arbitrary constant $\epsilon > 0$. We define the distribution χ to be $\bar{\Psi}_{\alpha(n)}$ where $\alpha(n) = o(1/(\sqrt{n} \log n))$ (recall from Section 2.4 that it means $\lim_{n\to\infty} \alpha(n) \cdot \sqrt{n} \log n = 0$).

KeyGen:

- Choose $\boldsymbol{s} \in \mathbb{Z}_q^n$ uniformly at random. This is the **private key**.
- For $i = 1, \ldots, m$ choose m vectors $\boldsymbol{a}_i \in \mathbb{Z}_q^n$ independent from the uniform distribution. Additionally choose m elements $e_i \in \mathbb{Z}_q$ independently according to χ. The **public key** is the array of m vectors of the form $(\boldsymbol{a}_i, b_i)$ where each b_i is given by $b_i = \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i$.

Encrypt:
To encrypt a single bit we choose a random set S uniformly among all 2^m subsets of $\{1, \ldots, m\}$. The encryption is $(\sum_{i \in S} \boldsymbol{a}_i, \sum_{i \in S} b_i)$ if the bit is 0, and $(\sum_{i \in S} \boldsymbol{a}_i, \lfloor q/2 \rfloor + \sum_{i \in S} b_i)$ otherwise.
Decrypt:
The decryption of a pair $(\boldsymbol{a}, b)$ is 0 if $b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle$ is closer to 0 than to $\lfloor q/2 \rfloor$ modulo q.

Example 4.2. Almost exactly like in the Example 4.1, set $n = 4$, $q = 17$ and $m = 4$. We pick our **secret key** $\boldsymbol{s} = (2, 1, 3, 7)$ and artificially (i.e. by design and not uniformly at random) pick

$$\boldsymbol{A} = [\boldsymbol{a}_1\, \boldsymbol{a}_2\, \boldsymbol{a}_3\, \boldsymbol{a}_4] = \begin{pmatrix} 1 & 16 & 4 & 5 \\ 16 & 4 & 5 & 1 \\ 4 & 5 & 1 & 16 \\ 5 & 1 & 16 & 4 \end{pmatrix}.$$

Take $e_1 = e_2 = 1$ and $e_3 = e_4 = -1$. Now we can compute the **public key**

$$\bar{\boldsymbol{A}} = \begin{bmatrix} \boldsymbol{A} \\ \boldsymbol{b} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A} \\ \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i \end{bmatrix} = \begin{pmatrix} 1 & 16 & 4 & 5 \\ 16 & 4 & 5 & 1 \\ 4 & 5 & 1 & 16 \\ 5 & 1 & 16 & 4 \\ 15 & 8 & 8 & 11 \end{pmatrix}.$$

To now encrypt a bit 1, we can take as our S the set $\{1, 2, 4\}$ and output the encryption as

$$(\boldsymbol{a}, b)^T = \begin{pmatrix} 1 + 16 + 5 \\ 16 + 4 + 1 \\ 4 + 5 + 16 \\ 5 + 1 + 4 \\ \lfloor 17/2 \rfloor + 15 + 8 + 11 \end{pmatrix} = \begin{pmatrix} 5 \\ 4 \\ 8 \\ 10 \\ 0 \end{pmatrix}.$$

Decryption is just another simple computation, we first compute $\langle \boldsymbol{a}, \boldsymbol{s} \rangle = 108 \equiv 6 \mod 17$ which gives us $b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = 0 - 6 \equiv 11$. Let us compare the distances to 0 and $\lfloor q/2 \rfloor = 8$. $|11 - 17| = 6 > 3 = |11 - 8|$. Hence, our decryption worked correctly as indeed, the result is closer to $\lfloor q/2 \rfloor$ than to 0. Finally, we output 1 as the decryption of our message and we are done.

Analysis Now, that we have finally defined a cryptographic scheme we need to verify it. The two remaining questions we now have are first, is this scheme correct? That is, does the decryption algorithm correctly evaluate back to the original message? This is much more difficult to prove compared to the scheme over the integers presented in 3.1. The following is a somewhat simpler version of Claim 5.2 in **regev**.

Claim 4.13 (Correctness)**.** *For the above choice of parameters and e following the χ distribution we have*

$$\Pr\left[|e| < \left\lfloor \frac{q}{2} \right\rfloor /2\right] > 1 - \delta(n) \tag{4.2}$$

for some negligible function $\delta(n)$.

This, in turn, implies that (this is Lemma 5.1)

Lemma 4.14. *The decryption is correct with probability $1 - \delta(n)$ where the $\delta(n)$ is some negligible function.*

Proof. Consider first the encryption of 0. It is given by $(\boldsymbol{a}, b)$ with $\boldsymbol{a} = \sum_{i \in S} \boldsymbol{a}_i$ and $b = \sum_{i \in S} b_i = \sum_{i \in S} \langle \boldsymbol{a}_i, \boldsymbol{s} \rangle + e_i$. Then the decryption gives us precisely $b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = \sum_{i \in S} e_i$. By our assumption, $\left| \sum_{i \in S} e_i \right| < \lfloor \frac{q}{2} \rfloor /2$ with probability at least $1 - \delta(n)$. In that case, it is closer to 0 than $\lfloor \frac{q}{2} \rfloor$ and thus correctly decrypts to 0. The case for the encryption of 1 is similar. □

Note that it seems almost trivial that we decrypt correctly, the scheme was designed in that way. This is only the case when we know the secret key $\boldsymbol{s}$ that is definitely not know to the public. This ties closely to the second and last question, that is, how secure the scheme is? We have established hardness based on average and worst-case lattice problems. However, it might be the case that our choice of parameters required for correctness, hinders on the security. This is resolved with the following theorem. Let us first define some required terminology.

Proposition 4.15 (**regev**, Lemma 5.4)**.** *For any $\epsilon > 0$ and $m \geq (1 + \epsilon)(n + 1) \log q$, if there exists a polynomial time algorithm W that distinguishes between encryptions of 0 and 1 then there exists a distinguisher Z that distinguishes between $A_{\boldsymbol{s},\chi}$ and U for a non-negligible fraction of all possible $\boldsymbol{s}$.*

Note that this closely relates to the last lemma from the previous section on the hardness of LWE. For a thorough and less technical analysis than the one given in the original paper, the reader is encouraged to look into section 5.4 in **Micci2009**.

Epilogue

Until the day of writing this paper, LWE is one of the most influential schemes that can be used for post-quantum cryptographic schemes. We will now present few

alternatives to the results and proofs presented here.

Let us begin with the proof of the main theorem, namely, the quantum part. At some point, we would like to find entirely *classical* and efficient reduction algorithm that proves the hardness of the problem. This is because we understand classical computation in much more detail than its quantum equivalent. Using classical computers we have cracked Enigma and landed on the moon. In the meantime, only recently a factorization of the integer 15 was achieved on a quantum computer using Shor's algorithm - see [Krzys: find source of that statement]. Returning to the reduction problem, Chris Peikert in his paper **peikert_classical** from 2009 has done exactly this. However, it was done in a somewhat "inefficient" way. That is, exponentially many samples are needed in the classical reduction compared to polynomial amount in the quantum version. For more details see the paper by Peikert and compare it with the original approach from Regev. It remains an important open question [Krzys: also not sure about that] till this day, if the reduction can be made efficiently fully classical.

4.3 Ring-LWE

One of the recurring problems in lattice-based cryptography is the key-size and general efficiency. In the GGH cryptosystem, the key-size is $\tilde{O}(n^4)$. In the system based on the hardness of LWE presented in the previous section, the size is in the range of $\tilde{O}(n^2)$[13]. As we will also see later, there is some minimal efficiency needed for the scheme in order to enable the boostrapping (for FHE). Unfortunately, none of the schemes presented so far satisfy those criterions and so, we need to look for something better.

One idea to improve the efficiency, is to assume some underlying structure of the space we are performing computations in. For example, we can assume that the $\boldsymbol{a}$ vectors from previous section are given to us in block of n samples $\boldsymbol{a}_1, \boldsymbol{a}_2, \ldots, \boldsymbol{a}_n \in \mathbb{Z}_q^n$ where all of the elements are related. Namely, $\boldsymbol{a}_1 = (a_1, \ldots, a_n)$ is again chosen uniformly but each $\boldsymbol{a}_i = (a_i, \ldots, a_n, -a_1, \ldots, -a_{i-1})$ is a "anti-cyclic" of the initial $\boldsymbol{a}_1$. This choice seems rather arbitrary however we will show how it is a natural consequence of everything we did so far and yields arguably the best results. For example if $n = 4$ and $q = 17$ and $\boldsymbol{a}_1 = (1, 16, 4, 5)$ as before, then $\boldsymbol{a}_3$ has the form $(4, 5, -1, -16) = (4, 5, 16, 1)$. Note that representing n vectors now takes only $O(n)$ elements from $\mathbb{Z}_q$ rather than $O(n^2)$. The underlying structure is a ring, hence the name ring-LWE (or R-LWE), that is, we replace the group $\mathbb{Z}_q^n$ by picking some ring R of degree n over $\mathbb{Z}$ and a positive modulus q defining the quotient ring $R_q := R/qR$.

[13] There are m samples of length n. Turns out that for $m > n$, the problem can become only easier, but the same holds for $m \ll n$. Therefore, in most applications, m is chosen to be roughly the size of n.

In most of the cases (as well as in this exposition), R is taken to be a *cyclotomic* ring - i.e. $R_q = \mathbb{Z}_q[x]/\langle x^n + 1\rangle$ for $n = 2^k$ which turns out to yield much simpler proofs for the somewhat weaker results.

In the year 2010, Vadim Lyubashevski, Chris Peikert and Oded Regev presented their paper "On Ideal Lattices and Learning With Errors Over Rings" **ring-lwe**. The main purpose of the paper was to "translate" the LWE problem onto a ring as was done with the SIS problem (mainly by Micciancio **ring-sis** that was followed up by other works but these results are not presented in this paper) and followed the heuristic approach behind the NTRU[14] cryptosystem **ntru**. This in particular means first, defining the ring-LWE and later proving the hardness based on some difficult lattice problems like `SVP` along with pseudorandomness of the ring-LWE distribution (analogous to 4.12 which we define later). The second issue turned out to be quite nontrivial and required good insight in the algebraic number theory as well as Gaussian measures and distributions.

4.3.1 Background on Lattices II

4.3.2 Hardness of RLWE

The following is an equivalent of Theorem 4.7 of the classical LWE. Statement is almost identical however the proof turns out to be much more difficult to achieve than it may look at the first glimpse.

Theorem 4.16.

4.3.3 Pseudorandomness of Ring-LWE

Just like in the case of classical LWE

Chinese remainder theorem for rings –> Thm II.4.12 in Top's lecture notes. For instance, they have unique factorization of ideals, and their fractional ideals form a multiplicative group; in general, neither property holds in $\mathbb{Z}[x]/\langle f(x)\rangle$ for monic irreducible $f(x)$, as demonstrated by the ring $\mathbb{Z}[x]/\langle x^2 + 3\rangle = \mathbb{Z}[\sqrt{-3}]$. (For example, in this ring $4 = 22 = (1 + \sqrt{-3})(1 - \sqrt{-3})$, but 2, $1 + \sqrt{-3}$, and $1 - \sqrt{-3}$ are all irreducible.) Toward basing fully homomorphic encryption on worst-case hardness One of the applications is **qTESLA** signature scheme.

4.4 Fully Homomorphic Encryption Using Ideal Lattices

three "generations" of fhe schemes, first original gentry, smart and explain here how we can construct a really nice homomorphic encryption scheme using ideal lattices **gentry**. present the

[14] As mentioned by Peikert in his survey: "The meaning of the acronym NTRU is somewhat mysterious; plausible candidates include "Nth degree *tru*ncated polynomial ring" and "Number Theorists 'R' Us."" - **lattice-survey**

On Ideal Lattices and Learning With Errors Over Rings

this is somewhat too difficult for me i think so ill just present main findings without proofs and details **regev**,
First explain what lattices are.
How do lattices relate to LWE? The secret key is associated with a random vector. then show how ring-lwe satisfies both of our requirements **ring-lwe**, namely, the believed hardness for quantum computers (SVP or approximate SVP) and FHE. Show also the problem with ring-LWE because the lattices that are used there are ideal lattices which obviously possess more structure than "normal" lattices.

5 Conclusions

a quote from **intro_cryp**: "In real world scenarios, cryptosystems based on N P-hard or N P-complete problems tend to rely on a particular subclass of problems, either to achieve efficiency or to allow the creation of a trapdoor. When this is done, there is always the possibility that some special property of the chosen subclass of problems makes them easier to solve than the general case"

Remark 5.1. We are still faced with a problem that is inherent to all of modern-day cryptography. That is, we are assuming the hardness of the problem based on our inability to efficiently solve it. As correctly trivialized by Daniel J. Bernstein **bernstein**: “nobody has figured out an attack so we conjecture that no attack exists”. It might so happen that tomorrow someone finds an efficient (polynomial time) algorithm to find the shortest vector in a given lattice and our secrets are compromised. This is exactly what happened in the case of RSA cryptosystem when Shor found such efficient algorithm for integer factorization. There is not much we can do about it at least with our current approach to cryptography which is based on very precise complex-theoretic assumptions. This is because complexity theory does not provide any tools to prove that an efficient algorithm does not exist for any given problem. This puts cryptography as a empirical study (like physics)

References

algebra Daniel A. Marcus. *Number fields*. Universitext. Springer-Verlag, New York-Heidelberg, 1977, pp. viii+279. ISBN: 0-387-90279-1.

babai László Babai. "On Lovász' lattice reduction and the nearest lattice point problem". In: *Combinatorica* 6 (1986), pp. 1–13.

bernstein Daniel J. Bernstein. "Introduction to post-quantum cryptography". In: *Post-Quantum Cryptography*. Ed. by Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 147–191. ISBN: 978-3-540-88702-7. DOI: 10.1007/978-3-540-88702-7_5. URL: https://doi.org/10.1007/978-3-540-88702-7_5.

book Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. *An introduction to mathematical cryptography*. Second. Undergraduate Texts in Mathematics. Springer, New York, 2014, pp. xviii+538. ISBN: 978-1-4939-1710-5; 978-1-4939-1711-2. DOI: 10.1007/978-1-4939-1711-2. URL: https://doi.org/10.1007/978-1-4939-1711-2.

break1 Phong Q. Nguyen. "Cryptanalysis of the Goldreich-Goldwasser-Halevi Cryptosystem from Crypto '97". In: *Advances in Cryptology - CRYPTO '99, 19th Annual International Cryptology Conference, Santa Barbara, California, USA, August 15-19, 1999, Proceedings*. Vol. 1666. Lecture Notes in Computer Science. Springer, 1999, pp. 288–304. DOI: 10.1007/3-540-48405-1_18.

break2 Phong Q. Nguyen and Oded Regev. "Learning a Parallelepiped: Cryptanalysis of GGH and NTRU Signatures". In: *Advances in Cryptology - EUROCRYPT 2006*. Ed. by Serge Vaudenay. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 271–288. ISBN: 978-3-540-34547-3.

cvp-hard I. Dinur et al. "Approximating CVP to Within Almost-Polynomial Factors is NP-Hard". In: *Combinatorica* 23 (Apr. 2003), pp. 205–243. DOI: 10.1007/s00493-003-0019-y.

discr Chris Peikert. "An Efficient and Parallel Gaussian Sampler for Lattices". In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 80–97. ISBN: 978-3-642-14623-7.

easy_fhe Craig Gentry. "Computing Arbitrary Functions of Encrypted Data". In: *Commun. ACM* 53.3 (Mar. 2010), pp. 97–105. ISSN: 0001-0782. DOI: 10.1145/1666420.1666444. URL: https://doi.org/10.1145/1666420.1666444.

gentry Craig Gentry. "Fully Homomorphic Encryption Using Ideal Lattices". In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC '09. Bethesda, MD, USA: Association for Computing Machinery, 2009, pp. 169–178. ISBN: 9781605585062. DOI: 10.1145/1536414.1536440. URL: https://doi.org/10.1145/1536414.1536440.

gentry_phd Craig Gentry. "A Fully Homomorphic Encryption Scheme". AAI3382729. PhD thesis. Stanford, CA, USA, 2009. ISBN: 9781109444506.

ggh Oded Goldreich, Shafi Goldwasser, and Shai Halevi. "Public-key cryptosystems from lattice reduction problems". In: *Advances in Cryptology — CRYPTO '97*. Ed. by Burton S. Kaliski. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 112–131. ISBN: 978-3-540-69528-8.

hnf Daniele Micciancio. "Improving Lattice Based Cryptosystems Using the Hermite Normal Form". In: *LNCS* 2146 (Nov. 2001). DOI: 10.1007/3-540-44670-2_11.

int_scheme Marten van Dijk et al. "Fully Homomorphic Encryption over the Integers". In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 24–43. ISBN: 978-3-642-13190-5.

intro_cryp Jill Pipher Jeffrey Hoffstein and Joseph H. Silverman. *An Introduction to Mathematical Cryptography*. Springer New York, NY, 2016. DOI: https://doi.org/10.1007/978-1-4939-1711-2.

katz Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. 2nd. Chapman & Hall/CRC, 2014. ISBN: 1466570261.

lattice-survey Chris Peikert. "A Decade of Lattice Cryptography". In: *Foundations and Trends® in Theoretical Computer Science* 10.4 (2016), pp. 283–424. ISSN: 1551-305X. DOI: 10.1561/0400000074. URL: http://dx.doi.org/10.1561/0400000074.

Micci2009 Daniele Micciancio and Oded Regev. "Lattice-based Cryptography". In: *Post-Quantum Cryptography*. Ed. by Daniel J. Bernstein, Johannes Buchmann, and Erik Dahmen. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 147–191. ISBN: 978-3-540-88702-7. DOI: 10.1007/978-3-540-88702-7_5. URL: https://doi.org/10.1007/978-3-540-88702-7_5.

ntru "NTRU: A ring-based public key cryptosystem". In: *Algorithmic Number Theory*. Ed. by Joe P. Buhler. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 267–288. ISBN: 978-3-540-69113-6.

peikert_classical Chris Peikert. "Public-key cryptosystems from the worst-case shortest vector problem: extended abstract". In: *Electron. Colloquium Comput. Complex.* TR08 (2009).

primal Ronald L. Rivest, Len Adleman, and Michael L. Dertouzos. "On Data Banks and Privacy Homomorphisms". In: 1978.

qTESLA Erdem Alkim et al. *The Lattice-Based Digital Signature Scheme qTESLA*. Cryptology ePrint Archive, Paper 2019/085. https://eprint.iacr.org/2019/085. 2019. URL: https://eprint.iacr.org/2019/085.

reductions Daniele Micciancio. "Efficient reductions among lattice problems". In: *ACM-SIAM Symposium on Discrete Algorithms*. 2008.

regev Oded Regev. "On Lattices, Learning with Errors, Random Linear Codes, and Cryptography". In: vol. 56. Jan. 2005, pp. 84–93. DOI: 10.1145/1568318.1568324.

regev_survey Oded Regev. "The Learning with Errors Problem (Invited Survey)". In: *2010 IEEE 25th Annual Conference on Computational Complexity*. 2010, pp. 191–204. DOI: 10.1109/CCC.2010.26.

revisited Jung Hee Cheon and Damien Stehlé. "Fully Homomophic Encryption over the Integers Revisited". In: *Advances in Cryptology – EUROCRYPT 2015*. Ed. by Elisabeth Oswald and Marc Fischlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 513–536. ISBN: 978-3-662-46800-5.

ring-lwe Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *On Ideal Lattices and Learning with Errors Over Rings*. Cryptology ePrint Archive, Paper 2012/230. 2012. URL: https://eprint.iacr.org/2012/230.

ring-sis Daniele Micciancio. "Generalized Compact Knapsacks, Cyclic Lattices, and Efficient One-Way Functions". In: *computational complexity* 16.4 (Dec. 2007), pp. 365–411. ISSN: 1420-8954. DOI: 10.1007/s00037-007-0234-9. URL: https://doi.org/10.1007/s00037-007-0234-9.

stein William A. Stein. "A Brief Introduction to Classical and Adelic Algebraic Number Theory". In: 2004.

svp-hard Daniele Micciancio. "The Shortest Vector in a Lattice is Hard to Approximate to within Some Constant". In: *SIAM Journal on Computing* 30.6 (2001), pp. 2008–2035. DOI: 10.1137/S0097539700373039. eprint: https://doi.org/10.1137/S0097539700373039. URL: https://doi.org/10.1137/S0097539700373039.

two_faces Phong Q. Nguyen and Jacques Stern. "The Two Faces of Lattices in Cryptology". In: *Cryptography and Lattices*. Ed. by Joseph H. Silverman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 146–180. ISBN: 978-3-540-44670-5.