university of groningen / faculty of science and engineering / mathematics and applied mathematics

Usage of lattices in post-quantum cryptography

Bachelor's Project Mathematics

May 2023

Student: K.J. Pudowski

First supervisor: Dr. P. Kılıçer

Second assessor: Dr. M. Seri

Contents

1 Introduction

1.1 Actual introduction here with some funny name

In recent years, the topic of lattice based cryptography (LBC) has seen many developments towards an efficient and quantumly secure basis for cryptographic constructions. Starting with Ajtai's seminal work [Ajt96] on one-way functions that can be reduced to as one of the main contenders for an efficient and resistant to quantum computers scheme.

1.2 Motivation

include the table from page 16 of [Ber09] of systems broken by quantum computers
quote from [Ber09]: "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

This bachelor thesis can be seen as a survey on the recent developments in the LBC and improvements in fully homomorphic cryptographic schemes. Nonetheless, there are few minor contributions that we note here.

Firstly, and most importantly, the introduction of many examples that we hope will be of great help for anyone trying to learn and understand LBC schemes. These include comptutations as well as images and tables that help visualize some key concepts related to Gaussian samples or lattices.

Secondly, we present and explicitly state few results that are otherwise left simply as claims or propositions. These mostly include parts on the algebraic number theory for the Section 3.3 on ring-LWE.

1.4 Outline of the paper

The sections in this paper are to be read consecutively with one exception. The Section 2.3 on algebraic number theory is mostly useful to prove concepts for ring-LWE from Section 3.3. If one is not interested in the proofs from this section, it is possible to skip those two since the idea behind ring-LWE is almost identical to the LWE with just minor changes that are not crucial to understand other parts of this 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 $a, t, \beta, \ldots$ are represented by non-bold, latin or greek letters.
Bold symbols $\boldsymbol{v}, \boldsymbol{x}, \boldsymbol{s}, \ldots$ will denote vectors.
Algorithms will be represented by textsf font names such as Encrypt or Evaluate.
Traditionally, the symbols $\mathbb{Z}, \mathbb{Q}, \mathbb{R}, \mathbb{C}$ shall represent the sets of integers, 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 $\{1, 2, \ldots, \lfloor m \rfloor\}$.

2.2 Lattices

Basic definitions

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

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

In other words, Λ is the $\mathbb{Z}$-span of the specific basis $\boldsymbol{B}$.

There are many bases for a lattice (actually, for $n \geq 2$, 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 cryptosystems later like the GGH.

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 $\boldsymbol{b}_1 = \binom{1}{0}$ and $\boldsymbol{b}_2 = \binom{0}{1}$

2. $\mathcal{L} = \{(z_1, z_2) : z_1 + z_2 \text{ is even}\}$

3. $\mathcal{L} = \begin{pmatrix} 1/3 & \pi \\ 0 & 21/7 \end{pmatrix}$

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 arguably the two most fundamental resuls 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 [HPS14], chapter 6.4.

Lemma 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. $\boldsymbol{B}_1 = U\boldsymbol{B}_2$. Then it easily follows that

$$|\det(\boldsymbol{B}_1)| = \det(\mathcal{L}) = |\det(U \cdot \boldsymbol{B}_2)| = |\det(U)| \cdot |\det(\boldsymbol{B}_2)| = |\det(\boldsymbol{B}_2)|$$

□

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 [Gen09b]).

We can also need some notion of the shortest possible vector of a lattice. These will be very useful in the study of the hardness of a given problem.

Definition 2.5. For a n-dimensional lattice $\mathcal{L}$, we denote its shortest nonzero vector (also called the *minimum distance*) by $\lambda_1(\mathcal{L})$. Formally

$$\lambda_1(\mathcal{L}) := \min_{0 \neq \boldsymbol{v} \in \mathcal{L}} ||\boldsymbol{v}||$$

More generally, for $1 \leq i \leq n$, the ith sucessive minimum of $\mathcal{L}$ is

$$\lambda_i(\mathcal{L}) := \inf\{r : \mathcal{L} \text{ has } i \text{ linearly independent vectors of length at most } r\}.$$

For example, in order to be able to go "away" from the lattice $\mathcal{L}$ "out" into the ambient space $\mathbb{R}^n$ and still be able to return to the original vector, we need to make sure the distance d we travel is at most $\lambda_1(\mathcal{L})/2$. This is for example a requirement in the `BDD` problem where the decoding is *promised*. For the exact definition see Section 2.4.

Aditionally, most of the problems used in this paper are the *approximate* versions to some varying constant usually called γ. As we will see later, γ very often depends on $\lambda_1(\mathcal{L})$ which roughly measures how "strict" or "precise" we can make our problem. This will be made explicing in the Section 2.4 on the hardness and complexity theory.

Dual

Definition 2.6 (Dual)**.** For a lattice $\mathcal{L} \subset \mathbb{R}^n$ its $\mathbb{Z}$-*dual* is

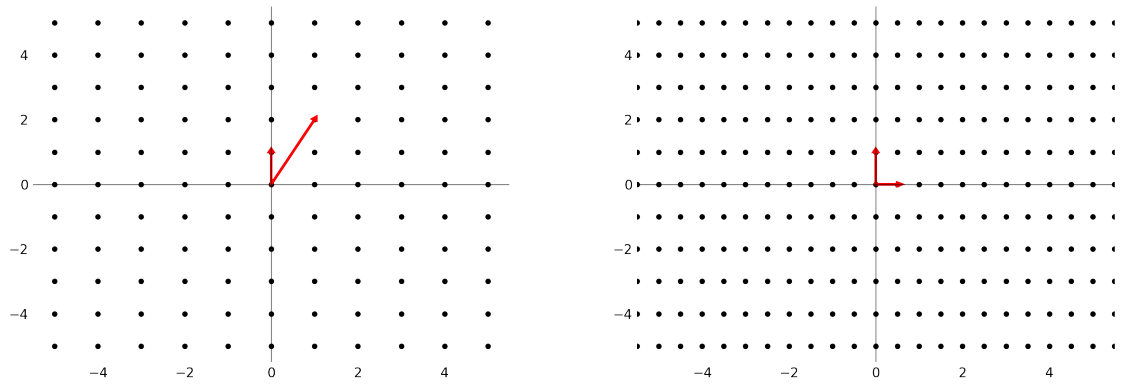
$$\mathcal{L}^\vee = \{y \in \mathbb{R}^n : \langle y, \mathcal{L} \rangle \subseteq \mathbb{Z}\}.$$

We simply require that the elements of the dual to be precisely those vectors, that yield an integer when "multiplied" with an element of our lattice. Note that this is different from our standard definition of a dual. Namely, it is not the orthogonal compliment of our starting space, i.e. not all of the elements of the dual have 0 dot product against the vectors of the lattice.

Example 2.5. Take $\mathcal{L} = \mathbb{Z}\binom{1}{2} + \mathbb{Z}\binom{0}{1}$. To calculate the dual of $\mathcal{L}$ we need our $y = \binom{a}{b}$ elements to satisfy $a + 2b \in \mathbb{Z}$ and $b \in \mathbb{Z}$ which is equivalent to asking $a \in (1/2)\mathbb{Z}$ and so $\mathcal{L}^\vee = \binom{1/2}{0}\mathbb{Z} + \binom{0}{1}\mathbb{Z}$ See Figure 1 where the red vectors represent the basis.

Note that $\mathcal{L}^\vee$ is itself a lattice of the same dimension. One can imagine the dual as the original lattice "flipped" in the sence that the shortes element becomes the longest and vice versa.

Figure 1: Lattice $\mathcal{L}$ from example 2.5 and its dual.



(a) $\mathcal{L}$ (b) $\mathcal{L}^\vee$

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 books like [Mar77] and [Rib01].

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 $\mathbb{Q}$ and when not stated otherwise, will be denoted by n. There exists a monic irreducible polynomial[1] $f \in \mathbb{Q}[x]$ such that $K \cong \mathbb{Q}[x]/(f)$. In fact, every monic and irreducible polynomial in $\mathbb{Q}[x]$ defines a number field via such isomorphism.

Definition 2.7 (Algebraic integer)**.** An element $\alpha \in \mathbb{C}$ is an *algebraic integer* if and only if, it is a root of some monic polynomial in $\mathbb{Z}[x]$.

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

Definition 2.8 (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:

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

[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 $\mathbb{Q}[x]$.

Example 2.6. 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 2.7. The ring of Gaussian integers $\mathbb{Z}[\sqrt{-1}] = \{a + b\sqrt{-1} : a, b \in \mathbb{Z}\}$ 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 $\mathbb{Q}$. Note that $\mathcal{O}_K$ is spanned by $\{1, \sqrt{-1}\}$.

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

Lemma 2.9. *Let $d \in \mathbb{Z}$ be a square-free integer. For the field $K = \mathbb{Q}(\sqrt{d})$, its ring of integers is*

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

Proof. Take $d \equiv 1 \bmod 4$ square-free. [Pinar: if you dont have time to finish the proof give a reference.] □

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

Remark 2.8. What is worth noting here, is that for a number field $\mathbb{Q}(\alpha)$ for some $\alpha \in \overline{\mathbb{Q}}$, 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 as motivated in the next subsection. In general, a field $K = \mathbb{Q}(\alpha)$ such that $\mathcal{O}_K = \mathbb{Z}[\alpha]$ is called a *monogenic* field or a *simple algebraic extension*. For more details on orders, look at for example Chapter 5 of [Ste04].

Recall that an ideal $\mathcal{I}$ of some ring R is an additive subgroup of R (if $x, y \in \mathcal{I}$ then $x - y \in \mathcal{I}$) and closed under multiplication of the elements in R ($x \in \mathcal{I}$ implies $r \cdot x \in \mathcal{I}$ for any $r \in R$). We call an ideal in a ring of integers by *integral ideal*.

One useful property of the integers we would like to preserve is the unique prime decomposition of its elements as stated in the Fundamental Theorem of Algebra. As the name might suggest[2], the ring of integers has a property analogous to such decompositions. Namely, that every nonzero proper ideal factors uniquely into prime ideals. Such ring in general is called a *Dedekind domain*.

More precisely, take D to be a Dedekind [Pinar: Dedekind] domain (for example $\mathcal{O}_K$) and $\mathcal{I} \subset D$ a nonzero ideal (and also not an empty set). Then $\mathcal{I}$ factorizes uniquely (up to ordering) into prime ideals $\mathcal{I} = \mathfrak{p}_1^{e_1}\mathfrak{p}_2^{e_2} \dots \mathfrak{p}_r^{e_r}$ where each e_i is an integer and each $\mathfrak{p}_i$ is prime. Additionally it can be shown that in a Dedekind domain, every prime ideal $\mathfrak{p}$ is also *maximal* (the only superideal [Pinar: superideal?] of $\mathfrak{p}$ is the D itself). This in turn implies that $D/\mathfrak{p}$ is a finite field of order equal to the index of $|\mathcal{O}_K/\mathcal{I}|$ as an additive subgroup of $\mathcal{O}_K$. [Pinar: D or $\mathcal{O}_K$? Mixing notation. Not $|\mathcal{O}_K/\mathcal{I}|$ but $|\mathcal{O}_K/\mathfrak{p}|$]

Definition 2.10. In a number field K of degree n, a lattice in K is the $\mathbb{Z}$-span of a $\mathbb{Q}$-basis of K.

Examples of lattices in K include the ring of integers $\mathcal{O}_K$, fractional ideals of $\mathcal{O}_K$ (introduced in the section on ring-LWE), and aforementioned orders in K.

[2] As stated on p.34 in [Ste04] "*...it is this unique factorization that initially motivated the introduction of rings of integers of number fields over a century ago*".

Embeddings in $\mathbb{C}$

Let $K = \mathbb{Q}(\alpha)$ be a number field of degree n for a root α of some irreducible $f \in \mathbb{Q}[x]$. It can be shown, that there are exactly n embeddings (injective ring homomorphisms) of K in $\mathbb{C}$. This can be shown noting that since $\mathbb{Q}$ has characteristic 0, all of the roots of f over K must be distinct. Therefore α can only 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 $\sqrt{d}$.

Definition 2.11 (Canonical embedding)**.** Let $K = \mathbb{Q}(\alpha)$ be a simple algebraic extension (generated by just α) of $\mathbb{Q}$ of degree n. Let $f(x)$ be the minimal polynomial of α. Denote by $\sigma_i : K \to K$ the $\mathbb{Q}-$automorphism such that $\sigma_i(\alpha) = \alpha_i$ where $\alpha := \alpha_1$ and α_i are the roots of $f(x)$. The *canonical embedding* σ is defined as

$$\sigma : K \to \mathbb{C}^n, \quad \alpha \mapsto (\sigma_1(\alpha), \sigma_2(\alpha), \ldots, \sigma_n(\alpha)).$$

Example 2.10. For $K = \mathbb{Q}(\sqrt{5})$ we have two embeddings $a + b\sqrt{5} \mapsto a \pm b\sqrt{5}$ and so the cannonical embedding is given by

$$a + b\sqrt{5} \mapsto (a + b\sqrt{5},\, a - b\sqrt{5}).$$

One last thing we will need to introduce are the *trace* of some field K.

Definition 2.12. Let K be a number field with $n = [K : \mathbb{Q}]$ and let $\sigma_1, \sigma_2, \ldots, \sigma_n$ denote the embeddings of K in $\mathbb{C}$ just like above. For $\alpha \in K$ we define the *trace* as

$$\mathrm{Tr}_{K/\mathbb{Q}}(\alpha) = \sum_{i \in [n]} \sigma_i(\alpha).$$

Cyclotomic fields

One particularly handy family of polynomials are the *cyclotomic polynomials*. As defined momentarily, they are the minimal polynomials of the primitive roots of unity which poses many interesting algebraic properties. They also turn out to be much easier to compute and in general perform computational operations over, in contrast to most other polynomials.

Definition 2.13 (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 $\zeta^k \neq 1$ for $1 \leq k < n$.

It is also true that all ζ^k for $1 \leq k < n$ and $\gcd(k, n) = 1$ are conjugates of ζ. It follows that $\mathbb{Q}(\zeta)$ has degree $\varphi(n)$ over $\mathbb{Q}$. This leads us to the following proposition. Recall that the Galois group $\mathrm{Gal}(K/\mathbb{Q})$ of a finite extension of degree n of K over $\mathbb{Q}$ (or any other field for that matter) is the group of all $\mathbb{Q}-$automorphism. That is, $\mathrm{Gal}(K/\mathbb{Q})$ consists of maps $\tau : K \to K$ that fix $\mathbb{Q}$ elementwise. As a result the image of $\mathbb{Q}$ under τ is still $\mathbb{Q}$.

Proposition 2.14 (p. 13 in [Mar77]). *The Galois group of $K = \mathbb{Q}(\zeta_n)$ over $\mathbb{Q}$ is isomorphic to the multiplicative group of integers mod n*

$$\mathbb{Z}_n^* = \{k : 1 \leq k < n,\ \gcd(k, n) = 1\}.$$

For each $k \in \mathbb{Z}_n^*$ the corresponding automorphism in $Gal(K/\mathbb{Q})$ sends ζ to ζ^k.

The minimal polynomial Φ_n of ζ over $\mathbb{Q}$ is called the n-th cyclotomic polynomial. Formally we define it as

$$\Phi_n(x) = \prod_{\substack{\gcd(k,n)=1 \\ 1 \leq k < n}} \left(x - \zeta^k\right) = \prod_{\substack{\gcd(k,n)=1 \\ 1 \leq k < n}} \left(x - e^{2\pi k\sqrt{-1}/n}\right) \in \mathbb{Z}[x].$$

The following equality is very useful for computing the polynomial itself:

$$\Phi_n(x) = (x^n - 1) \Big/ \prod_{\substack{d|n \\ 1 \leq d < n}} \Phi_d(x) \tag{2.1}$$

Example 2.11. To find $\Phi_8(x)$, we can use equation 2.1 to first precompute:

$$\begin{aligned}
\Phi_1(x) &= x - 1 \\
\Phi_2(x) &= x^2 - 1 \Big/ \Phi_1(x) = \frac{x^2 - 1}{x - 1} = x + 1 \\
\Phi_4(x) &= x^4 - 1 \Big/ \Phi_1(x) \cdot \Phi_2(x) = \frac{x^4 - 1}{(x-1)(x+1)} = x^2 + 1
\end{aligned}$$

And use it one more time to find:

$$\begin{aligned}
\Phi_8(x) &= (x^8 - 1) \Big/ \prod_{d|4} \Phi_d(x) = \Phi_1(x) \cdot \Phi_2(x) \cdot \Phi_4(x) \\
&= (x^8 - 1) \Big/ (x - 1)(x + 1)(x^2 + 1) \\
&= (x^4 + 1) \cdot (x^2 + 1)(x + 1)(x - 1) \Big/ (x^2 + 1)(x + 1)(x - 1) \\
&= x^4 + 1,
\end{aligned}$$

as desired.

As a useful corollary of the equation 2.1, we can prove that for n a power of 2, $n = 2^k$ for some $k \geq 1$, the cyclotomic polynomial $\Phi_n(x)$ is of the form $x^{2^{k-1}} + 1$.

Corollary 2.15. *Let $n = 2^k$ for some $k \in \mathbb{Z}_{>0}$. Then $\Phi_n(x) = x^{2^{k-1}} + 1$.*

Proof. Attempting to prove it by induction, we first check that indeed $\Phi_2(x) = x + 1$. For the step, assume that we have $\Phi_n(x) = x^{2^{k-1}} + 1$ for some $n = 2^k$. Then it is not difficult to see that $x^{2^k} - 1$ "splits" as

$$\begin{aligned}
x^{2^k} &= (x^{2^{k-1}} + 1)(x^{2^{k-1}} - 1) \\
&= (x^{2^{k-1}} + 1)(x^{2^{k-2}} + 1) \cdot \ldots \cdot (x + 1)(x - 1) \\
&= (x^{2^{k-1}} + 1) \cdot \Phi_{2^{k-1}}(x) \cdot \ldots \cdot \Phi_2(x) \cdot \Phi_1(x) \\
&= (x^{2^{k-1}} + 1) \cdot \prod_{d|n} \Phi_d(x)
\end{aligned}$$

and so the equation 2.1 becomes

$$\begin{aligned}\Phi_n(x) =&(x^n-1)\Big/\prod_{d|n}\Phi_d(x)\\ =&(x^{2^{k-1}}+1)\cdot\prod_{d|n}\Phi_d(x)\Big/\prod_{d|n}\Phi_d(x)\\ =&x^{2^{k-1}}+1\end{aligned}$$

□

As mentioned in Remark 2.8, rings of integers are usually not generated by a single element. 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 as we will only need to look at what happens with ζ instead of all other generators.

Proposition 2.16. *The ring of integers of a cyclotomic number field $K = \mathbb{Q}[x]/\Phi_m(x) \cong \mathbb{Q}(\zeta)$ is $\mathcal{O}_K = \mathbb{Z}[\zeta]$. Additionally, if $n = \varphi(m)$ is the degree of K, then $\mathbb{Z}[\zeta]$ is generated by $\{1, \zeta, \zeta^2, \ldots, \zeta^{n-1}\}$.*

Example 2.12. Similarly, for $K = \mathbb{Q}(\zeta_5) = \mathbb{Q}[x]/(x^4+x^3+x^2+x+1)$, we have four embeddings. These are precisely $\sigma_i(\zeta) = \zeta^i$ and now the canonical embedding maps $a + b\zeta + c\zeta^2 + d\zeta^3$ to

$$\begin{pmatrix} a+b\zeta+c\zeta^2+d\zeta^3\\ a+b\zeta^2+c\zeta^4+d\zeta^6\\ a+b\zeta^3+c\zeta^6+d\zeta^9\\ a+b\zeta^4+c\zeta^8+d\zeta^{12}\end{pmatrix} = \begin{pmatrix} a+b\zeta+c\zeta^2+d\zeta^3\\ a+b\zeta^2+c\zeta^4+d\zeta\\ a+b\zeta^3+c\zeta+d\zeta^4\\ a+b\zeta^4+c\zeta^3+d\zeta^2\end{pmatrix} =$$

$$\begin{pmatrix} a+b\zeta+c\zeta^2+d\zeta^3\\ a+d\zeta+b\zeta^2+e\zeta^3\\ a+c\zeta+e\zeta^2+b\zeta^3\\ a+e\zeta+d\zeta^2+c\zeta^3\end{pmatrix} = \big(sthhere\big)$$

[Krzys: i feel like i should get something nice but cant figure it our, its probs the ζ^4]

In order to make σ a ring homomorphism, we define the addition and multiplication in $\mathbb{C}^n$ to be coordinate-wise (all the conditions for a ring structure are immediate to see when written down because the $\mathbb{C}$ itself is a ring).

Another very handy property of the cyclotomic fields is that for $n > 3$, the n-th cyclotomic field has $\varphi(n)$ embeddings in $\mathbb{C}$. These are precisely the $\varphi(n)$ automorphisms where $\sigma_i(\zeta) = \zeta^i$ is simply the conjugation of the generator. Therefore K has only complex embeddings and we call such field a CM-field. Since the $\mathbb{Q}$-automorphisms map ζ to one of its conjugates, it is easy to see that any automorphism $\tau_k : K \to K$ mapping $\tau_k(\zeta) = \zeta^k$ simply permutes the coordinates of the canonical embedding. More precisely, $\sigma_i(\tau_k(\zeta)) = \sigma_i(\zeta^k) = \zeta^{ik}$ for all $i, k \in \mathbb{Z}^*_{\varphi(n)}$. Hence $\sigma \varnothing \tau_k$ is just a permutation of the coordinates.

Ideal lattices

Let us now get back to the topic of lattices for a moment and try to connect those two subjects a bit. Although not shown in this paper, we note that each element of $\mathcal{O}_K$ can be expressed as an integer linear combination of some basis $\boldsymbol{\mathcal{B}} = \{\beta_1, \beta_2, \ldots, \beta_n\} \subset \mathcal{O}_K$ (see the Exercise 2.7 as a simple case.). The set $\boldsymbol{\mathcal{B}}$ is also called the *integral basis* of $\mathcal{O}_K$ and is also a basis for K as $\mathbb{Q}$-vector space. Now recall from the beginning of the previous section that any lattice can be defined as the $\mathbb{Z}$-span of some specific basis $\boldsymbol{B} \subset \mathbb{R}^n$. Analogously, we can define a lattice for any number field K.

[Pinar: Define duals, embedding of ideals in $\mathbb{C}$.]

2.4 Complexity Theory and Hard Problems

When talking about cryptography, we cannot avoid talking about algorithms since an encryption scheme is simply that, an instruction on how to encode sensitive data. We will thus require some terminology from computational complexity theory – a field on the overlap of computer science and mathematics – and cryptography. The problems the former field is concerned with are those of the time and space required for solving computational problems. Most of the time, the goal of the study is to prove the lower bound on the resources required to solve a problem using the best know algorithm. For example, how long does it take to find a factorization of a large composite number. The concern of the later – cryptography – is how to efficiently share secrets. The goal of this section is to give an overview of few parts of both fields which we will be using throughout the paper.

Complexity theory

We begin with necessary terminology. Throughout this section, n shall denote the "size" of the input to the algorithm. Here the word "size" could mean many things like for example the bit-length of a number or its value in the base 10 representation.

In order to represent the time or space required to solve the problem, most of the time we will use the asymptotic notation of which the "big-O" is the most common. For any function f and g of some variable n, we say that $f(x) \in O(g(n))$ if there exists some constant $M > 0$ and n_0 such that $f(n) \leq Mg(n)$ for all $n \geq n_0$. Roughly speaking, the function g will eventually "grows faster" than f. Variations on this notations include:

- $f \in o(g)$ – just like $O(g)$ but the inequality holds for all constants M. This provides more strict bounds as for example $2n = o(n^2)$ but $2n^2 \neq o(n^2)$ contrary to $2n^2 = O(n^2)$ (as well as $2n = O(n^2)$).
- The "inverse" of the small o is ω. Formally $f \in \omega(g)$ if for any M there exists n_0 such that $f(n) \geq Mg(n)$ for all $n > n_0$. We call it the inverse of o because $f(n) = o(g(n))$ if and only if $g(n) = \omega(f(n))$.
- $\tilde{O}(n) = O(n \ln n)$. That is, $\tilde{O}$ hides the logarithmic factors. For example, the FFT (Fast Fourier Transform) computes the discrete Fourier transform in time $O(n \ln n) = \tilde{O}(n)$ for a sequence of length n and Shor's algorithm computes factorization of n-bit integer in time $O(n^2 \ln(n) \ln(\ln(n))) = \tilde{O}(n^2)$.

We will very often be talking about the "efficiency" of an algorithm. Intuitively, we want our procedure to run in a time reasonably proportional to the size of input. The consensus in the academia is to define "efficient" as running in polynomial time $\mathsf{poly}(n)$[3]. More precisely,

Definition 2.17. An algorithm $A(n)$ is `efficient` if there exists a polynomial $p(\cdot)$ such that the computation of $A(n)$ terminates within at most $p(n)$ steps.

To encapsulate the notion of "small" or "insignificant", 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 2.18. $f(n)$ is a `negligible` function in n if

$$\lim_{n\to\infty} n^{-c} \cdot f(n) = 0$$

for any $c > 0$.

Hard problems

The basis for modern day cryptography are so called "hard problems" of which usage was first formally introduced by S. Goldwasser and S.Micali in [GM84]. These are computational problems for which we do not know any efficient algorithm for (so far). The current approach to cryptography is to assume that some given problem is "difficult" to solve and then to prove that the scheme is secure given this assumption. The approach for the proof is to *reduce* our scheme to the one we assume is hard. We proceed by showing how, an adversary, which is able to efficiently solve (or break) our scheme, is able to solve the underlying problem. We call such procedure a *reduction* algorithm. For example, the basis for RSA is our inability to efficiently solve the factorization problem.

Definition 2.19 (`FACTORIZE`)**.** Given a composite integer n and $l < n$, find a non-trivial factor of n less than l.

In their paper [RSA78], Rivest et. al. showed that "*[. . .] breaking our system is at least as difficult as factoring n.*" which is precisely the idea behind a reduction algorithm. Another example of a hard problem would be the *discrete logarithm problem* on which the Diffie-Helman key exchange protocol is founded.

Our schemes need to change soon however. In 1997, Peter Shor published a paper [Sho97] showing a quantum algorithm that solves the above problem in polynomial time which, by our definition, is efficient. There are even more examples. As pointed out by Bernstein in [Ber09] it turns out that number theoretic problems seem to be the place where quantum computers improve our algorithms the most. These include Pell's equation [Hal02] or computing the unit group and class group of a number field [SV05], [Hal05]. This might however be due to our better understanding of number theory compared to any other basis for cryptography. We have been

[3] As elegantly stated by Christos Papadimitiou: *"It should not come as a surprise that our choice of polynomial algorithms as the mathematical concept that is supposed to capture the informal notion of 'practically efficient computation' is open to criticism from all sides. [. . .] Ultimately, our argument for our choice must be this: Adopting polynomial worst-case performance as our criterion of efficiency results in an elegant and useful theory that says something meaningful about practical computation, and would be impossible without this simplification."*

studying primes for centuries now but began looking into lattices only in late 19th century where the modern-day definition was introduced by Garrett Birkhoff in 1948 in [Bir48].

Fortunately for us the best quantum computers are still in their infancy. The greatest integer factored using Shor's algorithm was 21 [Mar+12] and it was done in 2012, eleven years after the famous Vandersypen et. al. experiment at IBM [Van+01] in which they factored the number 15. Nonetheless, we shouldn't wait until last minute and should therefore develop new schemes on problems for which we do not have any good (quantum) algorithms. One of the alternatives to number theoretic problems are lattices (among others like code-based or hash-based). Below we present few, most commonly used problems for lattices which will serve as a basis to prove hardness of our schemes.

Definition 2.20 (γ-Shortest Vector Problem (SVP))**.** Given a basis for a lattice $\mathcal{L}$ of dimension n, find a nonzero vector $\boldsymbol{v} \in \mathcal{L}$ of length at most $\gamma \cdot \lambda_1(\mathcal{L})$.

Definition 2.21 (γ-Shortest Independent Vector Problem (SIVP))**.** Like the SVP, except one outputs linearly independent $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_n \in \mathcal{L}$, all of length at most $\gamma \cdot \lambda_n(\mathcal{L})$.

Definition 2.22 (γ-Closest Vector Problem (CVP))**.** Given a basis for a lattice $\mathcal{L}$ of dimension n and a vector $\boldsymbol{t} \in \mathbb{R}^n$, output a nonzero vector $\boldsymbol{v} \in \mathcal{L}$ such that $\text{dist}(\boldsymbol{t}, \boldsymbol{v}) \leq \gamma \cdot \text{dist}(\mathcal{L}, \boldsymbol{t})$.

Definition 2.23 (γ-Bounded Distance Decoding Problem (BDD))**.** Same as γ-CVP, but with the promise that there is a unique solution – i.e., we are given a lattice $\mathcal{L}$ and a vector $\boldsymbol{t} \in R^n$ (within distance $\gamma \cdot \lambda_1(\mathcal{L})$ from the lattice), and we are asked to find a lattice point $\boldsymbol{v} \in \mathcal{L}$ within distance $\gamma \cdot \lambda_1(\mathcal{L})$ from the target.

We know that these problems are in fact NP-hard (for very small approximation factors) [Gen09b], [BP20]. The best know algorithms are variations of LLL lattice reduction algorithm by Lenstra et. al. [LLL82] or Babai's nearest plain algorithm [Bab86].

Remark 2.13. 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 [Ber09]: "*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.

We will also sometimes use an *oracle* in a reduction. An *oracle* is an entity (we often treat is as a black box) capable of solving a problem (even undecidable ones can be used). An oracle appears automatically when we want to prove that a problem reduces to another. When we assume we are capable of solving some underlying problem, we name such algorithm an oracle and do not investigate what the actual implementation is.

Basic cryptographic concepts

As the last part of this section, we will introduce few basic concepts that are often used in cryptography and that we mention at some point in this paper. The notions introduced here are very vague in order to keep this introduction compact. For more precise definitions see for example [KL14] Chapter 3.

The most fundamental of cryptographic concepts is what is called a *one-way function*. This is a function f that is relatively easy to compute for any input but near impossible to invert (i.e., find m given $c = f(m)$). Almost all other constructions are based on such function and this is also how lattice-based cryptography emerged. In 1996 M.Ajtai presented [Ajt96] a one way function based on the problem of solving Short Integer Problems (SIS) which sparked interest of other researchers to work on what we call today the lattice-based cryptography.

Security When we talk about (quantum) *security* of a scheme, we roughly mean that no one (not even quantum computer) is able to break the scheme in a polynomial-time with a non-negligible probability. By "break" we mean that they are not able to obtain a message m from the ciphertext $c \leftarrow \mathsf{Encrypt}(m)$. This, most of the times, boils down to reducing the scheme to an underlying hard problem. Here we present few basic security notions that we mention along the way.

The most common type of security is the *semantic security*. First formally introduced in [GM84], was informally defined as "*[...] whatever an eavesdropper can compute about the cleartext given the cyphertext, he can also compute without the cyphertext.*". This means that whenever we are given a ciphertext c, that encrypts either m_0 or m_1 we are unable to tell which one it was, even if we were to chose between the two. More on that will appear in the chapter on fully homomorphic encryption.

3 Lattice based cryptography

This section will mostly serve as a short survey of the main developments in lattice based cryptography in recent years. We adopt chronological narrative of the sections, starting with the oldest, the GGH scheme from 1997, progressing through works on (ring-)LWE and eventually arriving at the work of Gentry [Gen09a] on ideal lattices and further developments on FHE. For a good survey on the lattice based cryptography, see for example [NS01], [HPS14] chapter 6 or [Pei16].

3.1 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 [GGH97], 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 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[4] 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`.

GGH construction - concretely

The encryption algorithm can be seen in the Table 1.

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 [Ngu99] and broken completely in [NR06]. 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] 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 - [HPS14].

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**[a].

Encrypt:
To encrypt a message $\boldsymbol{m} = (m_1, m_2, \ldots, m_n) \in \mathbb{Z}^n$, choose random small vector $\boldsymbol{r} \in \mathbb{R}^n$ and compute $\boldsymbol{e} = \boldsymbol{mV} + \boldsymbol{r}$.
Decrypt:
Use Babai's algorithm to compute the vector $\boldsymbol{v} \in \mathcal{L}$ closest to $\boldsymbol{e}$. Finally, compute the $\boldsymbol{vW^{-1}}$ to recover $\boldsymbol{m}$.

[a] As an alternative, in [Mic01a], Micciancio suggested to use the Hermite Normal Form (HNF) of $\boldsymbol{B}_{good}$ which essentially provides the worst possible lattice choice for cryptoanalysis, yet making it the most efficient option.

Table 1: GGH encryption algorithm

3.2 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 [Reg05] (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, 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 due to its versatility. It can be used for digital signature schemes (like [SD22]), key exchange schemes (like [Din12]) and encryption schemes which will be mentioned at the end of this section. Arguably the most important contribution is that of laying groundwork for its ring equivalent - the ring-LWE [LPR12] that we will introduce after.

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 3.12 and 3.15.

The problem is parametrized by positive integers n, m[5] 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 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

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

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 $\chi : \mathbb{Z}_q \to \mathbb{R}_{>0}$ 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 3.1. 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 $e_i < 1$ for each $i \in [m]$.

$$\begin{aligned} 0s_1 + 11s_2 + 8s_3 + 2s_4 &\approx_{e_1} 3 \mod 17 \\ 9s_1 + 3s_2 + 7s_3 + 0s_4 &\approx_{e_2} 16 \mod 17 \\ 1s_1 + 15s_2 + 9s_3 + 5s_4 &\approx_{e_3} 16 \mod 17 \\ 0s_1 + 0s_2 + 13s_3 + 5s_4 &\approx_{e_4} 1 \mod 17 \end{aligned}$$

In this case, $\boldsymbol{s} = (7, 13, 12, 16)$. 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 [Reg05] revolves around proving the hardness of LWE. Specifically, that for appropriately chosen q and χ, a *quantum*[6] reduction algorithm exists that approximates worst-case lattice problems. The following is the main result presented in the paper.

Theorem 3.1 ([Reg05], 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 with standard deviation αq. It will be used to sample the errors with some nice and predictable properties. Later, by the end of this section, we will also show that in fact, there is sort of an equivalence between drawing them from continuous and discrete distributions.

The theorem can be rephrased as follows. Imagine that we have an efficient algorithm that solves the $\mathrm{LWE}_{q,\Psi_\alpha}$. 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 ([Mic01b], [Mic08], [Din+03], [MG02]) 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 the whole the approach.

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

3.2.1 Preliminaries

Here we will define essential and convenient definitions and lemmas concerning Gaussian distributions lattices and lattice-related problems. Note that no algebraic number theory is needed for this section as we work exclusively over $\mathbb{R}^n$ and $\mathbb{Z}_q^n$ for q an integer (it is only required for q to be prime in Theorem 3.15, otherwise no assumptions are made).

Gaussians The ideas presented here heavily rely on the distribution of errors. The most important distribution we will be using is the *Gaussian distribution* (also called the normal distribution). For some vector $\boldsymbol{x}$ and $r > 0$, we define the Gaussian function as

$$\rho_r(\boldsymbol{x}) := \exp(-\pi||\boldsymbol{x}/r||^2)$$

where $\exp(y)$ stands for e^y. We will also denote ρ_1 by ρ. By normalizing it, we obtain the n-dimensional Gaussian distribution

$$\Psi_\alpha = \rho_\alpha \Big/ \Big(\int_{\boldsymbol{s}\in\mathbb{R}^n} \rho_\alpha(\boldsymbol{s}) d\boldsymbol{s} \Big) = \rho_\alpha \cdot \alpha^{-n}.$$

We can also define the *discrete Gaussian distribution* $\mathcal{D}_{\mathcal{L},r}$ over a lattice $\mathcal{L}$ (or any countable set A for that matter) by

$$\mathcal{D}_{\mathcal{L},r}(\boldsymbol{x}) = \frac{\rho_r(\boldsymbol{x})}{\rho_r(\mathcal{L})} \tag{3.1}$$

where $\rho_{A,r}(\boldsymbol{x}) = \sum_{\boldsymbol{x}\in A} \rho_r(\boldsymbol{x})$ is an extension of ρ_r to a countable set.

We also need to define *distretization* of a probability density function for our cryptosystem introduced later. For an arbitrary probability density function $\phi : \mathbb{R} \to \mathbb{R}_{>0}$ and an integer $q \geq 2$, we define its discretization $\bar{\phi} : \mathbb{Z}_q \to R_{>0}$ as the discrete probability distribution obtained by sampling from Ψ, multiplying by q, and rounding to the closest integer modulo q. More formally,

$$\bar{\phi}(i) = \int_{(i-1/2)/q}^{(i+1/2)/q} \phi(x) dx \tag{3.2}$$

Lastly, following the definition given in [MG02], we define the *statistical distance* as follows.

Definition 3.2 (Statistical distance)**.** Let X, Y be two random variables over a countable set A. The statistical distance between them is defined as

$$\Delta(X,Y) = \sum_{a\in A} \Big| \Pr\{X = a\} - \Pr\{Y = a\} \Big|.$$

Similarly, if φ_X, φ_Y are the *probability density* functions over $\mathbb{R}^n$ of X and Y respectively, then

$$\Delta(\varphi_X, \varphi_Y) = \int_{\boldsymbol{s}\in\mathbb{R}^n} \Big| \varphi_X(\boldsymbol{s}) - \varphi_Y(\boldsymbol{s}) \Big| d\boldsymbol{s}.$$

It can be interpreted as a measure of how much apart two distributions are from each other. It is indeed not hard to show that this definition satisfies the properties of a distance function. We will be using this tool to measure how our elements are behaving, for example, in Lemma 3.15.

Smoothing parameter One of the most important concepts invoked in the paper is the so called *smoothing parameter* η of a lattice $\mathcal{L}$. It was first introduced in [MR07] and has proven useful over the course of the years. The definition is quite technical as it makes use of the dual rather than the lattice itself. The intuition behind it could be obtained by imagining the following. Pick an element from your favourite lattice $\mathcal{L}$. If one now picks a "noise" vector from a Gaussian distribution with width at least as large as the smoothing parameter, the resulting distribution is very close to uniform. The precise definition is as follows.

Definition 3.3 (Smoothing parameter)**.** For an n-dimensional lattice $\mathcal{L}$ and positive real $\epsilon > 0$, we define the smoothing parameter $\eta_\epsilon(\mathcal{L})$ to be the smallest s such that $\rho_{1/s}(\mathcal{L}^\vee \setminus \{\mathbf{0}\}) \leq \epsilon$.

In other words, $\eta_\epsilon(\mathcal{L})$ is the smallest s such that a Gaussian measure scaled by $1/s$ on the dual lattice $\mathcal{L}^\vee$ gives all but $\epsilon/(1+\epsilon)$ of its weight to the origin - [Reg05]. Note that we need to use the dual in the definition because we cannot really make sense of the "longest" element of the lattice $\mathcal{L}$ as there is none. Instead, we look at the shortest element of the dual $\mathcal{L}^\vee$ which we know exists. ϵ is our desired tolerance and as the name suggests, it is usually taken to be very small (in our cases, a negligible function of n). It is also worth noting that $\eta_\epsilon(\mathcal{L})$ is a continuous function of ϵ.

In the Figure 4, there are plotted four Gaussians. The first one has very small deviation and assigns most of its values to the origin. The second one, with the deviation $\sqrt{10}$ times the size of the first one, looks a bit more *smooth* and continuous. Finally, the last distribution is almost uniform or "flat". For an appropriate choice of ϵ, the smoothing parameter will lay somewhere between those four values of r.

Proposition 3.4 (Sum of Gaussians)**.** *If $X_1, X_2, \cdots, X_n$ are mutually independent normal random variables with means $\mu_1, \mu_2, \cdots, \mu_n$ and variances $\sigma_1, \sigma_2, \cdots, \sigma_n$, then the linear combination*

$$Y = \sum_{i \in [n]} c_i X_i$$

follows the normal distribution

$$\mathcal{N}\Big(\sum_{i \in [n]} c_i \mu_i, \sum_{i \in [n]} c_i^2 \sigma_i^2\Big).$$

Problems We will now define problems related specifically to the LWE. First, let us define the distribution induced by LWE.

Definition 3.5 (LWE distribution)**.** Let $q \geq 2$ be some integer, and let $\chi : \mathbb{Z}_q \to \mathbb{R}_{>0}$ 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 $\mathbb{Z}_q^n \times \mathbb{Z}_q$.

There are two types of problems we can define over LWE - the *search* and *decision*. We will prove at the end of this section that they are in fact equivalent, i.e. there exists a reduction between them. Search-LWE is asking us to find $\boldsymbol{s}$ from a list of samples distributed according to $A_{\boldsymbol{s},\chi}$ for some unknown χ. Similarly, the *decision* asks us to distinguish samples $(\boldsymbol{a}, b)$ that were

draw from either $A_{\boldsymbol{s},\chi}$ or the uniform distribution. Although not present in the original paper, we define them formally here.

Definition 3.6 (Search-LWE)**.** Given access to arbitrarily many samples from the $A_{\boldsymbol{s},\chi}$, find $\boldsymbol{s}$.

Definition 3.7 (Decision-LWE)**.** Given access to arbitrarily many samples from the $A_{\boldsymbol{s},\chi}$ and the same amount of samples from the uniform distribution, distinguish (with non-negligible advantage) between them.

The following are used as an intermediate step in our reduction to SIVP. The acronym DGS stands for *discrete Gaussian sampling* problem and is concerned exactly with that – to find a sample from the normal distribution over our lattice. Formally:

Definition 3.8 (DGS)**.** Given an n-dimensional lattice $\mathcal{L}$ and a number $r \geq \sqrt{2}n \cdot \eta_\epsilon(\mathcal{L})/\alpha$, output a sample from $\mathcal{D}_{\mathcal{L},r}$.

Along with the decision version of the SVP problem:

Definition 3.9. An instance of GapSVP_γ is given by an n-dimensional lattice $\mathcal{L}$ and a number $d > 0$. In YES instances, $\lambda_1(\mathcal{L}) \leq d$ whereas in NO instances $\lambda_1(\mathcal{L}) > \gamma(n) \cdot d$.

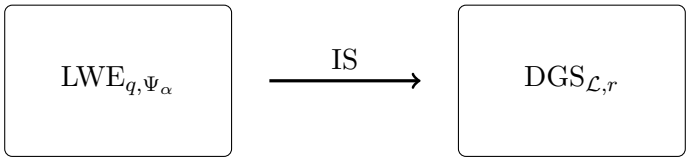
3.2.2 Hardness of search LWE

In this section we will base the hardness of LWE on some hard lattice problems like SIVP following the approach in [Reg05]. 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 what we want to prove, that being able to solve $\text{LWE}_{q,\chi}$ 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 another. From this point on, a probabilistic algorithm which solves a given $\text{LWE}_{q,\chi}$ instance, will be called an LWE-*oracle* (or, when there is no confusion, simply an oracle).

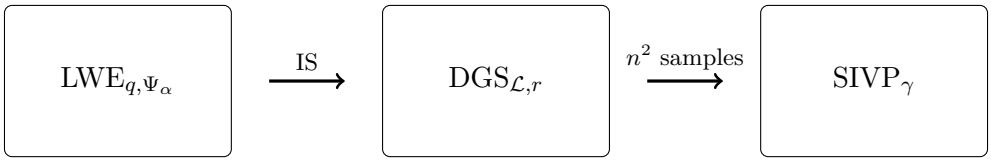
The high-level version of the proof is as follows. Let us assume that we have such an oracle that solves $\text{LWE}_{q,\chi}$ on a lattice $\mathcal{L}$ just like in the assumption. The procedure is based on the so called *iterative step* (IS). It is repeatedly used to reduce our LWE problem to the 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 3.8). 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 3.1.



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 $\mathtt{DGS}_{\mathcal{L},r}$ and we are done.

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 $\gamma = \tilde{O}(n/\alpha)$.



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 [Reg05]) is the core of the hardness results of (search) LWE. Given an oracle for 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 3.15). 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 3.10. *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 $DGS_{\sqrt{2}n\cdot\eta_\epsilon(\mathcal{L})/\alpha}$.*

Proof. The proof follows in a straight-forward manner by the Lemma 3.11. We sample a polynomial amount of elements from the $\mathcal{D}_{\mathcal{L},r}$ for r large enough[7]. 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 3.10 relies mainly on the following algorithm. This is slightly altered Lemma 3.3 in [Reg05].

Lemma 3.11 (Iterative Step)**.** *Let $\epsilon = \epsilon(n)$ be a negligible function, $\alpha > 0$ real, and $q \geq 2$ be an integer. Assume that we have access to an oracle that solves $LWE_{q,\Psi_{\leq\alpha}}$ given a polynomial number*

[7] By Claim 2.13 and Lemma 3.2 of [Reg05], we can efficiently draw samples that are exponentially close to $\mathcal{D}_{\mathcal{L},r}$ if $r > 2^{2n}\lambda_n(\mathcal{L})$.

of samples. Then, there exists an efficient quantum algorithm that, given any n-dimensional lattice $\mathcal{L}$, a number $r > \sqrt{2}q\eta_\epsilon(\mathcal{L})$, and a polynomial list of samples from the discrete Gaussian distribution $\mathcal{D}_{\mathcal{L},r}$, produces a sample from $\mathcal{D}_{\mathcal{L},r\sqrt{n}/(\alpha q)}$.

Proof. As mentioned before, the algorithm consists of two parts. The first part is presented in 3.12, where, given an oracle W and samples from $\mathcal{D}_{\mathcal{L},r}$, solves $\texttt{CVP}_{\mathcal{L}^\vee,\alpha q/(\sqrt{2}r)}$. The second part - the quantum algorithm 3.13 - when given an oracle that solves $\texttt{CVP}_{\mathcal{L}^\vee,\alpha q/(\sqrt{2}r)}$, yields us a sample from $\mathcal{D}_{\mathcal{L},r\sqrt{n}/(\alpha q)}$. □

Pictographically, two iterations of the algorithm are presented in Figure 5.

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 3.12 (Step 1 - classical)**.** *Let $\epsilon = \epsilon(n)$ be a negligible function, $\alpha > 0$ real, and $q \geq 2$ be an integer. Assume that we have access to an oracle that solves $LWE_{q,\Psi_{\leq\alpha}}$ given a polynomial number of samples. 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 list of samples from $D_{\mathcal{L},r}$, solves $\texttt{CVP}_{\mathcal{L}^\vee,\alpha q/(\sqrt{2}r)}$.*

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 $A_{\boldsymbol{s},\Psi_{\leq\alpha}}$ 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 $\boldsymbol{x}$. We say an algorithm solves $\texttt{CVP}_{\mathcal{L},d}$ if, given any point $\boldsymbol{x} \in \mathbb{R}^n$ within distance d of $\mathcal{L}$ it gives $\boldsymbol{y} \bmod q \in \mathbb{Z}_q^n$, the coefficient vector of the closest vector to $\boldsymbol{x}$ reduced modulo q[8].

The following proof is a simplified version of the proof of Lemma 3.11 from [Reg05] 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 $\boldsymbol{a} = \mathcal{L}^{-1}\boldsymbol{v} \bmod q$ - 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 [Bab86]) to obtain the full, unreduced solution. For the details, see [Reg05] Lemma 3.5. We now output

$$(\boldsymbol{a}, \langle \boldsymbol{x}, \boldsymbol{v} \rangle + e' \bmod q) \tag{3.3}$$

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

To see this, first note that $\boldsymbol{a}$ is a uniform sample ([Reg05], Claim 3.8). This is because $r \geq \eta_\epsilon(\mathcal{L})$ and by our discussion in 3.3, the $\mathcal{D}_{\mathcal{L},r}$ will behave like a uniform distribution. Let us now fix $\boldsymbol{a}$ and consider $\boldsymbol{y} = \boldsymbol{x} - \boldsymbol{e}$. Then

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

[8]This is technically a solution to $\texttt{CVP}^{(q)}_{\mathcal{L},d}$ - regular CVP but reduced modulo q. However as we will see later in the proof, there exists a reduction from one to another.

Focusing now on the second term, we observe that

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

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

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

Proposition 3.13 (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 $\mathit{CVP}_{\mathcal{L}^\vee, d}$, outputs a sample from $D_{\mathcal{L}, \sqrt{n}/(\sqrt{2}d)}$.*

Lastly, we can finish the proof of Theorem 3.1 by presenting the following proposition that provides reduction from `SIVP` and `GapSVP` to `DGS`.

Proposition 3.14. *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.*

3.2.3 Pseudorandomness of LWE

In this section we will prove that the search version of LWE - Definition 3.6 - to the decision version - Definition 3.7. 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 3.15 ([Reg05], 4.2 - Decision to Search)**.** *Let $n \geq 1$ be some integer, $2 \leq q \leq \mathit{poly}(n)$ be a prime, and χ be some distribution on $\mathbb{Z}_q$. Assume that we have access to procedure `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 `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 $b = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e$. 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 possible k in polynomial time. □

3.2.4 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 $\bar{\Psi}_{\alpha(n)}$ (the discretized Gaussian distribution Ψ_α) where $\alpha(n) = o(1/(\sqrt{n} \log n))$ (recall from Section 2.4 that it means $\lim_{n\to\infty} \alpha(n) \cdot \sqrt{n} \log n = 0$). The scheme is presented in Table 2.

Example 3.2. Almost exactly like in the Example 3.1, 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**

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

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.

Table 2: LWE encryption scheme

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 $\langle \boldsymbol{a}, \boldsymbol{s} \rangle = 108 \equiv 6 \bmod 17$ which gives us $b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle = 0 - 6 \equiv 11 \bmod 17$. 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 4.1. The following is a somewhat simpler version of Claim 5.2 in [Reg05].

Lemma 3.16 (Correctness)**.** *For the above choice of parameters and e following the $\bar{\Psi}_\alpha$ distribution we have*

$$\Pr_{e \sim \bar{\Psi}_\alpha} \left[|e| < \left\lfloor \frac{q}{2} \right\rfloor / 2 \right] > 1 - \delta(n) \tag{3.4}$$

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

Proof. Note that if not for the error term, the decryption would always be correct. Hence we need to pay attention to the case where the error term is greater than $q/4$. Since there are m such terms, each with standard deviation of $\sigma^2 = \alpha q$, we can use Proposition 3.4 to show that if $X_i \sim \mathcal{N}(0, \alpha q)$ are i.i.d. for $i \in [m]$, then the sum

$$\sum_{i \in [m]} X_i \sim \mathcal{N}(0, \sum_{i \in [m]} \sigma^2) = \mathcal{N}(0, m \cdot \sigma^2)$$

is distributed with standard deviation equal to $\sqrt{m}\alpha q < q/\log(n)$ and the probability that such sample is greater than $q/4$ is negligible. □

An example of a normal distribution with standard deviation $\sigma^2 = q/\log n$ and mean 0 can be seen in Figure 6. The shaded region corresponds to $|e| > q/4$. At the begining, for n not as big, we can notice that the area does not necessarily correspond to an intuitive notion of "negligible". However, the definition concerns the asymptotic behavior of n and indeed, we can see the area becomes quite small. One might even call it "negligible".

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

Lemma 3.17. *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, $|\sum_{i\in S} e_i| < \lfloor \frac{q}{2} \rfloor/2$ 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 3.18 ([Reg05], 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 [MR09].

Epilogue

[Krzys: do last if theres time] Until the day of writing this paper, LWE is one of the most influential schemes that can be used for post-quantum cryptographic schemes. It was used as a basis for schemes like the one introduced in [PVW07] (along with an oblivious transfer protocol), [LP10] or [Pei09]. However, arguably the most important contribution, was that of laying groundwork for the *ring*-LWE scheme introduced in the next section. We will now present few alternatives to the results and proofs presented here.

3.3 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 $\tilde{O}(n^2)$[9]. 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 $\boldsymbol{a}_i = (a_{\mu(1)}, a_{\mu(2)}, \ldots, a_{\mu(n)})$ is a permutation by some μ 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 $\boldsymbol{a}_3$ could have the form $(4, 5, 16, 1)$. 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 $R_q := R/qR$. In this exposition, to simplify some steps, R is taken to be a *cyclotomic* ring – i.e. $R_q = \mathbb{Z}_q[x]/(x^n + 1)$ for $n = 2^k$ which turns out to yield much simpler proofs for 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" [LPR12]. 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 [Mic07] that was followed up by other works but these results are not presented in this paper) and followed the heuristic approach behind the NTRU[10] cryptosystem [Buh98]. 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 3.15 whose definition will appear 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. This is finally where we can use all the theory developed in the Section 2.3.

Somewhat analogous to the previous section, this one is also split into two main parts. First part - Section 3.3.2 - focuses on the hardness of the search version of the RLWE. The approach is identical to the one presented for the standard LWE. However, one needs to pay attention to details that are implied by the shift to R like for example the shape of the error distribution under the canonical embedding. Fortunately for us, the quantum part of the reduction can be adopted almost as is and so, we will only mention it briefly. The second part - Section 3.3.3 - deals with pseudorandomness of the RLWE distribution and thus proving the equivalence between the search and decision versions. This one will require much more insight into the algebraic number theory compared to any other part of this paper and so, we will discuss it in much more detail. Before we dwell any further, we need to establish some terminology and useful lemmas. This is done in the following section.

[9] 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.

[10] As mentioned by Peikert in his survey: "The meaning of the acronym NTRU is somewhat mysterious; plausible candidates include "*N*th degree *tru*ncated polynomial ring" and "Number Theorists 'R' Us."" - [Pei16]

3.3.1 Preliminaries

In this section we will formally introduce the underlying ring structure that will be used throughout this section. We will try to spell out in more detail facts and results that are used implicitly in the original [LPR12] providing few examples along the way. Most of those results will mainly be useful for the second part which is the decision to search reduction. In the first part, the hardness of search version, holds for any number field, not necessarily cyclotomic.

Let us therefore fix a cyclotomic number field K of degree n. More precisely, if we take $m = 2^k$ for some $k \geq 2$ and set $n = \phi(m) = m/2 = 2^{k-1}$, then by Corollary 2.15 we obtain

$$K := \mathbb{Q}[x]/\Phi_m(x) = \mathbb{Q}[x]/(x^n + 1) \cong \mathbb{Q}(\zeta),$$

where $\zeta = \exp(2\pi\sqrt{-1}/m)$ is the m-th root of unity. Finally, we define

$$R := \mathcal{O}_K = \mathbb{Z}[\zeta] \tag{3.5}$$

where the equality holds by Proposition 2.16 and therefore is generated by $\{1, \zeta, \ldots, \zeta^{n-1}\}$. Recal at this point the notion of the dual lattice in a number field

This ring however is slightly too big for our desire. Analogously to the LWE case we therefore consider the coset obtained by reducing all elements modulo some integer q. Let us now consider two specific cases.

Ideal Lattices

We can finally introduce the ideal lattices used in the construction. We will first take a small detour into the theory of ideals and then use the canonical embedding to obtain lattices. For a good discussion on the *different* idal, see Keith Conrads paper.

[Krzys: add discussion about fractional ideals and the different ideal]

Chinese Remainder Theorem Lastly, we will present the Chinese Remainder Theorem for rings and few related propositions that will be useful for our reductions in the two following sections.

Proposition 3.19 (Chinese Remainder Theorem)**.** *Let $\mathcal{I}_1, \ldots, \mathcal{I}_r$ be pairwise coprime ideals in R, and let $\mathcal{I} = \prod_{i\in[n]} \mathcal{I}_i$. Then there is an isomorphism of rings*

$$R/\mathcal{I} \to \bigoplus_{i\in[n]} (R/\mathcal{I}_i).$$

This is slightly generalized Theorem II.4.12 from J.Top lecture notes for algebraic structures and the proof is almost identical with the replacement of two coprime ideals with n-many.

The proofs to the following propositions can be found in [LPR12].

Proposition 3.20. *Let $\mathcal{I}$ and $\mathcal{J}$ be ideals in R. There exists $t \in \mathcal{I}$ such that the ideal $t \cdot \mathcal{I}^{-1} \subseteq R$ is coprime to $\mathcal{J}$.*

Proposition 3.21. *Let $\mathcal{I}$ and $\mathcal{J}$ be ideals in R and let $t \in \mathcal{I}$ be such that $t \cdot \mathcal{I}^{-1}$ is coprime with $\mathcal{J}$. Then the function $\theta_t : K \to K$ defined as $\theta_t(u) = t \cdot u$ induces an isomorphism from $R/\mathcal{J}R$ to $\mathcal{I}R/\mathcal{I}\mathcal{J}R$.*

Why cyclotomic? As mentioned earlier, we wish to fix the degree of our cyclotomic polynomial to a power-of-two. This leads to greatly simplified proofs of some results like for example search-to-decision reduction in section 3.3.3. Nonetheless, in places where it is not necessary, we will be using arbitrary number fields, sometimes not even cyclotomic. We will now list few desirable properties of cyclotomic rings (also those of power of two) that will be useful for us later on. Let us fix m, n and k to some positive integers.

- Under the canonical embedding, the cyclotomic ring $R = \mathcal{O}_K$ for $\Phi_m(x) = x^{\varphi(m)} - 1$ embeds as a lattice which in general, is not self-dual (this is only the case for $\mathbb{Z}^n$). Instead, its dual lattice corresponds to a fractional ideal $R^\vee \subset K$ such that $R \subseteq R^\vee \subseteq m^{-1}R$. In the case where $n = \varphi(m) = 2^{k-1}$ these two are actually equivalent, namely $R^\vee = n^{-1}R$.
- Polynomial arithmetic modulo $\Phi_m(X)$ can be performed very efficiently using slightly adjusted, classical n-dimensional Fast Fourier Transform - [LPR13], [Lyu+08].
- In general, the smaller the shortest vector $\lambda_1(\mathcal{L})$ is, the less secure the system. Cyclotomic fields have relatively large $\lambda_1(R^\vee)$ - [PRS17].
- In fact, for $n = 2^k$, we have $\lambda_n(R^\vee) = 1/\sqrt{n}$
- The cyclotomic number field is a CM-field which means that we do need to differentiate between real and complex embeddings since there are only exactly $n/2$ of the complex ones. As one can imagine this can simplify some technical steps in proofs and definitions.
- Cyclotomic number fields have Galois groups that "act transitively" on the ideals of the ring of integers. We will explain this in more details in the Subsection 3.3.3.

Unfortunately, cyclotomic fields with degree of a power-of-two are quite rare and restrictive. Imagine that our system is deemed insecure for some large $n = 2^k$. It might so happen that the next power of two is completely impractical when implemented. We should be able to find something inbetween the two instead.

The RLWE distribution We now formally introduce the R-LWE distribution and related problems.

Definition 3.22 (Ring-LWE Distribution)**.** For $s \in R_q$ (the "secret") and an error distribution ψ over R_q, a sample from the ring-LWE distribution $A_{s,\psi}$ over $R_q \times R_q$ is generated by choosing $a \leftarrow R_q$ uniformly at random, choosing $e \leftarrow \psi$, and outputting $(a, b = a \cdot s + e)$.

Definition 3.23 (Ring-LWE, Search))**.** Let Ψ be a family of distributions over R_q. The search version of the ring-LWE problem, denoted R-LWE$_{q,\Psi}$, is defined as follows: given access to arbitrarily many independent samples from $A_{s,\psi}$ for some arbitrary $s \in R_q$ and $\psi \in \Psi$, find s.

Distribution From this point onwards, for $\alpha > 0$, we will denote by $\Psi_{\leq\alpha}$ the set of all elliptical Gaussian distributions $\mathcal{D}_{\boldsymbol{r}}$ (over R_q) where each parameter $r_i \leq \alpha$.

3.3.2 Hardness of search Ring-LWE

We can finally discuss the main results from the [LPR12]. This section focuses on the quantum reduction from R-LWE$_{q,\Psi_{\leq\alpha}}$ to the K-DGS$_\gamma$ – the approximate (to within γ) discrete Gaussian sampling problem (the definition for an arbitrary number field K is created by simply replacing the lattice $\mathcal{L}$ with an ideal $\mathcal{I}$ in Definition 3.8, and so, the samples are defined to be from $\mathrm{D}_{\mathcal{I},r}$.). Throughout this section, let K be an arbitrary number field of degree n and $R = \mathcal{O}_K$ its ring of integers. We will fix q to denote an integer (not necessarily prime). We note at this points that all the results from this section apply to general number fields, not only cyclotomic but one might keep that example in mind for concreteness.

The following is an equivalent of Theorem 3.10 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 3.24. *Let K and R be as above. Also let $\alpha = \alpha(n) > 0$, and $q = q(n) \geq 2$ be such that $\alpha q > 2\sqrt{n}$. For some negligible $\epsilon = \epsilon(n)$, there is a probabilistic polynomial-time quantum reduction from K-DGS$_\gamma$ to R-LWE$_{q,\Psi_{\leq\alpha}}$ where $\gamma > 0$ is usually taken[11] to be $\gamma = \omega(\log n)$.*

And just as before, the proof relies on the (adjusted) *iterative step* (IS) in parallel with the previous section. It presents as follows.

Lemma 3.25 (Iterative Step)**.** *Let $\epsilon = \epsilon(n)$ be a negligible function, $\alpha > 0$ real, and $q \geq 2$ be an integer. Assume that we have access to an oracle that solves LWE$_{q,\Psi_{\leq\alpha}}$ given a polynomial number of samples. Then, there exists an efficient quantum algorithm that, given a fractional ideal $\mathcal{I}$ in K, a number $r > \sqrt{2}q \cdot \eta_\epsilon(\mathcal{I})$ and a (polynomial) list of samples from the discrete Gaussian distribution $D_{\mathcal{I},r}$, produces a sample from $D_{\mathcal{I},r\cdot\gamma/(\alpha q)}$.*

It is easy to notice that the statement is almost identical to the Lemma 3.11. The place of a n-dimensional lattice $\mathcal{L}$ has been taken up by a fractional ideal $\mathcal{I}$ (which, when embedded, lives in $\mathbb{C}^n$ as an n-dimensional lattice). The proof is also almost intact. We will use Lemma 3.26 and slightly adjusted Lemma 3.13 from LWE to create a sequence of discrete Gaussians with decreasing radii to obtain the solution to K-DGS$_\gamma$.

Lemma 3.26 (Step 1 - classical)**.** *Let $\epsilon = \epsilon(n)$ be a negligible function, $\alpha > 0$ real, and $q \geq 2$ be an integer with known factorization. Let $\mathcal{I}$ be a fractional ideal in K, and let $r \geq \sqrt{2}q\cdot\eta_\epsilon(\mathcal{I})$. Given polynomial list of samples from the discrete Gaussian distribution $D_{\mathcal{I},r}$, there is a probabilistic polynomial-time (classical) reduction from $\mathtt{CVP}_{\mathcal{I}^\vee,d}$ to R-LWE$_{q,\Psi_{\leq\alpha}}$, where $d = \alpha q/(\sqrt{2}r)$.*

Remark 3.3. In the original statement in [LPR12], CVP is replaced by BDD to within the same distance. These statements are very similar with a simple difference that BDD is concerned with finding *any* lattice vector within d whereas CVP wants to find the *closest* vector. Since the distance d is less than $\lambda_1(\mathcal{L})/2$, BDD will find the closest, unique vector to $\mathcal{L}$. Hence, in this case, they are equivalent and we use CVP to keep it consistent with previous section.

Proof. Just as before, it is enough to show reduction to $\mathtt{CVP}^{(q)}_{\mathcal{I},d}$ by a similar argument. The high-lever reduction presents now as follows. We are given a $\mathtt{CVP}^{(q)}_{\mathcal{I},d}$ instance $y = x + e$ where

[11] See the discussion just before Section 4.1 in [LPR12].

$x \in \mathcal{I}^\vee$ and $\|e\|_\infty \leq d$. We are also granted access to (as many as necessary) samples from the discrete Gaussian over $\mathcal{I}$ and standard deviation r as well as an oracle for R-LWE. We will then try to relate the $y = x + e$ to a sample from $\mathrm{A}_{s,\psi}$ where $\psi \in \Psi_{\leq\alpha}$ so that we can use the oracle to produce the solution to R-LWE and ultimately to the $\mathtt{CVP}^{(q)}_{\mathcal{I},d}$. We will proof the lemma using three steps.

1. Linking. First, find an element $t \in \mathcal{I}$ such that $t \cdot \mathcal{I}^{-1}$ and (q) are coprime. By Proposition 3.21, such t exists and is efficiently computable but we do not need to know the exact form of t. We can now define our input to the R-LWE oracle. For each sample $z \leftarrow \mathcal{D}_{\mathcal{I},r}$ set $e' \leftarrow \mathcal{D}_{\alpha/\sqrt{2}}$ and

$$\begin{aligned} a &= \theta_t^{-1}(z \bmod q\mathcal{I}) \in R_q \quad \text{and} \\ b &= z \cdot y + e'. \end{aligned}$$

   Recall that θ_t is a bijection and we can compute it efficiently - [LPR13].

2. Correctness. The correctnes of this transformation is proven as a separate lemma for better readibility, Lemma 3.27.

3. Error distribution. We also need to pay attention to how the error distribution $\mathcal{D}_r$ is behaving in this case. This is slightly technical and we refer the reader to the original proof in Lemma 4.8 in [LPR12].

□

Lemma 3.27. *Let y be the $\mathtt{CVP}^{(q)}_{\mathcal{I}^\vee,d}$ instance given to the reduction above, where $y = x + e$ for some $x \in \mathcal{I}^\vee$ and $\|e\|_\infty \leq d$. Each pair (a, b) produced by the Linking procedure has distribution $A_{s,\psi}$ (up to negligible statistical distance), for $s = \theta_t(x \bmod q\mathcal{I}) = t \cdot x \in R_q$ and some $\psi \in \Psi_{\leq\alpha}$.*

Proof. First, we need to show that the designed $a \in R_q$ is within negligible statistical distance of the uniform distribution. This follows by our choice of parameters, namely, that r was chosen such that $r \geq q\eta_\epsilon(\mathcal{I})$ and we have taken $z \leftarrow \mathcal{D}_{\mathcal{I},r}$ so just like argued in the proof of the Lemma 3.12, z will be within negligible statistical distance of uniform. Since θ_t is a bijection, $a = \theta_t^{-1}(z \bmod q\mathcal{I})$ will be as well.

Now condition on a fixed value of a. Let us analyze the $b = z \cdot y + e' = z \cdot x + z \cdot e + e'$ component. Consider the first term – the $z \cdot x$. By definition $a = \theta_t^{-1}(z)$ so we obtain $z = t \cdot a$. By our the choice of $s = \theta_t(x) = t \cdot x$ we can now compute

$$z \cdot x = a \cdot t \cdot x = a \cdot s$$

We are now left with the $ze + e'$ term. We need to show that it is within negligible statistical distance of the elliptical Gaussian $\mathcal{D}_r$. As mentioned earlier, we omit that part and the reader is redirected to the Lemma 4.8 in [LPR12] □

3.3.3 Pseudorandomness of Ring-LWE

We also want to show that the ring-LWE distribution is pseudorandom – i.e. samples from the R-LWE distribution are indistinguishable from truly random (uniform) ones. This is encapsulated as the Theorem 3.28. Let us for convenience denote by $R\text{-DLWE}_{q,\Psi}$ the *decision* R-LWE problem, where instead of *searching* for the secret s, we want to distinguish (or decide) between a sample from $A_{s,\Psi}$ and a uniform one.

Just like in the LWE case, we will only prove the reduction mentioned above. One could also prove the worst-case to average-case reduction as is indeed done in the original. We will not be concerned with that right here for compactness reasons as it requires much more investment in the theory of Gaussian distributions.

From now on, denote by $\zeta = \zeta_m$ the m-th root of unity. $K = \mathbb{Q}(\zeta)$ is the m-th cyclotomic number field and denote by $n = \varphi(m)$. On top of that, fix a prime q such that $q \equiv 1 \bmod m$.

The main theorem presents as follows.

Theorem 3.28. *Let R and q be as above and let $\alpha q \geq \eta_\epsilon(R^\vee)$ for some negligible $\epsilon = \epsilon(n)$. Then there is a reduction from $R\text{-LWE}_{q,\Psi_{\leq\alpha}}$ to $R\text{-DLWE}_{q,\Psi_{\leq\alpha}}$.*

The reduction follows from search-R-LWE to decision-R-LWE and in high-level (in our case), it consists of 2 steps presented in Figure 8 and described in more detail below. The idea behind the reduction is quite similar to the one for standard LWE but we need to pay attention to details of our construction.

1. $\text{LWE}_{q,\Psi}$ to $\mathfrak{q}_i\text{-LWE}_{q,\Psi}$. Due to each of the coefficients of $a \in R_q$ being dependent on every other. This is very beneficial to the efficiency but poses a problem to us. Namely, if we apply the same transformation as in Lemma 3.15, altering one coefficient alters all of the other ones as well. We can resolve this issue by looking at the problem relative to only one coordinate which is in turn possible thanks to the automorphisms of R_q for our choice of parameters as explained in the next paragraph. This is why we reduce the LWE to the so called $\mathfrak{q}_i\text{-LWE}_{q,\Psi}$ of which the definition will follow.

2. $\mathfrak{q}_i\text{-LWE}_{q,\Psi}$ to $\text{DLWE}_{q,\Psi_{\leq\alpha}}$. Having the way to reduce our problem to only one coordinate, we can apply something alike the transformation from Lemma 3.15. This requires introducing a bit more notation and careful handling of the samples, but if one understands what happens in the precursor, this should be a straight-away translation to the R_q setting. We note that this reduction is only for the worst-case choice of s and we will not present the reduction to the average-case.

What do we mean by "act transitively"? Although not present in the original, we now explain the implicit details of the proof below. Most importantly, what do we mean by "act transitively" and why it is the case.

Recall from section 2.3 that the Galois group of a cyclotomic number field $\mathbb{Q}(\zeta_m)$ is isomorphic to $\mathbb{Z}_m^* = \{k : 1 \leq l < m, \gcd(l, m) = 1\}$. For our choice of $q \equiv 1 \bmod m = 2n = 2^k$, this group is actually just

$$\mathbb{Z}_m^* = \{l \in \mathbb{Z} : l < 2^k, \gcd(l, 2^k) = 1\} = \{l < 2n : l \text{ is odd}\},$$

the set of odd integers less than $2n$. When counted, there are precisely n of them. The elements of the group are automorphisms τ_i such that $\tau_i(\zeta) = \zeta^i$ and the elements of $\mathbb{Q}$ are fixed. Note that, thanks to that isomorphism, we have $\tau_i^{-1} = \tau_{i^{-1}}$.

Let us look now at the q and denote by $\mathbb{Z}_q = \mathbb{Z}_q^*$ the group modulo q under multiplication. Since we picked $q \equiv 1 \bmod 2n$ we can take[12] for simplicity that $q = 2n + 1$. Then we know that $\varphi(q) = 2n$ – there are exactly $2n$ integers less than q and coprime with q. Therefore there exists some $z \in \mathbb{Z}_q$ such that $z^{2n} \equiv 1 \bmod q$ and $z^l \neq 1 \bmod q$. Such z is the $2n$-th primitive root of unity modulo q. Long time ago, Euler proved that for any prime q, there are no nontrivial (i.e. other than 1 and -1) square roots modulo q. Therefore we must have

$$z^{2n} \equiv 1 \bmod q \implies z^n \equiv -1 \bmod q \implies z^n + 1 \equiv 0 \bmod q$$

Recall also, from the section about cyclotomic fields, that this means that the set of all conjugates of z are those z^r such that the $\gcd(2n, r) = 1$ which are precisely all the odd integers $< 2n$. Hence, the set

$$\{z, z^3, z^5, \ldots, z^{2n-1}\}$$

is the set of all the $2n$-th roots of unity and all of those elements are actually in $\mathbb{Z}_q$. We conclude that the polynomial $x^n + 1$ "splits completely" into linear factors modulo q:

$$x^n + 1 = (x - z)(x - z^3) \cdot \ldots \cdot (x - z^{2n-1}) = \prod_{\substack{r < 2n \\ r \text{ is odd}}} (x - z^r).$$

By the CRT (Proposition 3.19), there is an isomorphism

$$R_q = \mathbb{Z}_q[x] \Big/ \prod_{\substack{r < 2n \\ r \text{ is odd}}} (x - z^r) \to \mathbb{Z}_q[x]/(x - z) \times \mathbb{Z}_q[x]/(x - z^3) \times \ldots \times \mathbb{Z}_q[x]/(x - z^{2n-1}).$$

Since our underlying ring is the ring of integers (it is a Dedekind domain) and $(x - z^r)$ are prime (for a, b odd, if $z^a \cdot z^b \in (z^r)$ then either a or b must thus be in (z^r) because a sum of two odd integers is even) for all odd $r < 2n$, they are actually maximal. Therefore, we have $\mathbb{Z}_q[x]/(x - z^r) \cong \mathbb{Z}_q$. [Krzys: im not sure if that is usefull at all]

Finally tying those two concepts together, the Galois group and $\mathbb{Z}_q$, we

Search to Worst-Case Decision We begin with the reduction of the LWE to $\mathfrak{q}_i$-LWE distribution relative to only one $\mathfrak{q}_i$ arbitrary prime ideal.

Definition 3.29. The $\mathfrak{q}_i$-LWE$_{q,\Psi}$ problem is: given access to $\mathrm{A}_{s,\Psi}$ for some arbitrary $s \in R_q$ and $\psi \in \Psi$, find $s \bmod \mathfrak{q}_i R$.

Lemma 3.30 (LWE to $\mathfrak{q}_i$-LWE, Lemma 5.5 [LPR12])**.** *Suppose that the family Ψ is closed under all the automorphisms of K, i.e., $\psi \in \Psi \Rightarrow \tau_k(\psi) \in \Psi$ for every $k \in \mathbb{Z}_m^*$. Then for every $i \in \mathbb{Z}_m^*$, there is a deterministic polynomial-time reduction from LWE$_{q,\Psi}$ to $\mathfrak{q}_i$-LWE$_{q,\Psi}$.*

[12] The case where $q = c(2n + 1)$ for some $c > 1$ is a bit more technical but the following proof works in general for $q = 2n + 1$.

Proof. By assumptions, we are given access to an $\mathfrak{q}_i$-LWE oracle along with n field automorphisms τ_k that "act transitively" on the prime ideals $\mathfrak{q}_i$ thanks to the underlying field being a cyclotomic number field as explained above. The idea is to use the oracle to recover the value of s relative to every $\mathfrak{q}_j R$ using the automorphisms. Once we have that, we can (efficiently) recover the s using the Chinese Remainder Theorem.

The reduction to find s mod $\mathfrak{q}_j R$ works as follows: transform each given sample $(a, b) \leftarrow \mathrm{A}_{s,\psi}$ to $(\tau_k(a), \tau_k(b)) \in R_q \times R_q$ where $k = j \cdot i^{-1} \in \mathbb{Z}_m^*$ which gives $\tau_k(\mathfrak{q}_j) = \mathfrak{q}_i$. Give the transformed sample to the oracle which outputs some $t \in R/\mathfrak{q}_i R$. Since τ_k is a bijection, we can compute $\tau_k^{-1}(t) \in R/\mathfrak{q}_j R$.

We now need to verify that this output is actually equal to s mod $\mathfrak{q}_j R$. This is equivalent to asking if $(\tau_k(a), \tau_k(b))$ is distributed according to $\mathrm{A}_{\tau_k(s),\psi'}$ for $\psi' = \tau_k(\psi)$. First, note that the automorphisms fix the underlying structure. Namely, $\tau_k(R) = R$ and in particular, $\tau_k(q) = q$ for any $k \in \mathbb{Z}_m^*$. We therefore have

$$\tau_k(b) = \tau_k(a) \cdot \tau_k(s) + \tau_k(e).$$

If a was uniformly distributed, $\tau_k(a)$ will be as well. The pairs are therefore distributed according to $\mathrm{A}_{\tau_k(s),\psi'}$ for $\psi' = \tau_k(\psi) \in \Psi$. The t returned by an oracle must therefore be $t = \tau_k(s)$ mod $\mathfrak{q}_i R$ and so $\tau_k^{-1}(t) = s$ mod $\tau_k^{-1}(\mathfrak{q}_i R) = s$ mod $\mathfrak{q}_j R$ as required.

All we need to show now is that the automorphisms preserve the distribution. Note that since the automorphisms simply permute the coordinates of the canonical embedding (see Section 2.3), then for any $\psi = \mathcal{D}_{\boldsymbol{r}} \in \Psi_{\leq\alpha}$ (each r_i of the vector $\boldsymbol{r}$ is $\leq \alpha$) and any $\mathbb{Q}$-automorphism $\tau_k : K \to K$, we have $\tau_k(\mathcal{D}_{\boldsymbol{r}}) = \mathcal{D}_{\boldsymbol{r'}}$ where $\boldsymbol{r'}$ is just a permutation of the coordinates of $\boldsymbol{r}$ and so each r_i is still $\leq \alpha$.

□

We ilustrate how the automorphisms act on the ring with the following examples. They present as follows:

Example 3.4. Take $n = 2$ and $q = 2n + 1 = 5$. The ring now looks like $R_q := \mathbb{Z}_q[x]/\Phi_{2n}(x) = \mathbb{Z}_5[x]/(x^2+1)$. Since $z = 2$ is the primitive root modulo 5, our polynomial splits as $x^2+1 \bmod 5 \equiv (x-2)(x-2^3) \equiv (x-2)(x-3)$. So our ideals are $\mathfrak{q}_1 = (x-2)$ and $\mathfrak{q}_2 = (x-3)$.

Just like in the proof, we are given access to an oracle – call it $\mathfrak{O}$ (in this case we will act as one because we already know the secret s, in principle, we need the $\mathfrak{O}$ to be efficient for the reduction to work) – that yields $s(x)$ modulo only one ideal, say it will be $\mathfrak{q}_1$ in this example.

Let us now take $s(x), a(x) \in \mathbb{Z}_5[x]/(x^2+1)$ to be $s(x) = 1 + 4x$ and $a(x) = 2x$. Since the oracle $\mathfrak{O}$ will return us the correct result for any input, it will also return us the correct input when the error e is 0. Let us then take it as such.

So we now encounter a sample $(a, b) \leftarrow A_{s,\psi}$ that looks like

$$(a, a \cdot s + e) = (2x, 2x(1 + 4x)) = (2x, 2x + 8x^2) = (2x, 2 + 2x).$$

Let us give this sample to the $\mathfrak{O}$ which will tell us what is $s(x)$ mod $\mathfrak{q}_1 R_q$. Since we are the oracle, we need to do the computation, and so

$$s(x) \bmod (x-2)R_q = 4$$

is the answer.

Compute now $b \bmod \mathfrak{q}_3 R_q = 2 + 2x \bmod (x-3)R_q = 3$ so that we can use the automorphism $\tau_k = \tau_{j/i} = \tau_3$ to map it to the result modulo $\mathfrak{q}_1$ for which we can obtain the answer. Note that we in fact can compute this because the q is know to everyone. Only the secret s and the distribution ψ are unknown, all the other variables are public. We therefore have

$$\tau_3(2 + 2x) = 2 + 2x^3 = 2 - 2x = 2 + 3x$$

and the oracle responds with $2 + 3x \bmod q_1 R_q = 3$. We now map $3(x-3)$ using $\tau_3^{-1} = \tau_{3^{-1}} = \tau_2$:

$$\tau_2(3(x-3)) = \tau(3x-9) = \tau_2(3x+1) = 3x^2 + 1 = -3 + 1 = 3.$$

Since we exhausted all the automorphisms (there are only 2), we can finally use the Chinese Remainder Theorem on the following system of equations where we set $s(x) = \alpha x + \beta$ as the intermediate values:

$$\begin{cases} \alpha x + \beta = 3 \bmod (x-2)R_q \\ \alpha x + \beta = 4 \bmod (x-3)R_q \end{cases} \implies \begin{cases} 2\alpha + \beta = 4 \\ 3\alpha + \beta = 3 \end{cases}$$

$$\beta = 4 - 2\alpha \implies 3\alpha + 4 - 2\alpha = 3 \implies \alpha = 4$$
$$\implies \beta = 4 - 2 \cdot 4 = 1$$

So $s(x) = \alpha x + \beta = 4x + 1$ which is (unsurprisingly) the correct answer. [Krzys: need to make this last align somewhat prettier]

Example 3.5. As the second example, take $n = 8$ and $q = 2n + 1 = 17$. Set $a, s \in \mathbb{Z}_1 7[x]/(x^8 + 1)$ as $a = x^2 + 16x^7$ and $s = 1 + x + 2x^2$. Just like in the previous example, we will take $e = 0$ and we compute:

$$\begin{aligned} a \cdot s &= (x^2 + 16x^7)(1 + x + 2x^2) \\ &= x^2 + x^3 + 2x^4 + 16x^7 + 16x^8 + 32x^9 \\ &= 1 + 2x + x^2 + x^3 + 2x^4 + 16x^7 \end{aligned}$$

The primitive root modulo 17 is 3 and so $x^8 + 1 \bmod 17$ splits as $x^n + 1 = \mathfrak{q}_1 \cdot \mathfrak{q}_3 \cdot \ldots \cdot \mathfrak{q}_{13} \cdot \mathfrak{q}_{15}$ where each $\mathfrak{q}_r = (x - z^r) = (x - 3^r)$ giving:

$$\begin{aligned} \mathfrak{q}_1 &= (x-3) & \mathfrak{q}_9 &= (x-14) \\ \mathfrak{q}_3 &= (x-10) & \mathfrak{q}_{11} &= (x-7) \\ \mathfrak{q}_5 &= (x-5) & \mathfrak{q}_{13} &= (x-12) \\ \mathfrak{q}_7 &= (x-11) & \mathfrak{q}_{15} &= (x-6) \end{aligned}$$

The last step left for us is the reduction to the decision-RLWE. We

Lemma 3.31 ($\mathfrak{q}_i$-LWE to DLWE)**.**

Proof. □

Alternative proof There is great disparity between required assumptions for the search and the decision version of the problem. The former holds for any number ring, (not even cyclotomic) whereas we need to make many assumptions about the structure of the ring from which we sample our variables from. It would be very practical if we could attain the same results but for more general structure. For one, the Galois fields are rare to encouter and the application might require us to consider something more modest. Secondly, from the cryptoanalytical point of view, usually the more assumptions we make, the less secure our problem is in the sence that one could abuse the additional structure for a specified attack.

One of the solutions was presented by C.Peikert O. Regev and N. Stephens-Davidowitz in their 2020 paper *Pseudorandomness of Ring-LWE for Any Ring and Modulus* – [PRS17]. The approach they use differs greately from the one we introduced here. Firstly, the reduction is from the decision-RLWE to the `BDD` (which was replaced by `CVP` in our case) directly, without invoking an oracle for the search version at all. Secondly, as mentioned, this reduction works for any number field K, not only cyclotomic, as well as for any modulus q, not only prime.

The very brief description of their approach is as follows. We take a point t from the ambient space and use the decision oracle to measure how close it is to the actual lattice point. The goal is to use the so called "acceptance probability" of the oracle on some new point, call it t', and check if it is significantly closer to the desired solution. This approach seems to be very easy, however, there are many important questions that need to be answered before proving that it actually works. For example, how do we pick the point t' and, more importantly, how do we know it is in fact closer to the desired solution? Answering the second question is the main concern of the paper. For the full proof, the reader is redirected to the paper itself – [PRS17].

4 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 [RAD78] (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 [Gen09a]. 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 we need a third party (e.g. someone with more computational power) to perform operations on our data while still retaining privacy. Alice can store her data somewhere on external server (for example the cloud) and ask to perform computations on it. We can for example query searches without the engine knowing what is actually being searched for.

In other words, we would like our encryption scheme – call it $\mathcal{E}$ – 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. $\mathcal{E}$ being *fully homomorphic* means that whenever f is a composition of **arbitrily many** additions and multiplications, then $\mathsf{Decrypt}_{\mathcal{E}}(f(c_1, \ldots, c_n)) = f(m_1, \ldots, m_n)$[13] which is also refered to as the *correctness* of the scheme.

Remark 4.1. Typically, an encryption scheme $\mathcal{E}$ is a tuple of $\mathsf{KeyGen}_{\mathcal{E}}$, $\mathsf{Encrypt}_{\mathcal{E}}$ 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 [KL14] or [HPS14] for more details on the abstract build of a encryption scheme). 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 [Gen10a] we will denote by $\mathcal{F}_{\mathcal{E}}$, the generalized set of such functions.

One might ask a question now, how secure can such scheme ultimately be? After all, we are giving the adversary a quite powerful tool in the form of being able to compute (ultimately) *any* function on our data.

4.1 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 [Dij+10] 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

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

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 p (no necessarily prime). To encrypt our message m, we choose integers q and r at random (such that the magnitude of $2r$ is smaller than $p/2$). We obtain the ciphertext c by computing:

$$c = pq + 2r + m. \tag{4.1}$$

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

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

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

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 $2(m_1 r_2 + m_2 r_1 + 2r_1 r_2)$ is small enough – i.e. smaller than p[14] – 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" – a concept we will introduce later on.

Example 4.2. Let's say you want to secretely tell your friend how many times Poland has won the football world cup and compute the square of that number because this makes it somehow meaningful for you. You pick $p = 5$, $q = 7$ and $r = 2 < p/2$ and compute $c = pq + 2r + m = 5 \cdot 7 + 2 \cdot 2 + 0 = 39$. Now, since you do not have a calculator at home and your phone is an old Nokia, you ask him to square it for you. So the friend takes the encryption of $m = 0$ being $c = 39$ and computes $39^2 = 1521$. He sends it back to you via a secret channel and you recieve the ciphertext 1521. You decrypt it with your secret key p and see $(1521 \bmod 5) \bmod 2 = 1 \bmod 2 = 1$. But this is not the correct answer. You (wrongfully) celebrate because you know that squaring the number makes it magically meaningful and so Poland (eventually) will become world champion. Unfortunately for 39 million people, it was an error due to the encryption scheme being not Fully Homomorphic. You happily sit down on your chair and turn on the TV instead of reading Gentry's paper on Fully Homomorphic Encryption.

This simple encryption scheme is thus somewhat homomorphic as per definition by Gentry in [Gen09a] – namely, it can be used to evaluate low-degree polynomials over encrypted data. Further on in the Section 6 of [Dij+10], 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

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

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 $x_0 \bmod p$ 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

$$c = (m + 2r + 2\sum_{i \in S} x_i) \bmod x_0. \tag{4.2}$$

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

The security of this preliminary SH scheme relies on the *Approximate GCD Problem*[15]. 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 [Gen10a] Section 3.4 and [Dij+10] Section 3 for details) we do not know any efficient (running in polynomial time) algorithm even if we are given arbitrarily many samples $c_i = r_i + p \cdot q_i$.

However, this comfortable security comes at great cost because, as shown in [Dij+10], 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 $\cdot$ 16-bits per letter $= 5 \cdot 16 \cdot 5 = 650$ seconds which is almost 11 minutes! As one can imagine, this is completely impractical for most applications.

4.2 Fully Homomorphic Encryption

We will now present the main idea introduced in Gentry's PhD thesis [Gen09a]. Namely, what is bootstrapping, why do we need it and how does it work.

4.2.1 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. Is there a way for us to somehow expand this set and yet retain the homomorphic properties of the scheme? Can we, further on, expand this set to include an arbitrary polynomial function? The answer turns out to be yes. As shown by Gentry, one of the requirements for this is that the scheme can decrypt its encryption correctly "and some". A bit more formally, we require that the $\mathsf{Decrypt}_\mathcal{E} \in \mathcal{F}_\mathcal{E}$. This part will be a brief explanation of what the bootstrapping is and how we can achieve it using (also introduced in the same paper) "squashing".

Remark 4.3. In cryptography, which is a field in the intersection of mathematics and computing science, instead of general functions, one often considers a *circuit* instead. Roughly speaking, a circut is a translation of what a mathematician thinks when they say "function". It consists of (finite amount) of gates which are just boolean functions. For example there is a NOT gate

[15] Later in [CS15], 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.

that takes a bit $b \in \{0,1\}$ and outputs $b+1$, where all the operations are performed modulo 2. Others include for example XOR - this is an exclusive logical OR - or NAND which is a AND followed up by a NOT gate. From a theoretical point of view, one can use solely a NAND gate to represent any circuit and thus we will be mostly concerned with that one.

Imagine we have a SH scheme $\mathcal{E}$ (one can think of the one from previous section as a concrete example) that is correct for its own decryption circuit augumented by an NAND gate. We call such scheme *bootstrappable*. As shown in the paper, one can use this circuit to create a (leveled) fully homomorphic encryption $\mathcal{E}^{(d)}$. It is called *leveled* because the correctnes depends on the "depth" (or the level) d of the circuit making it depend on d. It can be show that by appropriately augumenting the public key, such leveled scheme can be made independent of depth and thus made into a *fully homomorphic* one. Moreover, if the scheme itself is secure[16], then so is any $\mathsf{Evaluate}_{\mathcal{E}^{(d)}}$ algorithm. All of this is captured in the following theorem.

Theorem 4.1 ([Gen09b])**.** *One can construct a (semantically secure) family $\{\mathcal{E}^{(d)}\}$ of leveled fully homomorphic encryption schemes from any (semantically secure) bootstrappable encryption scheme $\mathcal{E}$.*

Why is this the correct requirement for a FHE? Suppose that there is an "error" associated with each ciphertext, just like in the scheme from Section 4.1. Then, as also noted there, after we perform one operations too many, the error builds up soo much that we are no longer able to decrypt correctly. We would therefore like to somehow make the error small enough again and "refresh" the ciphertext without using the secret key. Clearly we could get rid of the error completely if we were to decrypt it and create a "fresh" ciphertext of the same message. This is the precisely the idea, to decrypt the message but do it homomorphically! We obtain a homomorphic encryption by encrypting homomorphically – we are bootstrapping. Gentry's idea on how to do it, was by appending the encryption of the secret key in the public key itself.

Remark 4.4. Note that this requires another assumption called "circular security" or KDM (Key Dependent Message) security. That is, we are assuming that the publication of the encryption of a secret key does not leak any valuable information about the key itself. This is however very difficult to prove in practice but also no known attacks are know and hence it is just assumed along.

One last question we have is, why do we even need the error? Cannot we create a *semantically* secure encryption scheme that does not depend on any "error" to hide the message, and consequently, create a FHE without any sort of bootstrapping? It turns out that it is in fact not possible. Before we present the reason for this, let us motivate why we care about the semantic security so much in the case of HE.

Example 4.5. Using cloud computing as an example, imagine that after we stored few files in the cloud, we want to retrive files from the cloud that contain the *Answer to the Ultimate Question of Life, the Universe, and Everything*. We thus homomorphically query the engine for the file number 42. But can't the cloud just "notice" that the encryption of the file it is sending us now

[16]The precise security mentioned is the semantic security against chosen plaintext attacks. One can think of it as requiring that the same message m has different outputs c on different runs of the same encryption algorithm (it is non-deterministic). See [KL14] or [Pei16] for a more precise definition.

is the same as the encryption of the *Answer to the Ultimate Question of Life, the Universe, and Everything*? It cannot, if our encryption is semantically secure, because in that case, it is difficult to tell if two ciphertexts encrypt the same message.

Let us now get back to why we need the error. For any *deterministic* algorithm (like AES or RSA) that encrypts m as c every time it is run, any adversary can check if any m_0 encrypts as c and compare the two. Thus, by definition, such scheme cannot be semantically secure. We thus need a *probabilistic* algorithm that gives us different output every time it is run. This implies some sort of randomizing the ciphertext which can only be done (to our understanding) using some error to mask the message. Therefore we need some sort of "bootstrapping" (which is not necessarily the one mentioned above, this is only one way to achieve FHE but there could be more) to create a FHE.

How to bootstrap We will not present the precise idea behind *bootstrapping*. To this end, recall that our generic homomorphic scheme $\mathcal{E}$ consists of four algorithms. These are:

1. $\mathsf{KeyGen}_{\mathcal{E}}$ which generates the public and secret keys – pk and sk respectively.
2. $\mathsf{Encrypt}_{\mathcal{E}}(\mathrm{pk}, m)$ which encrypts the message m from the plaintext space $\mathcal{P}$ under the public key pk and outputs the ciphertext c.
3. $\mathsf{Decrypt}_{\mathcal{E}}(\mathrm{sk}, c)$ which decrypts the ciphertext c using the secret key sk.
4. $\mathsf{Evaluate}_{\mathcal{E}}(\mathrm{pk}, f, c_1, \ldots, c_t)$ which evaluates the function $f \in \mathcal{F}_{\mathcal{E}}$ from the set of permitted functions on the ciphertexts $c_1, \ldots, c_t$ using the public key pk.

We shall now construct a new algorithm which we will call $\mathsf{Recrypt}_{\mathcal{E}}$ using only those four above. For simplicity, we assume that our plaintext space is just $\{0, 1\}$. Imagine we have a valid ciphertext c_1 that encrypts m_1 under the public key pk_1. In order to decrypt it homomorphically, assume additionally that we have the secret key sk_1 encrypted under another public key pk_2 – denote this encryption by $\hat{\mathrm{sk}}_1$ ([Krzys: idk how to make it span the whole variable]). Define now the following algorithm:
$\mathsf{Recrypt}_{\mathcal{E}}(\mathrm{pk}_2, \mathsf{Decrypt}_{\mathcal{E}}, \bar{\mathrm{sk}}_1, c_1)$:

Set $\hat{c}_1 \leftarrow \mathsf{Encrypt}_{\mathcal{E}}(\mathrm{pk}_2, c_1)$
Set $c_2 \leftarrow \mathsf{Evaluate}_{\mathcal{E}}(\mathrm{pk}_2, \mathsf{Decrypt}_{\mathcal{E}}, \hat{\mathrm{sk}}_1, \hat{c}_1)$
Return c_2

Here we use the encryption of our secret key to evaluate the decryption function homomorphically under a new public key. Now we can see why we need the assumption that the scheme is KDM secure. Since the encryption of sk_1 is public, we need to make sure it does not leak any information about it. The statement of the Theorem 4.1 should now be intuitively clear. As long as the scheme $\mathcal{E}$ can evaluate its own decryption function, and the Recrypt algorithm leaves the new message with less error than we began with, we can use $\mathcal{E}$ to construct a leveled FHE scheme. In order to now evaluate *any* function on our data, we would also like our scheme to correctly evaluate a NAND gate (recall that we can create any gate using a finite combination of NAND gates) in order to make some progress before the error "blows up".

4.2.2 Simple lattice based scheme

The motivation for the choice of lattices as opposed to number theoretic constructions (like RSA or ElGamma for example, which are based on exponentiation), is that the former has few desirable properties. Firstly, the lattices have lower decryption complexity and are therefore more suitable as a bootstrappable encryption scheme. Next, one requires not only one supported homomorphism like addition but also the multiplication – lattices as ideals poses both. Those factors among many others, (see [Gen09b] for more details) have led to the choice of *ideal lattices*.

The idea behind the encryption is somewhat similar to the one of GGH. That is, we are going to fix a basis for a lattice (which is an ideal), publish the bad basis as the public key and keep the good basis as a secret key. The security of this scheme is based on the Ideal Coset Problem[17] (ICP) introduced below. Briefly stated, given an element of some ring $t \in R$, the ICP asks us to distinguish between a representative of a coset taken from some distribution over an arbitrary ring and a uniform sample.

For a ring R and a basis[18] $\boldsymbol{B}_{\mathcal{I}}$ for an ideal $\mathcal{I} \subset R$, we use the notation $R \bmod \boldsymbol{B}_{\mathcal{I}}$ to represent the set of $r + \mathcal{I}$ for $r \in R$.

Definition 4.2 (Ideal Coset Problem)**.** Fix a ring R along with an ideal $\mathcal{I} \subseteq R$ and sample an element $r \in R$ given a sampling algorithm Samp. Now pick a basis $\boldsymbol{B}_{\mathcal{I}}$ for the ideal $\mathcal{I}$. Given the basis $\boldsymbol{B}_{\mathcal{I}}$, the challenger is asked to distinguish between $t \equiv r \bmod \boldsymbol{B}_{\mathcal{I}}$ and t being chosen uniformly.

Few remarks are in place now. Firstly, this definition hides few details in favour of clearer notation. As one, we actually fix some ideal $\mathcal{J} \subseteq R$ such that $\mathcal{I}$ and $\mathcal{J}$ are coprime (i.e. $\mathcal{I}+\mathcal{J} = R$) first, and then, with respect to such $\mathcal{J}$, we instantiate the $\mathcal{I}$. This will become evident later in this section why this is required. Secondly, the procedure somewhat depends on the efficiency of those choices. One example would be how to pick such coprime ideals as well as how to sample the r from the ring itself. The questions (partially) are answered in the paper [Gen09a] itself.

From now on, let R be an arbitrary ring, and set $\mathcal{I}, \mathcal{J} \subset R$ as two coprime ideals of R. We will also be using a sampling algorithm which we call $\mathsf{Samp}(x, \boldsymbol{B}_{\mathcal{I}}, R, \boldsymbol{B}_{\mathcal{J}})$ that samples from the coset $x + \mathcal{I}$. Let us now introduce the scheme in terms of rings and ideals only.

$\mathsf{KeyGen}(\boldsymbol{B}_{\mathcal{I}}, R)$
Generate two basis for $\mathcal{J}$, $\boldsymbol{B}^{sk}_{\mathcal{J}}$ and $\boldsymbol{B}^{pk}_{\mathcal{J}}$. The **public key** pk includes R, $\boldsymbol{B}_{\mathcal{I}}$, $\boldsymbol{B}^{pk}_{\mathcal{J}}$ and the sampling algorithm Samp. The **secret key** sk also includes the $\boldsymbol{B}^{sk}_{\mathcal{J}}$. We denote by $\mathcal{P}$ the plaintext space which is (a subset of) $R \bmod \boldsymbol{B}_{\mathcal{I}}$.
$\mathsf{Encrypt}(\mathrm{pk}, m)$
For a message $m \in \mathcal{P}$, set $c' \leftarrow \mathsf{Samp}(m, \boldsymbol{B}_{\mathcal{I}}, R, \boldsymbol{B}^{pk}_{\mathcal{J}})$ and output $c \equiv c' \bmod \boldsymbol{B}^{pk}_{\mathcal{J}}$.
$\mathsf{Decrypt}(\mathrm{sk}, c)$
For the ciphertext c, output $(c \bmod \boldsymbol{B}^{sk}_{\mathcal{J}}) \bmod \boldsymbol{B}_{\mathcal{I}}$.
$\mathsf{Evaluate}(\mathrm{pk}, f, C)$
This algorithm takes a set of ciphertexts $C = (c_1, \ldots, c_t)$ and applies the function $f \in \mathcal{F}_{\mathcal{E}}$

[17] Later, in 2010, Gentry showed that the scheme can be based on the worst-case BDD problem – [Gen10b].
[18] By the word "basis" we simply mean the set of generators for the ideal but we use that instead to highlight the connection between the ideals and lattices as well as keep it consistent with Gentry's work.

using the public key pk. It outputs $f(c_1, \ldots, c_t)$. Since we only support two operations from the ring, $f : R \to R$ needs to be a homomorphism.

Having the bootstrapping in mind, we should pay close attention to the correctness of the decryption algorithm. Before proving it, let us state few definitions first.

Definition 4.3. Let X_{Enc} be the image of Samp. Notice that all ciphertexts output by $\mathsf{Encrypt}$ are in $X_{Enc} + J$. Let X_{Dec} equal R mod $\boldsymbol{B}_{\mathcal{J}}^{sk}$, the distinguished representatives of cosets of $\mathcal{J}$ wrt the secret basis $\boldsymbol{B}_{\mathcal{J}}^{sk}$.

Definition 4.4 (Permitted Functions)**.** Let

$$\mathcal{F}_{\mathcal{E}} := \{f : \forall (x_1, \ldots, x_t) \in X_{Enc}^T, f(x_1, \ldots, x_t) \in X_{Dec}.\}$$

In other words, $\mathcal{F}_{\mathcal{E}}$ is the set of those functions of which output will always be in X_{Dec} when the input is from X_{End}.

Aditionally, we say the $\hat{c}$ a *valid ciphertext* if it equals $\mathsf{Evaluate}(\text{pk}, f, (c_1, \ldots, c_t)$ for some function $f \in \mathcal{F}_{\mathcal{E}}$. We can now state and prove the correctness of our scheme.

Theorem 4.5 (Correctness)**.** *Assume $\mathcal{F}_{\mathcal{E}}$ is a set of permitted functions containing the identity. $\mathcal{E}$ is correct for $\mathcal{F}_{\mathcal{E}}$ — i.e., $\mathsf{Decrypt}$ correctly decrypts valid ciphertexts.*

Proof. For $C = \{c_1, \ldots, c_t\}, c_k = m_k + i_k + j_k$, where $m_k \in \mathcal{P}, i_k \in \mathcal{I}, j_k \in \mathcal{J}$ and $m_k + i_k \in X_{Enc}$, we have

$$\begin{aligned}
\mathsf{Evaluate}(\text{pk}, f, C) =& f(C) \bmod \boldsymbol{B}_{\mathcal{J}}^{\text{pk}} \\
=& f(m_1 + i_1 + j_1, \ldots, m_t + i_t + j_t) \bmod \boldsymbol{B}_{\mathcal{J}}^{\text{pk}} \\
\in& f(m_1 + i_1, \ldots, m_t + i_t) + \mathcal{J}.
\end{aligned}$$

If $f \in \mathcal{F}_{\mathcal{E}}$, then we have $f(X_{Enc}, \ldots, X_{Enc}) \in X_{Dec}$ and so

$$\begin{aligned}
&\mathsf{Decrypt}(\text{sk}, \mathsf{Evaluate}(\text{pk}, C)) \\
&\quad = f(m_1 + i_1, \ldots, m_t + i_t) \bmod \boldsymbol{B}_{\mathcal{I}} \\
&\quad = f(m_1, \ldots, m_t)
\end{aligned}$$

as required. □

Once we have the correctness, we can focus on the actual implementations of any fully homomorphic schemes. This in turn implies that we should pay very close attention to the size of X_{Enc} and X_{Dec}. Namely, we want to maximize X_{Dec} and minimize X_{Enc} (while still retaining security). This is where we introduce the lattices.

4.3 Further developments in FHE

Since the inception of FHE, many new ideas have emerged as potential replacement for the details in the implementation in Gentry's work. In this section, we will briefly introduce some of such techniques and ideas. Finally the work we have done on LWE and its ring equivalent will pay off as we can rely on those results to abstractly create FHE scheme [Krzys: i dont like this sentence but idk how to make it prettier].

4.3.1 FHE from the standard LWE

One of such ideas is to replace the ideal lattices by an LWE sample instead. Let us now attempt to construct a SH scheme that is using the LWE assumption along the ideas presented in [BV11a]. Recall that a $\text{LWE}_{q,\chi}$ is specified by the odd modulus $q > 2$ and error distribution χ. It provides us with the LWE distribution that we called $A_{s,\chi}$ where $\boldsymbol{s}$ was a random vector representing the secret key outputting us samples of the form

$$A_{s,\chi} \to (\boldsymbol{a}, b) = (\boldsymbol{a}, \langle \boldsymbol{a}, \boldsymbol{s} \rangle + e),$$

where e was drawn from some distribution χ (usually taken to be the discrete Gaussian distribution with small standard deviation) over $\mathbb{Z}_q$ for q prime[19]. We can now imagine a scheme that works as follows: to encrypt a single bit $m \in \{0, 1\}$ using a secret key $\boldsymbol{s}$, draw a sample from $A_{s,\chi}$ and output the ciphertext

$$c \;=\; (\boldsymbol{a}, b = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + 2e + m) \quad \in \mathbb{Z}_q^n \times \mathbb{Z}_q$$

Note that we have replaced the error e with its two-times multiple in contrast to the original formulation by Regev as can be seen in Table 2. This is not a problem at all because 2 and q are coprime. To decrypt the message, first compute $\langle \boldsymbol{a}, \boldsymbol{s} \rangle$ and substract that from b, giving $2e + m \bmod q$ which, since $e \ll q$, is actually equal to $2e + m$ exactly. Finally, reduce modulo 2 and we are left with the original message m.

The scheme is clearly additively homomorphic. The issue arises when we try to multiply two ciphertexts. As shown in [GHV10], the (slight variation of) this scheme supports only a *single* homomorphic multiplication with the expense of huge blowup in the ciphertext size.

To see how this is the case, consider a symbolic function $f_{\boldsymbol{a},b} : \mathbb{Z}_q^n \to \mathbb{Z}_q$ defined as:

$$f_{\boldsymbol{a},b}(\boldsymbol{x}) = b - \langle \boldsymbol{a}, \boldsymbol{x} \rangle \bmod q = b - \boldsymbol{a}_i \boldsymbol{x}^i \quad \in \mathbb{Z}_q$$

where we are using Eisenstein notation to represent elements $\boldsymbol{x} \in \mathbb{Z}_q^n$ (the transpose of the first element is implicit). Note now that the decryption is nothing else but evaluating this function on the secret key $\boldsymbol{s}$ and taking the result modulo 2. We can now define addition and multiplication using this function. Addition is straightforward, $f_{\boldsymbol{a},b}$ is a linear function and so sum of two linear functions is still linear. Symbolically, $f_{\boldsymbol{a},b}(\boldsymbol{x}) + f_{\boldsymbol{a}',b'}(\boldsymbol{x}) = f_{(\boldsymbol{a}+\boldsymbol{a}',b+b')}(\boldsymbol{x})$ will represent the homomorphically added ciphertext $(\boldsymbol{a} + \boldsymbol{a}', b + b')$. Similarly, multiplying two such functions gives us:[Krzys: i think ill get rid of eisenstein notation coz im not really consistent with it]

$$\begin{aligned} f_{\boldsymbol{a},b}(\boldsymbol{x}) \cdot f_{\boldsymbol{a}',b'}(\boldsymbol{x}) &= (b - \boldsymbol{a}_i \boldsymbol{x}^i) \cdot (b' - \boldsymbol{a}'_i \boldsymbol{x}^i) \\ &= h_0 + h_i \cdot \boldsymbol{x}^i + h_{i,j} \cdot \boldsymbol{x}^i \boldsymbol{x}^j, \end{aligned} \tag{4.3}$$

which yields us a second degree polynomial with coefficients $h_{i,j}$ that can be computed by expanding the parenthesis of the upper equation. The decryption is as before, that is evaluating at $\boldsymbol{s}$ and reducing modulo 2. Hence, the scheme is trully homomorphic. Unfortunately, nothing in life comes for free and indeed, the multiplication which took a ciphertexts of size $n + 1$, expanded it to the one of size approximately $n^2/2$. As one might imagine this is completely unacceptable from an efficiency point of view and surely not enough for bootstrapping. This is where the main contribution of [BV11a] comes in – the *re-linearization* technique.

[19] Since we will be using the decision version of the problem, we need to assume q to be prime.

Re-linearization The goal is to reduce the ciphertext blow-up for multiplication. As turns out we can actually reduce the result back to just a $n+1$ size assuming something that resembles KDM security or "circular security" - i.e. we need to assume that the encryption of a secret key does not leak any information about it. However, the key difference is that we encrypt all of the linear and quadratic terms of $\boldsymbol{s}$ but using a different key, call it $\boldsymbol{t}$. More precisely, we encrypt numbers $\boldsymbol{s}^i$ as well as $\boldsymbol{s}^{i,j}$ using the new secret key $\boldsymbol{t}$. The adjusted equation 4.3 with $\boldsymbol{s}$ plugged in for $\boldsymbol{x}$ now (approximately) looks like:

$$\begin{aligned} & h_0 + h_i \cdot \boldsymbol{s}^i + h_{i,j} \cdot \boldsymbol{s}^i \boldsymbol{s}^j \\ = \; & h_0 + h_i \cdot (b^i - \langle \boldsymbol{a}^i, \boldsymbol{t} \rangle) + h_{i,j} \cdot (b^{i,j} - \langle \boldsymbol{a}^{i,j}, \boldsymbol{t} \rangle), \end{aligned}$$

which is just a linear function in $\boldsymbol{t}$! The key take-away is that multiplying the two linear functions and later re-linearizing, gives us another linear function that, when evaluated in $\boldsymbol{t}$, outputs the product of the original messages.

4.3.2 FHE from the ring-LWE

A step forward in losing the need for KDM security (or circular security) was presented by Z. Brakerski and V. Vaikuntanathan in [BV11b]. There, they have used the ring-LWE assumptions to create a FHE scheme with provable security for KDM. The scheme is relatively simple thanks to the assumptions that guarantee security based on ring-LWE assumptions. We will begin the construction with the SH scheme and later prove the circular security.

Ring-LWE SHE scheme

The assumptions for this scheme are almost identical as for the ring-LWE, i.e. we take a randomly from some ring that we call R_q and the error term e from some distribution χ (again, usually taken to be Gaussian). Since we will be basing our scheme on the decision version, we should take q to be prime just like in the previous scheme. For compactness we avoid a precise formulation of the parameters, we refer the reader to the Section 3.3 on the hardness of ring-LWE. The only difference in assumptions is that we will take the secret s also from the same distribution χ which will be useful for the circular security. Note that this choice does not affect the results presented for ring-LWE.

Let us first introduce a simpler (symmetric) scheme that is only additively homomorphic. This is not difficult as shown couple of times above. We first take a uniform and sample $s, e \leftarrow \chi$ both from the same error distribution χ. Now, to encrypt a message $m \in R_2 = \mathbb{Z}[x]/(x^n+1)$ that lives in the set of polynomials with binary coefficients, we set

$$\boldsymbol{c} = (c_0, c_1) = (as + 2e + m, -a).$$

To decrypt the message $\boldsymbol{c} = (c_0, c_1)$ we compute

$$(c_0 + c_1 s) \bmod 2 = as + 2e + m - as \bmod 2 = m.$$

Let us now check the claimed additive homomorphism of the scheme. Assuming $\boldsymbol{c'}, \boldsymbol{c}$ encrypt messages m, m' respectively using the same secret key, compute

$$\begin{aligned}\boldsymbol{c'} + \boldsymbol{c} =& (c_0 + c'_0, c_1 + c'_1) = (as + 2e + m + a's + 2e' + m', -a - a') = \\ & ((a + a')s + 2(e + e') + (m + m'), -a - a')\end{aligned}$$

and see that it decrypts correctly as long as the error is not too big.

It should not come as a surprise that it is the multiplication we should pay most attention to. Let us then compute what a product of the first term $c_0 \cdot c'_0$ will look like

$$\begin{aligned}c_0 \cdot c'_0 =& (as + 2e + m)(a's + 2e' + m') \\ =& \ aa's^2 + 2(2ee' + em' + e'm) + mm' \\ & + s(a'(2e + m) + a(2e' + m')) \\ =& \ aa's^2 + 2(2ee' + em' + e'm) + mm' \\ & + s(a'c_0 + ac'_0 - 2aa's) \\ =& \ -aa's^2 + s(a'c_0 + ac'_0) + 2(2ee' + em' + e'm) + mm'.\end{aligned}$$

This almost looks like a valid ciphertext for $c_0 \cdot c'_0$ if not for the $-aa's^2$ term. We can fix that by adding one more value to our ciphertext to make it look like $\boldsymbol{c}_{mult} = (c_{mult,0}, c_{mult,1}, c_{mult,2})$ where $c_{mult,2} = c_1 c'_1$, $c_{mult,1} = c_0 c'_1 + c'_0 c_1$ and $c_{mult,0} = c_0 c'_0$. To now decrypt the 3 element ciphertext, compute $c_0 + c_1 s + c_2 s^2$ mod 2. It is important to note that we can still add and multiply the ciphertexts of all lengths – the addition yields the ciphertext of the bigger size of the operands and multiplication yields the ciphertext of size equal to the sum of operands minus one.

Parameters We will skip most of the parameter choices as they often depend on the implementation. Most of them are just like in the standard case presented in the Section 3.3, i.e. q is prime and R_q is the ring of integers of some cyclotomic field of degree 2^k. Additionally, instead of taking our messages to be only binary, we shall widen our scope to any element of R_t where $t \in \mathbb{Z}_q^*$ is also prime – this will be also useful for the KDM security. If we assume that the greatest "depth" we can evaluate our polynomial functions at is D, we define the general scheme as follows.

KeyGen

Sample a ring element $s \leftarrow \chi$ and set the secret key vector as $(1, s, s^2, \ldots, s^D) \in R_q^{D+1}$.

Encrypt

Recall that the plaintext space is R_t. To encrypt, sample $(a, as + te) \in R_q^2$ where a was uniform and $e \leftarrow \chi$ taken from some distribution χ. Output the $\boldsymbol{c} = (c_0, c_1) \in R_q^2$ computed as

$$c_0 = b + m \in R_q; \quad c_1 = -a.$$

While the encryption of a single element lives in R_q^2, homomorphic operations might add more elements as seen above. Thus, a more generic form for the ciphertext is $\boldsymbol{c} = (c_0, c_1, \ldots, c_D)$ and we can represent any ciphertext like that because padding with zeros does not influence the ciphertext.

Decrypt

Since the general form of the ciphertext is $\boldsymbol{c} = (c_0, c_1, \ldots, c_D) \in R_q^{D+1}$, the decryption is a

simple $\langle \boldsymbol{c}, \boldsymbol{s} \rangle \bmod t = \sum_{i=0}^{D} c_i s^i \bmod t \in R_q$

Evaluate

The sum of two ciphertexts $\boldsymbol{c}, \boldsymbol{c'}$ is simply the coordinate-wise addition (note that we assume that they are the same length here, for example by padding). For the multiplication, denote $\boldsymbol{c} = (c_0, \ldots, c_d)$ and $\boldsymbol{c'} = (c'_0, \ldots, c_{d'})$ and use x as a *symbolic* variable to compute

$$\left(\sum_{i=0}^{d} c_i x^i \right) \cdot \left(\sum_{i=0}^{d'} c'_i x^i \right) = \left(\sum_{i=0}^{d+d'} \hat{c}_i x^i \right)$$

where the $\hat{c}_i$'s are the result of multiplication modulo R_q. The output ciphertext is $\boldsymbol{c}_{mult} = (\hat{c_0}, \ldots, \hat{c}_{d+d'})$.

KDM security Let us now briefly motivate how this scheme is circularly secure for a linear function. For a more rigorous presentation, see [BV11b].

Consider the ciphertext $\boldsymbol{c} = (as + 2e + s, -a)$ which "looks like" and encryption of the secret key s. If we now define $a' = a + 1$ we have $\boldsymbol{c} = (a's + 2e, -a' + 1)$ we notice that the part $a's + 2e$ is exactly a RLWE sample which (by previous section) is completely indistinguishable from an uniform sample – call it u. Therefore, $\boldsymbol{c} \approx (u, -a' + 1)$ which is a completely uniform pair and so, we cannot tell what the s was.

4.3.3 Other works

5 Conclusions

We have now shown cryptographic lattice based cryptographic schemes and how we can use them to create a Fully Homomorphic Encryption.

a quote from [JS16]: "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 [Ber09]: “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.

jj

References

[Ajt96] M. Ajtai. "Generating Hard Instances of Lattice Problems (Extended Abstract)". In: *Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing*. STOC '96. Philadelphia, Pennsylvania, USA: Association for Computing Machinery, 1996, pp. 99–108. ISBN: 0897917855. DOI: 10.1145/237814.237838. URL: https://doi.org/10.1145/237814.237838.

[Bab86] László Babai. "On Lovász' lattice reduction and the nearest lattice point problem". In: *Combinatorica* 6 (1986), pp. 1–13.

[Ber09] 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.

[Bir48] G. Birkhoff. *Lattice Theory*. Colloquium publications t. 25. American Mathematical Society, 1948. URL: https://books.google.pl/books?id=2xcPAAAAIAAJ.

[BP20] Huck Bennett and Chris Peikert. "Hardness of Bounded Distance Decoding on Lattices in l_p Norms". In: *CoRR* abs/2003.07903 (2020). arXiv: 2003.07903. URL: https://arxiv.org/abs/2003.07903.

[Buh98] "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.

[BV11a] Zvika Brakerski and Vinod Vaikuntanathan. "Efficient Fully Homomorphic Encryption from (Standard) LWE". In: *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*. 2011, pp. 97–106. DOI: 10.1109/FOCS.2011.12.

[BV11b] Zvika Brakerski and Vinod Vaikuntanathan. "Fully Homomorphic Encryption from Ring-LWE and Security for Key Dependent Messages". In: *Advances in Cryptology – CRYPTO 2011*. Ed. by Phillip Rogaway. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 505–524. ISBN: 978-3-642-22792-9.

[CS15] 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.

[Dij+10] 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.

[Din+03] 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.

[Din12] Jintai Ding. "A Simple Provably Secure Key Exchange Scheme Based on the Learning with Errors Problem". In: *IACR Cryptol. ePrint Arch.* 2012 (2012), p. 688.

[Gen09a] Craig Gentry. “A Fully Homomorphic Encryption Scheme”. AAI3382729. PhD thesis. Stanford, CA, USA, 2009. ISBN: 9781109444506.

[Gen09b] 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.

[Gen10a] 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.

[Gen10b] Craig Gentry. “Toward Basing Fully Homomorphic Encryption on Worst-Case Hardness”. In: *Advances in Cryptology – CRYPTO 2010*. Ed. by Tal Rabin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 116–137. ISBN: 978-3-642-14623-7.

[GGH97] 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.

[GHV10] Craig Gentry, Shai Halevi, and Vinod Vaikuntanathan. “A Simple BGN-Type Cryptosystem from LWE”. In: *Advances in Cryptology – EUROCRYPT 2010*. Ed. by Henri Gilbert. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 506–522. ISBN: 978-3-642-13190-5.

[GM84] Shafi Goldwasser and Silvio Micali. “Probabilistic encryption”. In: *Journal of Computer and System Sciences* 28.2 (1984), pp. 270–299. ISSN: 0022-0000. DOI: https://doi.org/10.1016/0022-0000(84)90070-9. URL: https://www.sciencedirect.com/science/article/pii/0022000084900709.

[Hal02] Sean Hallgren. “Polynomial-Time Quantum Algorithms for Pell’s Equation and the Principal Ideal Problem”. In: *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*. STOC ’02. Montreal, Quebec, Canada: Association for Computing Machinery, 2002, pp. 653–658. ISBN: 1581134959. DOI: 10.1145/509907.510001. URL: https://doi.org/10.1145/509907.510001.

[Hal05] Sean Hallgren. “Fast Quantum Algorithms for Computing the Unit Group and Class Group of a Number Field”. In: *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*. STOC ’05. Baltimore, MD, USA: Association for Computing Machinery, 2005, pp. 468–474. ISBN: 1581139608. DOI: 10.1145/1060590.1060660. URL: https://doi.org/10.1145/1060590.1060660.

[HPS14] 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.

[JS16] 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.

[KL14] Jonathan Katz and Yehuda Lindell. *Introduction to Modern Cryptography, Second Edition*. 2nd. Chapman & Hall/CRC, 2014. ISBN: 1466570261.

[LLL82] H.W. jr. Lenstra, A.K. Lenstra, and L. Lovász. "Factoring Polynomials with Rational Coefficients." In: *Mathematische Annalen* 261 (1982), pp. 515–534. URL: http://eudml.org/doc/182903.

[LP10] Richard Lindner and Chris Peikert. *Better Key Sizes (and Attacks) for LWE-Based Encryption*. Cryptology ePrint Archive, Paper 2010/613. https://eprint.iacr.org/2010/613. 2010. URL: https://eprint.iacr.org/2010/613.

[LPR12] 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.

[LPR13] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. *A Toolkit for Ring-LWE Cryptography*. Cryptology ePrint Archive, Paper 2013/293. https://eprint.iacr.org/2013/293. 2013. URL: https://eprint.iacr.org/2013/293.

[Lyu+08] Vadim Lyubashevsky et al. "SWIFFT: A Modest Proposal for FFT Hashing". In: *Fast Software Encryption*. Ed. by Kaisa Nyberg. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 54–72. ISBN: 978-3-540-71039-4.

[Mar+12] Enrique Martín-López et al. "Experimental realization of Shor's quantum factoring algorithm using qubit recycling". In: *Nature Photonics* 6.11 (Nov. 2012), pp. 773–776. ISSN: 1749-4893. DOI: 10.1038/nphoton.2012.259. URL: https://doi.org/10.1038/nphoton.2012.259.

[Mar77] Daniel A. Marcus. *Number fields*. Universitext. Springer-Verlag, New York-Heidelberg, 1977, pp. viii+279. ISBN: 0-387-90279-1.

[MG02] Daniele Micciancio and Shafi Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*. Vol. 671. The Kluwer International Series in Engineering and Computer Science. Boston, Massachusetts: Kluwer Academic Publishers, Mar. 2002.

[Mic01a] Daniele Micciancio. "Improving Lattice Based Cryptosystems Using the Hermite Normal Form". In: *LNCS* 2146 (Nov. 2001). DOI: 10.1007/3-540-44670-2_11.

[Mic01b] 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.

[Mic07] 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.

[Mic08] Daniele Micciancio. “Efficient reductions among lattice problems”. In: *ACM-SIAM Symposium on Discrete Algorithms*. 2008.

[MR07] Daniele Micciancio and Oded Regev. “Worst-Case to Average-Case Reductions Based on Gaussian Measures”. In: *SIAM Journal on Computing* 37.1 (2007), pp. 267–302. DOI: 10.1137/S0097539705447360. eprint: https://doi.org/10.1137/S0097539705447360. URL: https://doi.org/10.1137/S0097539705447360.

[MR09] 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.

[Ngu99] 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.

[NR06] 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.

[NS01] 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.

[Pei09] Chris Peikert. “Public-Key Cryptosystems from the Worst-Case Shortest Vector Problem: Extended Abstract”. In: *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*. STOC ’09. Bethesda, MD, USA: Association for Computing Machinery, 2009, pp. 333–342. ISBN: 9781605585062. DOI: 10.1145/1536414.1536461. URL: https://doi.org/10.1145/1536414.1536461.

[Pei16] 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.

[PRS17] Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. *Pseudorandomness of Ring-LWE for Any Ring and Modulus*. Cryptology ePrint Archive, Paper 2017/258. https://eprint.iacr.org/2017/258. 2017. URL: https://eprint.iacr.org/2017/258.

[PVW07] Chris Peikert, Vinod Vaikuntanathan, and Brent Waters. “A Framework for Efficient and Composable Oblivious Transfer.” In: *IACR Cryptology ePrint Archive* 2007 (Jan. 2007), p. 348.

[RAD78] Ronald L. Rivest, Len Adleman, and Michael L. Dertouzos. “On Data Banks and Privacy Homomorphisms”. In: 1978.

[Reg05] 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.

[Rib01] P. Ribenboim. *Classical Theory of Algebraic Numbers*. Universitext. Springer New York, 2001. ISBN: 9780387950709. URL: https://books.google.nl/books?id=u5443xdaNZcC.

[RSA78] R. L. Rivest, A. Shamir, and L. Adleman. “A Method for Obtaining Digital Signatures and Public-Key Cryptosystems”. In: *Commun. ACM* 21.2 (Feb. 1978), pp. 120–126. ISSN: 0001-0782. DOI: 10.1145/359340.359342. URL: https://doi.org/10.1145/359340.359342.

[SD22] Javad Sharafi and Hassan Daghigh. “A Ring-LWE-based digital signature inspired by Lindner-Peikert scheme”. In: *Journal of Mathematical Cryptology* 16.1 (2022), pp. 205–214. DOI: doi:10.1515/jmc-2021-0013. URL: https://doi.org/10.1515/jmc-2021-0013.

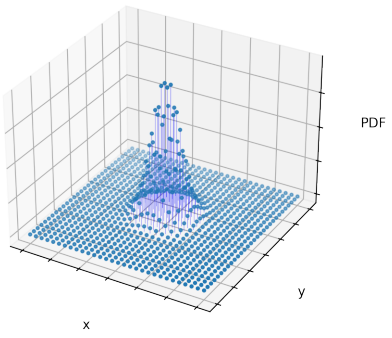
[Sho97] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. DOI: 10.1137/s0097539795293172. URL: https://arxiv.org/abs/quant-ph/9508027.

[Ste04] William A. Stein. “A Brief Introduction to Classical and Adelic Algebraic Number Theory”. In: 2004.

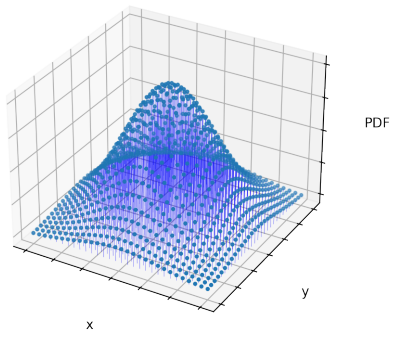
[SV05] Arthur Schmidt and Ulrich Vollmer. “Polynomial Time Quantum Algorithm for the Computation of the Unit Group of a Number Field”. In: *Proceedings of the Thirty-Seventh Annual ACM Symposium on Theory of Computing*. STOC ’05. Baltimore, MD, USA: Association for Computing Machinery, 2005, pp. 475–480. ISBN: 1581139608. DOI: 10.1145/1060590.1060661. URL: https://doi.org/10.1145/1060590.1060661.

[Van+01] Lieven M. K. Vandersypen et al. “Experimental realization of Shor’s quantum factoring algorithm using nuclear magnetic resonance”. In: *Nature* 414.6866 (Dec. 2001), pp. 883–887. ISSN: 1476-4687. DOI: 10.1038/414883a. URL: https://doi.org/10.1038/414883a.

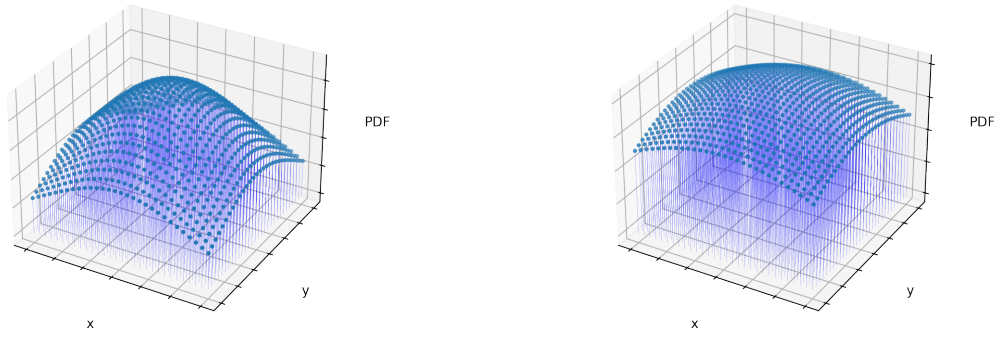
Figure 2: Priestess of Delphi (1891) by John Collier.



(a) $r = 0.2$



(b) $r = 2$



(a) $r = 7$ (b) $r = 20$

Figure 4: 2-dimensional Gaussian with standard deviation r. The z-axis represents the probability.

Figure 5: Two iterations of the iterative step

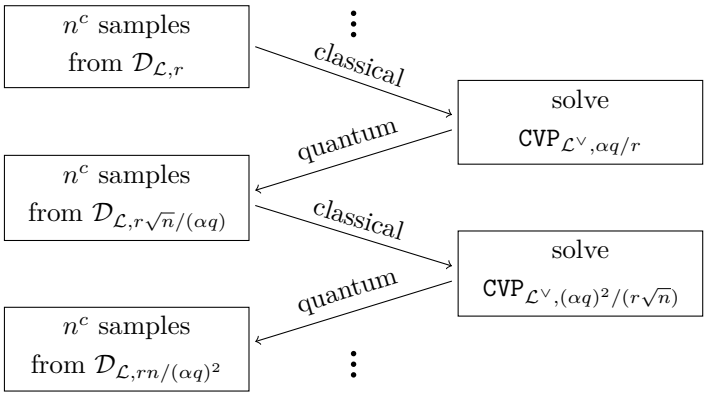
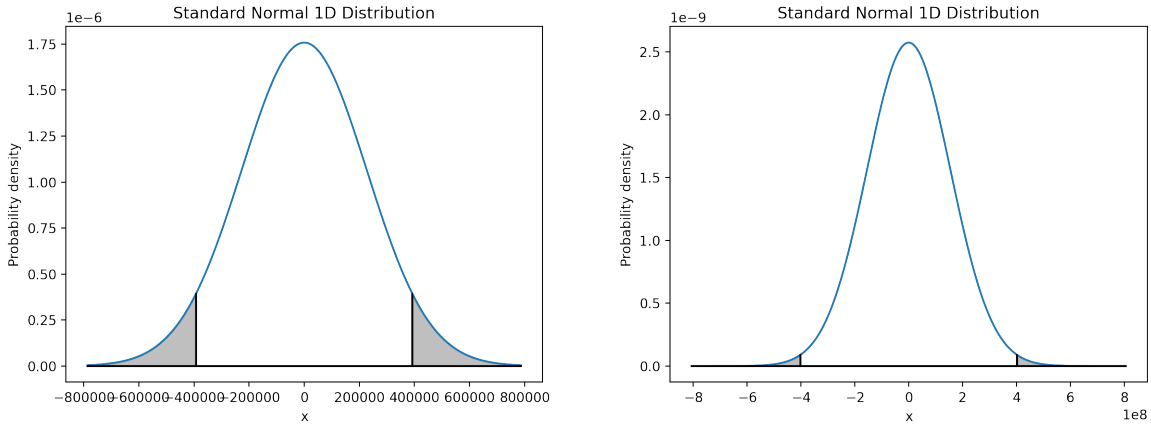


Figure 6: 1 dimensional Gaussian with standard deviation $\sigma^2 = q/\log n$ for:



(a) $n = 1024$ (b) $n = 32768$

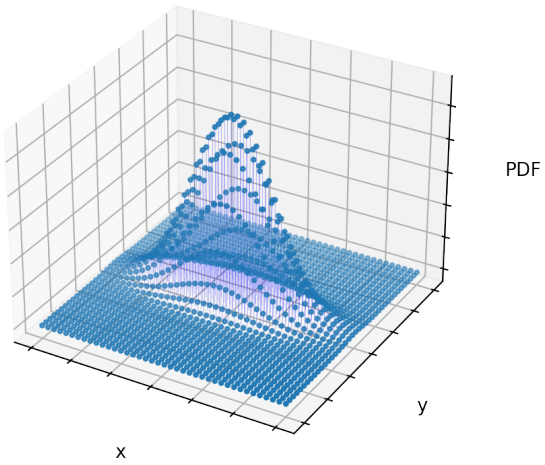


Figure 7: Eliptical Gaussian with $r_1 = 0.5$ and $r_2 = 4$. The level sets $z = k$ constitute ellipses.

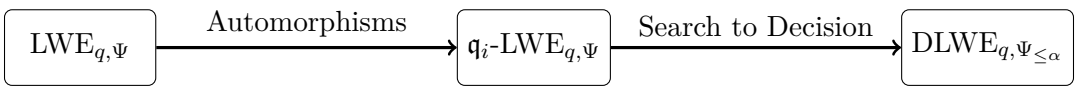


Figure 8: Schematic of reductions from R-LWE to R-DLWE