

University of Technology Darmstadt
Department of Computer Science
Cryptography and Computeralgebra

A Bachelor Thesis

April 2007

On Lattices, Codes and Regev's Cryptosystem



Krasimira Pancheva

University of Technology Darmstadt
Department of Mathematics

Supervised by Prof. Dr. Johannes Buchmann,
Richard Lindner

Contents

Introduction	1
1 Preliminaries	3
1.1 General	3
1.2 Lattices	4
1.3 Learning with errors	6
1.4 Several Distributions	8
2 Main Theorem	11
3 Regev's Cryptosystem	15
3.1 Example	15
3.2 Formal Definition	16
3.3 Correctness	16
3.4 Security	17
3.5 Tight bounds	19
Conclusion	20

Introduction

Public-key cryptosystems were invented in the late 1970's, with some help from the development of complexity theory around that time. Previous cryptosystems required that any two users share a secret key if they want to communicate securely, whereas in the new public-key paradigm, encryption and decryption used different keys: an encryption key which is public, and a decryption key which is kept secret. With the public key one could encrypt messages, and decrypt them with the private key. Thus the owner of the private key would be the only one who could decrypt the messages, but anyone knowing the public key could send them in privacy. The basic idea of the construction of the new cryptosystem which is still the most important ingredient of any public-key cryptosystem nowadays, is a difficult computational problem. Thus the security of the system is based on the fact that the private key can be computed from the public key only by solving such a problem, for example factoring large integers. In other words, implementation can be done efficiently, but cannot be cracked efficiently. The wide interest in public-key cryptography has produced several practically important cryptosystems. For instance, the Rabin cryptosystem where deciphering messages is provably as hard as factoring large integers of the form $N = pq$, p and q are primes. ElGamal presented another cryptosystem which is secure provided that the so-called discrete logarithm problem is computationally intractable. The most commonly used public-key algorithm RSA relies on the problem of factoring as well as Rabin and the Diffie-Hellmann key-exchange protocol is built on a problem related to discrete logarithms. In addition, there are some other cryptosystems that rely on different mathematical problems. There are only a few interesting knapsack public-key cryptosystems, none of which are of practical importance, i.e. the Merkle-Hellmann and Chor-Rivest cryptosystem, both constructed on the ground of a combinatorial problem known as knapsack problem.

Unfortunately, these cryptosystems and many others have either been shown to be impractical or to be insecure, some have been cracked already like the Merkle-Hellmann which was broken in 1980. This was the main reason for the scientists to look for a new type of cryptosystems relying on different security assumptions. As a result of this search, in the recent years large interest has been directed to lattice-based cryptosystems. One of the reasons is that certain lattice-problems are NP-hard, and several efficient cryptosystems have been proposed and appeared strong until they have been broken. For example, the Goldreich-Goldwasser-Halevi, Ajtai-Dwork cryptosystems.

The goal of this thesis is to present a new public-key cryptosystem, with security based on the worst-case quantum hardness of SVP (Shortest Vector Problem) and SIVP (Shortest Independent Vector Problem). Although Regev's cryptosystem is quite similar to that of Ajtai-Dwork [3], the one by Ajtai was based on unique-SVP, just a special case of SVP, hardness of which is not so well understood. Regev has used a reduction from worst-case lattice problems such as SVP and SIVP to a certain learning problem as a base for his cryptosystem. Hence, an efficient solution implies a quantum algorithm for SVP and SIVP. The most important feature of this cryptosystem is its improved efficiency. Comparing it

to the previous one by Ajtai and Dwork reveals that in the new one the public key size was reduced to $\tilde{O}(n^2)$ from $\tilde{O}(n^4)$ and the encryption increases the size of messages by a factor of $\tilde{O}(n)$, while before it was $\tilde{O}(n^2)$. Moreover, under the assumption that all parties share a random bit string of length $\tilde{O}(n^2)$, the size of the public key can be reduced to $\tilde{O}(n)$. These facts make the cryptosystem practical, although it does not quite solve the open problem of basing a cryptosystem on the worst-case hardness of SVP and SIVP, because of its quantum aspect. Finally, the cryptosystem is classical, it is only the security proof that needs some quantum computation, which we skip in this thesis for simplicity. The rest of the thesis is organized as follows. Chapter 1 treats the basic, general definitions of lattices, learning with errors problem and a brief discussion of the best known algorithm for that problem, as well as some definitions of a few distributions. Chapter 2 describes the main theorem of Regev and his algorithm for finding short linearly independent lattice vector. Finally, chapter 3 presents the cryptosystem itself, a simple example and briefly explains the security and correctness lemmas. At the end of the chapter there is a discussion of the settings of the parameters of the Regev's system.

The following is just a list of the parameters of the cryptosystem that we will use all the time. The reader might find it helpful to have a look at them, any time needed:

- n : some integer, $n \geq 1$
 - p : be some prime integer, such that $\text{poly}(n) \leq p(n) = p \geq 2$
and $n^2 < p < 2n^2$, $p = O(n^2)$
 - α : be some real, $\alpha = \alpha(n) \in (0, 1)$ such that $\alpha > \frac{2\sqrt{n}}{p}$
 - m : be some integer, $m = \text{poly}(n) = 5(n+1)(1+2\log n)$,
 - χ : be some probability distribution, $\chi : \mathbb{Z}_p \rightarrow \mathbb{R}^+$.
- In the Regev's cryptosystem it is taken to be $\bar{\psi}_\alpha$.

1 Preliminaries

In this section we present some useful notations that will be used throughout the whole thesis.

1.1 General

We define a function $f(n)$ to be called *negligible amount* in n if $f(n)$ is asymptotically smaller than n^{-c} for any constant $c > 0$. (We mean that $f(n)$ is asymptotically smaller or negligible than $g(n)$ if $f(n) = o(g(n))$. Instead of writing $f(n)$, we will use sometimes just n .) For example let $f(n) = 2n$ and $g(n) = n^2$. So in this case we have $2n = o(n^2)$. Note that the parameter n is an integer and indicates the input size. Similarly, a *non-negligible amount* is one which is at least n^{-c} for some constant $c > 0$. Thus, we define $f(n)$ in n to be a *negligible function*, if $\lim_{n \rightarrow \infty} n^c f(n) = 0$ for any $c > 0$. In other words, a function is negligible if $f(n) < 1/n^{-c}$ for any $c > 0$ and sufficiently large n , defined in the sense above, i.e. $f(n) = 1/2^n$ is a possible example. Also, when we say that an expression is *exponentially small* in n we mean that it is at most $2^{-\Omega(n)}$ and by an expression is *exponentially close to 1*, we mean that it is $1 - 2^{-\Omega(n)}$. Finally, we should say what is hidden behind an expression, in our application it is a number, corresponding to some probability.

Example :

Let n be a negligible amount and $(99/100)^n$ be an expression, then we can conclude that this expression is exponentially small, i.e.

$(99/100)^n = 99^n \cdot 1/(2^2 \cdot 25)^n = (99/25)^n \cdot 2^{-2n} = 2^{-\Omega(n)}$. Here the last equation holds because of the definitions of Ω (which can be found on next page) and the fact that for any polynomial function in n , $f(n) = \sum_{i=0}^m a_i n^i$ with $a_m > 0$ we can always "drop" the less significant terms (i.e. terms involving powers of n which are less than m), as well as the leading coefficient a_m . It holds $f(n) = \Omega(n^m)$.

We need some technical explanations, as well. For $x \in \mathbb{R}$, $\lfloor x \rfloor$ is just the integer closest to x or, in case two such exist, it is the smaller one. For two real numbers x and $y > 0$, $x \bmod y = x - \lfloor x/y \rfloor y$, where $\lfloor x \rfloor$ is just the smallest integer closer to x .

We say that an algorithm W with oracle access is a *distinguisher* between two distributions if the difference of its acceptance probability, when the oracle outputs samples of the first distribution and its acceptance probability, when the oracle outputs samples of the second distribution is a non-negligible amount.

Another important notation used mainly in the last chapter of security is that of a *statistical distance*. Thus given two probability density functions ϕ_1, ϕ_2 on \mathbb{R}^n , the statistical distance between them is defined as:

$$\Delta(\phi_1, \phi_2) = \int_{\mathbb{R}^n} |\phi_1(x) - \phi_2(x)| dx$$

Notice that with this definition, the statistical distance ranges in $[0, 2]$. It cannot increase by applying a possible random function f , i.e.

$$\Delta(f(X), f(Y)) \leq \Delta(X, Y)$$

This implies that the acceptance probability of any algorithm on inputs X differs from its acceptance probability on inputs from Y by at most $\frac{1}{2} \Delta(X, Y)$.

The table coming next is just a revision of the most common complexity classes we use:

Notation	Definition
$f(n) \in O(g(n))$	$\exists c, k > 0$ such that $0 \leq f(n) \leq cg(n)$ for all $n \geq k$
$f(n) \in \Omega(g(n))$	$\exists c, k > 0$ such that $0 \leq f(n) \geq cg(n)$ for all $n \geq k$
$f(n) \in o(g(n))$	for all $c > 0 \exists$ some $k > 0$ such that $0 \leq f(n) < cg(n)$ for all $n \geq k$
$f(n) \in \Theta(g(n))$	$0 < \liminf_{n \rightarrow \infty} \left \frac{f(n)}{g(n)} \right \leq \limsup_{n \rightarrow \infty} \left \frac{f(n)}{g(n)} \right < \infty$
$f(n) \in \tilde{O}(g(n))$	<i>if</i> $\exists a, c \geq 0 \mid f(n) \leq ag(n)\log^c g(n)$ for all suff. large n

Last but not least, recall the definition of the cyclic group \mathbb{Z}_p of integers with addition modulo p . This is the group $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ with p elements and identity element 0. We need to know the basic notation of group theory and the properties of \mathbb{Z}_p as we use them as a basic ground in the whole thesis. Finally, we explain the abbreviation *poly*(n). It means just a polynomial in n .

1.2 Lattices

In this subsection we characterize some basic definitions related to lattices.

Definition 1. (Lattice): An n -dimensional **lattice** is the set of all linear combinations

$$\Lambda := \left\{ \sum_{i=1}^n b_i x_i \mid x_i \in \mathbb{Z} \text{ for } 1 \leq i \leq n \right\}$$

of n linearly independent vectors $b_1, \dots, b_n \in \mathbb{R}^m$.

The set b_1, \dots, b_n is called a basis for the lattice and can be represented as a matrix as well: $B = [b_1, \dots, b_n] \in \mathbb{R}^{m \times n}$ having the basis vectors as columns. For simplicity, we denote the matrix generated by B with the same letter as the lattice itself, $\Lambda := \{Bx : x \in \mathbb{Z}^n\}$, where Bx is the usual matrix-vector multiplication.

We say the *rank* of the lattice is n and its *dimension* is m . If $n = m$, the lattice is called a *full-rank lattice*.

Definition 2. (The Fundamental Parallelepiped):

Given a basis B of a lattice Λ , the fundamental parallelepiped is defined as

$$\mathbb{P}(B) := \{Bx \mid x \in [0, 1]^n\}$$

Note that the lattice has a different fundamental parallelepiped for every possible basis.

Definition 3. (Determinant of a lattice): Let $B \in \mathbb{R}^{m \times n}$ be a lattice-basis. The determinant of a lattice is defined as the n -dimensional volume of the fundamental parallelepiped associated to B:

$$\det(\Lambda(B)) := \text{vol}(\mathbb{P}(B))$$

Notice that when $B \in \mathbb{R}^{n \times n}$ is a non-singular square matrix then $\det(\Lambda(B)) = |\det(B)|$.

Definition 4. (The dual lattice): The dual of a lattice Λ in \mathbb{R}^n , is the set of all vectors $y \in \mathbb{R}^n$ whose inner product with any of the vectors in Λ is an integer. More formally,

$$\Lambda^* := \{y \in \mathbb{R}^n | \forall x \in \Lambda, \langle x, y \rangle \in \mathbb{Z}\}$$

Definition 5. (Span): The span of a lattice $\Lambda(B)$ is the linear space spanned by its vectors,

$$\text{span}(\Lambda(B)) = \text{span}(B) = \{By | y \in \mathbb{R}^n\}$$

More, there are two main lattice problems that are interesting for our topic of discussion, the shortest vector problem (SVP) and the closest vector problem (CVP). Next, we give the exact definitions of both problems. For more details and further discussion on lattice problems see [10],[12].

Definition 6. (The 1st successive minimum) : It is the minimum distance between any two distinct lattice vectors, and equals the length of the shortest non-zero lattice vector (where by length we mean the Euclidean norm). It is denoted by $\lambda_1(\Lambda(B))$. Formally,

$$\lambda_1(\Lambda(B)) := \min \{ \text{dist}(x, y) : x \neq y \in \Lambda \} = \min \{ \|x\|_2 | x \in \Lambda \setminus \{0\} \}$$

In other words, given a lattice the first successive minimum denotes the radius of the smallest ball centered at the origin, which contains a non-zero lattice vector. Likewise λ_2 denotes the radius of the smallest ball containing two linearly independent non-zero vectors, and is called the second minimum of the lattice. In general, the i -th minimum λ_i of a lattice is the radius of the smallest ball containing i linearly independent non-zero lattice vectors (we have to ask for a non-zero vector since the zero vector is always contained in a lattice and its norm is zero). The formal definition is the following:

Definition 7. (The i -th successive minimum) : Let Λ be a lattice of rank n . For $i \in \{1, \dots, n\}$ the i -th successive minimum is

$$\lambda_i(\Lambda(B)) := \inf \{ r : \dim(\text{span}(\Lambda \cap \overline{B}(0, r))) \geq i \}$$

where $\overline{B}(0, r) = \{x \in \mathbb{R}^m | \|x\| \leq r\}$ is the closed ball of radius r around 0. Next comes the problem concerned with finding (or approximating) $\lambda_1(\Lambda)$:

Definition 8. (The Shortest Vector Problem or SVP): Given a lattice basis $B \in \mathbb{Z}^{m \times n}$ find $v \in \Lambda(B)$ such that $\|v\|_2 = \lambda_1(\Lambda(B))$. This is the Euclidean norm, $\|v\|_2 = \sqrt{\sum v_i^2}$.

In the 2-dimensional case, it is possible to locate short vectors in polynomial-time, using the generalized Gauss algorithm [12]. Unfortunately, no good algorithm for finding short vectors in n -dimensional lattices is known (even for specific norms like the euclidean norm). Still there is an approximation algorithm developed by A.K. Lenstra, H.W. Lenstra, Jr. and L. Lovasz, usually called **LLL** algorithm. Note that an approximation algorithm does not necessarily

find the shortest vector in the lattices, but it computes a lattice vector that is guaranteed to be at most, let say γ times the length of the shortest, where γ is the approximation factor. The best known approximation factor is $\gamma = (\frac{2}{\sqrt{3}})^n$. For even better approximation factors, one can use Schnorr's algorithm. It can be proved that the **LLL** algorithm terminates and it is actually polynomial time.

The slightly modified problem of the SVP is called the shortest independent vector problem. It is concerned with finding the i -th successive minimum of a lattice, $\lambda_i(\Lambda(B))$ and will be used in this thesis, defined as follows:

Definition 9. (The Shortest Independent Vector Problem or SIVP):

For a rank n lattice Λ , let $\lambda_n(\Lambda)$ denote the minimum length of a set of n linearly independent lattice vectors from Λ , where the length of a set is defined as the length of the longest vector in it. The goal is to find a set of n linearly independent vectors whose length is at most $poly(n)\lambda_n(\Lambda)$.

Finally, we define the second problem concerned with lattices, which is not of less importance:

Definition 10. (The Closest Vector Problem or CVP): Given a basis B and a target vector t , find the lattice vector $v \in \Lambda(B)$ closest to t .

CVP is **NP**-hard. However there is a polynomial time approximation algorithm for it, called Babai's nearest plane algorithm, see [4] for more detailed discussion. It obtains approximation ratio of $2(\frac{2}{\sqrt{3}})^n$, where n is the rank of the lattice. In many applications, this algorithm is applied for a constant n , in which cases it yields a constant approximation factor. In fact, Regev presents a factor of $\gamma = 2^{\frac{n}{2}}$ in [15].

Indeed, he defines a non-standard variant of the closest vector problem, which he needs for his Main theorem. It is given and used as follows:

Definition 11. (Approximate CVP): Let Λ be n -dimensional lattice and $d > 0$ such that $d \leq \lambda_1(\Lambda)/2$ holds. Given a point $x \in \mathbb{R}^n$ whose distance to Λ is at most d find the closest lattice point y to x .

In the rest of the thesis, d satisfies always the conditions above hence, in this variant the closest vector is unique.

1.3 Learning with errors

The LWE problem can be presented as the problem of decoding random linear codes. More specifically, we bring out the definitions we are going to use:

Definition 12. (Decoding random linear codes): Let $m = poly(n)$ be arbitrary. Given a random $m \times n$ matrix $Q \in \mathbb{Z}_p^{m \times n}$ and a vector $t \in \mathbb{Z}_p^m$, such that $t = Qs + e \pmod{p}$ where coordinate of the error vector $e \in \mathbb{Z}_p^m$ is chosen independently according to a probability distribution. Recover s .

Usually the 'learning from parity with errors' modulo p problem is defined formally in the following way:

Definition 13. ($LWE_{p,\chi}$): Let n be some integer and $\epsilon \geq 0$ be some real. $p = (p) \leq poly(n)$ is also some prime integer. We are given a list of equations with error:

$$\begin{aligned}
b_1 &= \langle s, a_1 \rangle + e_1 \pmod{p} \\
b_2 &= \langle s, a_2 \rangle + e_2 \pmod{p} \\
&\vdots
\end{aligned}$$

where a_i are uniformly distributed on \mathbb{Z}_p^n and $b_i \in \mathbb{Z}_p$. The errors $e_i \in \mathbb{Z}_p$ are chosen independently according to a probability distribution $\chi : \mathbb{Z}_p \rightarrow \mathbb{R}^+$. Then the input of the problem consists of the pairs (a_i, b_i) and the output is a guess for s .

Informally speaking, the problem is to find s from samples of the form $(a, sa + e)$ where a is chosen independently and uniformly from \mathbb{Z}_p^n and e is chosen from \mathbb{Z}_p .

The special case $e = 0$ can be solved efficiently with Gaussian elimination. With error ϵ and $p = 2$, we have $\chi(0) = 1 - \epsilon$ and $\chi(1) = \epsilon$, i.e. each equation is independently chosen to be correct with probability $1 - \epsilon$ and incorrect with probability ϵ . This requires $O(n)$ equations and $poly(n)$ time. The problem becomes significantly harder when we take any positive $\epsilon > 0$.

The first algorithm for $p = 2$ is due to Blum, Kalai and Wassermann [5]. It is based on the following idea:

Find a set S of $O(\sqrt{n})$ equations among $2^{O(n/\log n)}$, such that $\sum_S a_i = (1, 0, \dots, 0)$. This gives us a guess for the first bit of s which is correct with probability $\frac{1}{2} + 2^{-O(\sqrt{n})}$. By repeating this process $O(\sqrt{n})$ we will get the correct value for the first bit of s with high probability. Repeating further the whole procedure, we can determine the remaining bits of s with high probability.

This algorithm is subexponential. It requires only $2^{O(n/\log n)}$ equations and runs in time $2^{O(n/\log n)}$. It is currently the best known algorithm for the LWE problem with $p = 2$. Moreover it runs in polynomial time for the case when all non-zero bits of $b = (b_1, b_2, \dots)$ are one of its first $O(\log n \log \log n)$ bits. It is comparatively faster than the maximum likelihood algorithm which works in a similar way but needs $O(n)$ equations and runs in $2^{O(n)}$. This 'learning' algorithm of Blum, Kalai and Wasserman [5] has other applications, too. Kumar and Sivakumar [7] present an algorithm that in $2^{O(n)}$ time approximates the length of the shortest lattice vector to within a polynomial factor. Their algorithm uses Ajtai's reduction [1] of the problem of approximating the length of the shortest lattice vector in a family $\Lambda(n, m, q) = L(A)$ of lattice of the random class. Ajtai defines a lattice $L(A)$ of the random class in the following way: for an $n \times m$ matrix A over \mathbb{Z}_q , $L(A)$ is the lattice of all vectors in \mathbb{Z}^m which are orthogonal to the rows of A . He shows that if there exist an algorithm \mathbf{C} , that for certain values of q and m , computes a non-zero vector in $L(A)$ of certain length, depending on m , then there exist an algorithm \mathbf{B} , that for any lattice $L \in \mathbb{R}^n$, computes approximation of the length of the shortest lattice vector in L up to a factor in $poly(n)$.

So the last step for approximating the SVP-length is to construct an algorithm \mathbf{C} for finding a non-zero vector of certain length, depending on n , in $\Lambda(A)$. Using the idea of Blum, Kalai and Wasserman, as well as the one of Kumar and Sivakumar, we use the following procedure to construct \mathbf{C} : Let A be an $n \times m$ matrix over \mathbb{Z}_q . The multi-set that consists of columns of A gives us a uniform sample of m vectors in \mathbb{Z}_q^n . Then we can obtain a presentation of any vector $u \in \mathbb{Z}_q^n$ as a sum of at most n^ϵ vectors from S . Considering the case $u = 0$ we can find a non-zero vector in $\Lambda(A)$.

Thus the best known algorithm for the LWE problem with $p \leq \text{poly}(n)$ resembles the one presented by Blum, Kalai and Wassermann [5] and under some assumptions on the probability distribution uses $2^{O(n)}$ equations and runs in the same time.

1.4 Several Distributions

Next, we proceed with definitions of the distributions we need in the thesis. We start with the simplest continuous distribution, namely the *uniform distribution*. It has a constant probability density on given (real) interval and is zero elsewhere. Its probability density function on \mathbb{R} is : $1/(b - a)$ for $x \in (a, b)$ and otherwise 0. We will use the letter U for the uniform distribution on a segment (0,1) with density function 1, for elements in the segment and 0 otherwise. Then we recall the *normal distribution*. It is the distribution on \mathbb{R} with mean $\mu = 0$ and variance σ^2 given by the density function

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{1}{2}((x - \mu)/\sigma)^2)$$

Note that the sum of two independent normal variables with mean 0 and variance σ_1^2 and σ_2^2 is a normal variable with mean 0 and variance $\sigma_1^2 + \sigma_2^2$. In the following definitions we see some modified distributions that are based on the preceding paragraph.

Definition 14. (Gaussian function $\rho_s(x)$): A Gaussian function centered in c and scaled by a factor of s is defined in the following way, where we consider only the special case of $c = 0$:
So, for a vector $x \in \mathbb{R}^n$ and any $s > 0$, let

$$\rho_s(x) = \exp(-\pi \|x/s\|^2)$$

The total measure to $\rho_s(x)$ is $\int_{x \in \mathbb{R}^n} \rho_s(x) dx = s^n$. Therefore, we can define the continuous Gaussian distribution around 0 with parameter s with its probability density function

Definition 15. (The continuous Gaussian distribution $D_s(x)$):

$$\forall x \in \mathbb{R}^n, D_s(x) = \frac{\rho_s(x)}{s^n}$$

It is an n -dimensional density function and intuitively one can think of it as a sphere of radius $r = s\sqrt{n/(2\pi)}$ centered around 0. The smaller s is, the more concentrated is the probability around 0.

Notice that $D_s(x)$ can be expressed as the sum of n orthogonal 1-dimensional Gaussian distributions and each of them can be efficiently approximated with arbitrary precision using standard techniques. So, the distribution $D_s(x)$ can be efficiently approximated as well. For simplicity, we work with real numbers and assume that we can sample from $D_s(x)$ exactly.

If s is not specified, assume that it is 1 and functions are extended to sets in the usual way, that is why for any countable set A we have $\rho_s(A) = \sum_{x \in A} \rho_s(x)$.

Definition 16. (The discrete Gaussian distribution $D_{\Lambda,s}(x)$):

The discrete Gaussian distribution for any real $s > 0$ and a countable set Λ ,

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

As Micciancio and Regev show in their paper [11], for a large enough s , $D_{\Lambda,s}(x)$ behaves in many respects like a continuous Gaussian distribution $D_s(x)$. In fact, they define a new parameter that tells how big s has to be in order for this to happen. So, in the next paragraph we will give the definition of this parameter and some useful properties. For more details and explicit description, have a look at [11],[13].

Definition 17. (The smoothing parameter $\eta_\epsilon(\Lambda)$):

For an n -dimensional lattice Λ , and a positive real number $\epsilon > 0$, we define its smoothing parameter $\eta_\epsilon(\Lambda)$ to be the smallest s such that $\rho_{1/s}(\Lambda^* \setminus \{0\}) \leq \epsilon$.

Notice that $\eta_\epsilon(\Lambda)$ is a continuous and strictly decreasing function of ϵ . It is the smallest s such that a Gaussian measure $\rho_{1/s}$ on the dual lattice scaled with $1/s$ gives all but a negligible amount of its weight to the origin.

Informally, it says if we choose a random lattice point from an n -dimensional lattice Λ and add noise $D_s(x)$ for some $s > \eta_\epsilon(\Lambda)$ then the resulting distribution is within statistical distance ϵ (where we take ϵ to be some negligible function of the lattice dimension n) of the uniform distribution on \mathbb{R}^n . In fact, Regev shows another important property in [13]: For $s > \sqrt{2}\eta_\epsilon(\Lambda)$ if we sample a point from the discrete Gaussian distribution $D_{\Lambda,s}(x)$ and add Gaussian noise $D_s(x)$, we obtain distribution whose statistical distance to a continuous Gaussian is at most 4ϵ . Hence, intuitively the noise is enough to hide the discrete structure of $D_{\Lambda,s}(x)$. (Here we should make clear that no uniform probability distribution can be defined over a lattice or other countably infinite sets or even over the entire space. In fact the definitions that we just made follow [9] and capture the intuition of 'starting from' a random lattice point by working modulo the lattice.)

So what we want to do next is to define another probability distribution. We call it the error distribution and it is the distribution that we need in Regev's cryptosystem .

Definition 18. (The error distribution $\psi_\alpha(x)$):

For $\alpha \in \mathbb{R}^+$, it is a distribution on \mathbb{R}/\mathbb{Z} obtained by sampling from a normal variable with mean $\mu = 0$ and standard deviation $\sigma = \frac{\alpha}{\sqrt{2\pi}}$, reducing the result modulo 1. In fact one can (one must) identify the circle group \mathbb{R}/\mathbb{Z} with the interval $[0, 1)$. (The circle group shows up in a huge variety of forms in mathematics and we list some of the most common ones:

$\mathbb{R}/\mathbb{Z} \cong \mathbb{C}_1 := \{x + iy | x, y \in \mathbb{R}, x^2 + y^2 = 1\} = \{z \in \mathbb{C} | \|z\| = 1\}$, or it is just what we said, the unit circle.) Taking the parameters of the normal distribution with mean 0 and standard deviation $\sigma = \frac{\alpha}{\sqrt{2\pi}}$ and replacing them in the formula of the density function for a normal distribution, we can write the explicit formula for the distribution $\psi_\alpha(x) : \mathbb{R}/\mathbb{Z} \rightarrow \mathbb{R}^+$:

$$\psi_\alpha(x) := \sum_{k=-\infty}^{\infty} \frac{1}{\alpha} \exp(-\pi(\frac{x-k}{\alpha})^2)$$

Note that k is an integer running from minus infinity to plus infinity and as we said above that x is taken from the interval $[0,1)$. As you see, one can efficiently sample from $\psi_\alpha(x)$, as well. We want to choose from a normal distribution, but instead the actual result should be an integer modulo p . So, thinking

intuitively we choose (or precise sample) from the usual (real valued) normal distribution, then round the result to the nearest integer and at the end take modulo p . The actual 'rounding' is presented through the following function $\bar{\psi}_\alpha(i) : \mathbb{Z}_p \rightarrow \mathbb{R}^+$,

$$\bar{\psi}_\alpha(i) := \int_{(i-1/2)/p}^{(i+1/2)/p} \psi_\alpha(x) dx$$

where i goes from 0 to $p - 1$. In other words we are rounding numbers in $[-1/2p, 1/2p]$ to 0, those in $[1/2p, 3/2p]$ to 1, etc. We round to integers in $\{0, 1, \dots, p - 1\}$ instead of to $\{0, 1/p, \dots, p - 1/p\}$, which is more convenient, but still one can do it also for i in $\{0, 1/p, \dots, p - 1/p\}$. Note that the elements of \mathbb{Z}_p are arranged on the unit circle and rounding can be applied to any distribution, not only to ψ .

We needed this distribution for the cryptosystem, presented in chapter 3. Therefore let $p \geq 2$ be some integer, and let $\chi : \mathbb{Z}_p \rightarrow \mathbb{R}^+$ be some probability distribution on \mathbb{Z}_p . For n an integer and $s \in \mathbb{Z}_p^n$, a vector, let $A_{s,\chi}$ be a distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$ obtained by choosing a vector $a \in \mathbb{Z}_p^n$ uniformly at random, choosing $e \in \mathbb{Z}_p$ according to χ , and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$, where additions are performed in \mathbb{Z}_p , i.e. modulo p . Actually, a different letter for that distribution is used, because it is the common case, i.e. it can most probably be any other distribution, but Regev proved it for a normal one. We also define U as the uniform distribution on $\mathbb{Z}_p^n \times \mathbb{R}/\mathbb{Z}$.

Similarly, for a probability density function ψ on \mathbb{R}/\mathbb{Z} , we can define $A_{s,\psi}$ as the distribution on $\mathbb{Z}_p^n \times \mathbb{R}/\mathbb{Z}$ obtained by choosing a vector $a \in \mathbb{Z}_p^n$ uniformly at random, choosing $e \in \mathbb{R}/\mathbb{Z}$ according to ψ , and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle / p + e)$, where additions are performed in \mathbb{R}/\mathbb{Z} , i.e. modulo 1.

We will need some knowledge about Fourier transformation of a function and that is why we briefly review some of the important properties of the Fourier transform. (The following is used mainly in the proof of the main theorem of Regev, which is not of great interest for our topic of discussion, which is concentrated on the practical example of it Regev's cryptosystem). We start with a definition:

Definition 19. (The Fourier transform \hat{f}):

The Fourier transform of a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ is defined to be a function $\hat{f} : \mathbb{R}^n \rightarrow \mathbb{R}$ such that

$$\hat{f}(y) = \int_{\mathbb{R}^n} f(x) e^{(-2\pi i \langle x, y \rangle)} dx$$

Note that if $f(x) = g(x + v)$ for some function g and vector v then

$$\hat{f}(y) = e^{2\pi i \langle v, y \rangle} \hat{g}(y)$$

Similarly, if f is related to g by $f(x) = e^{2\pi i \langle x, v \rangle} g(x)$ then

$$\hat{f}(y) = \hat{g}(y - v)$$

Another important fact is that the Gaussian function $\rho_s(x)$ with $s = 1$ is its own Fourier transform, i.e.

$$\widehat{\rho}_s(y) = \rho_s(x) = e^{-\pi\|x\|^2}$$

More generally, for any $s > 0$ it holds that

$$\widehat{\rho}_s(y) = s^n \rho_{1/s}(x)$$

And finally we give the following formulation of the Poisson summation formula.

Lemma 1. (The Poisson summation formula): For any lattice Λ and any function $f : \mathbb{R}^n \rightarrow \mathbb{C}$, $f(\Lambda) = \det(\Lambda^*) \widehat{f}(\Lambda^*)$ where \widehat{f} denotes the Fourier transform of f .

For a more precise treatment, see [6] and [13].

2 Main Theorem

First we want to give some informal explanations of the problem and then we will define it properly:

Let n, p be integers and $\alpha \in (0, 1)$ be some real such that $\alpha > (2\sqrt{n})/p$. If there exists a polynomial time algorithm that solves LWE_{p, ψ_α} then there exists a quantum algorithm that approximates the shortest vector problem SVP and the shortest independent vector problem SIVP to within $\tilde{O}(n/\alpha)$ in the worst case. In other words, it shows that the learning modulo p problem (e.g. the problem of finding small solutions to random linear equations with coefficients in \mathbb{Z}_p^n) is as hard as the problem of reducing worst-case instances of lattice approximation problems (e.g. finding short lattice vectors or SVP). So, in order to perform such a reduction, one needs to sample (almost uniformly at random) the group \mathbb{Z}_p^n in a way that is related to an underlying lattice problem (for an arbitrary lattice). Before continuing with a more detailed description of the reduction algorithm we first give the exact definition of the main theorem:

Theorem 1. The Main Theorem

Let $\epsilon = \epsilon(n)$ be some negligible function of n and $\alpha = \alpha(n) \in (0, 1)$ some real. Also let $p = p(n)$ be some integer such that $p > (2\sqrt{n})/\alpha$. Assume there exists an efficient algorithm W that solves LWE_{p, ψ_α} , then there exists an efficient quantum algorithm for solving the following worst-case lattice problems:

- Find a set of n linearly independent lattice vectors of length at most $2n(\eta_\epsilon(\Lambda)/\alpha) \leq \tilde{O}(\lambda_n(\Lambda)(n/\alpha))$.
- Approximate $\lambda_1(\Lambda)$ to within $\tilde{O}(n/\alpha)$.

The resulting algorithm for the approximation of the SIVP is iterative. It starts with some long vectors and repeatedly finds shorter and shorter ones. It proceeds as follows:

Take n^c samples from the discrete Gaussian probability distribution $D_{\Lambda, r}$ for some large enough r and $c > 0$. Each iteration takes n^c samples and generates n^c samples from a new distribution $D_{\Lambda, r'}$ over the same lattice with $r' = r\sqrt{n}/\alpha p$. Hence $r' < r$, by repeating this iterative step we obtain samples from $D_{\Lambda, r}$ for smaller and smaller values of r . The algorithm stops when no longer n linearly independent vectors of length at most $r\sqrt{n}$ can be found. We do this in two steps:

First step: Using the given samples from the distribution mentioned above with parameter r , construct an algorithm that solves the CVP on Λ^* for points that are within distance $\alpha p/r$ of the lattice. This algorithm is classical and uses the LWE oracle W . (The reader can have a glimpse of the exact definition of the closest vector problem in chapter 1 Preliminaries, section 1.2 Lattices).

It turns out that the $CVP_{\Lambda^*, \alpha p/r}$ can be reduced to the following problem:

Given some input vector x within distance at most $\alpha p/r$ of Λ^* . Find the vector of coefficients $y \in \Lambda^*$ reduced modulo p . For our choice of r , y is unique and exactly the closest vector to x . Even more formally, our goal is to find:

$$\tau(x) := (\Lambda^*)^{-1}y \pmod p = \Lambda^T y \pmod p \in \mathbb{Z}_p^n$$

in case x is given.

We start with approximating the Fourier transform of the Gaussian distribution $D_{\Lambda, r}$. This was shown to be possible by Aharonov and Regev in [14]. In fact, if we are given a polynomial number of samples from some probability distribution on a lattice, they showed that one can compute an approximation to its Fourier transform to within $\pm 1/\text{poly}(n)$. So, taking our n^c samples from the distribution $D_{\Lambda + \Lambda a, r/p}$ gives us the desired good approximation. This is done in the following way:

- Choose $\mathbf{a} \in \mathbb{Z}_p^n$ uniformly at random
- Sample \mathbf{y} from $D_{\Lambda + \Lambda a, r/p}$ and output (\mathbf{a}, \mathbf{y})

Indeed, the Fourier transform of the Gaussian distribution $D_{\Lambda + \Lambda a, r/p}$ is the Fourier transform of $D_{\Lambda, r}$ up to a phase (i.e. meaning the phase component of an expression, or to make it even clearer it is just a component of an expression). So, if we can approximate the Fourier transform of Gaussian distribution over $\Lambda + \Lambda a$ with a parameter of r/p , we can then correct its phase and obtain the Fourier transform of Gaussian distribution over Λ with parameter r/p . The correction is done as follows:

Fix some x whose distance to Λ^* is smaller than $\alpha p/r$, for example let it be $\beta p/r$ for some $0 \leq \beta \leq \alpha$. One of the main contributions in [13] is that for such x , the distribution of $\langle x, y \rangle \pmod 1$, where y is sampled from $D_{\Lambda + \Lambda a, r/p}$ is closed to a Gaussian centered around $\langle a, \tau(x) \rangle / p \pmod 1$ with standard deviation β . Here Regev makes an important note, namely the distribution $\langle x, y \rangle \pmod 1$ is discrete and it is close to a Gaussian, i.e. by adding a small amount of noise, $D_{s(x)}$, the distribution becomes a continuous distribution that is very close to a Gaussian, or normal distribution. In other words, adding a small amount of noise to $\langle x, y \rangle \pmod 1$ will be equal to $\langle a, \tau(x) \rangle / p \pmod 1$ with very high probability. Therefore if \mathbf{y} is sampled from $D_{\Lambda + \Lambda a, r/p}$ then $p \langle x, y \rangle \pmod p$ is distributed like a Gaussian around $\langle a, \tau(x) \rangle \pmod p$ with standard deviation βp . By rounding, we obtain $\lfloor p \langle x, y \rangle \rfloor \pmod p$ is distributed like $\bar{\psi}_\beta$ around $\langle a, \tau(x) \rangle \pmod p$. This can be rewritten as 'equation with error':

$$\begin{aligned} \lfloor p \langle x, y \rangle \rfloor &= \langle \tau(x), a \rangle + e_1 \pmod p \\ \lfloor p \langle x, y \rangle \rfloor &= \langle \tau(x), a \rangle + e_2 \pmod p \\ &\vdots \end{aligned}$$

Using the LWE oracle we solve the polynomial number of equations and obtain $\tau(x)$. So, after all this we are able to make the correction in the phase and to obtain the Fourier transform of the Gaussian distribution over Λ within parameter r/p .

Actually, there is another remark we need to add here: we want to ask what exactly is $D_{\Lambda+\Lambda a, r/p}$? The answer to this question is the following: it is a distribution that looks like a translation of $D_{\Lambda, r/p}$, where Λ has been partitioned into p^n translates of the lattice $p\Lambda$. Namely, for each $a \in \mathbb{Z}_p^n$ the set $p\Lambda + \Lambda a = \{\Lambda b | b \in \mathbb{Z}^n, b \bmod p = a, a \in \mathbb{Z}_p^n\}$ forms a partition of Λ . In [13] Regev shows that for any $a \in \mathbb{Z}_p^n$, the probability that a point sampled from $D_{\Lambda, r}$ is in $p\Lambda + \Lambda a$ is very close to p^{-n} .

Second step: Use the algorithm from the first step to generate $poly(n)$ samples from $D_{\Lambda, r'}$. Remember that $r' = r\sqrt{n}/(\alpha p)$. It proceeds as follows: Create a quantum state that corresponds to the Fourier transformation of the Gaussian distribution over Λ with a parameter r/p . Then apply the quantum Fourier transform and obtain a quantum state corresponding to $D_{\Lambda, r}$. By measuring this state we obtain a sample from $D_{\Lambda, r}$ and repeating the process we end up with $poly(n)$ samples. Each time we replace r with r' until r is small enough, the result is the closest lattice point from the given one, i.e. the shortest lattice vector.

This was just briefly going through the explanations, as the detailed description requires much deeper knowledge and is quite tedious. In fact this is not really of interest for our work, but the cryptosystem itself. More details can be found in [13].

Note that the theorem works only for $p > 2\sqrt{n}$. So, an important open question left is the problem of determining the hardness of the LWE problem in the special case $p = 2$.

The real proof presents an algorithm for finding short linearly independent lattice vectors. We do not want to write the whole proof, but just go over the explanations and the structure of this algorithm. For more details, see [13]. First we need a few lemmas.

Lemma 2. Given any lattice Λ , we can efficiently sample from $D_{\Lambda, r}$ for $r > 2^{4n} \lambda_n(\Lambda)$.

Lemma 3. Let Λ be an n -dimensional lattice, $\epsilon = \epsilon(n)$ be some negligible function, $\alpha = \alpha(n) \in (0, 1)$ be some real, and $p = p(n) \geq 2$ be some integer. Assume that we are given n^{c_1} samples from $D_{\Lambda, r}$ for some $r > p\eta_\epsilon(\Lambda)$ and a large enough c_1 . Also assume that we have an efficient algorithm W that solves LWE_{p, ψ_α} , then there exists an efficient quantum algorithm that produces any polynomial number of samples from $D_{\Lambda, r\sqrt{n}/(\alpha p)}$.

Remark 1. The first part of the proof describes a classical algorithm that using W and the samples from $D_{\Lambda, r}$ solves $CVP_{\Lambda^*, \alpha p/r}$. The second part presents a quantum algorithm that, given an oracle that solves $CVP_{\Lambda^*, \alpha p/r}$ outputs samples from $D_{\Lambda, r\sqrt{n}/(\alpha p)}$. Note that it holds:

$$\frac{\alpha p}{r} \leq \frac{\alpha}{\eta_\epsilon(\Lambda)} \leq \frac{1}{\eta_\epsilon(\Lambda)} \leq \frac{\lambda_1(\Lambda^*)}{2}$$

and hence the $CVP_{\Lambda^*, \alpha p/r}$ is well-defined.

Lemma 4. Let Λ be any n -dimensional lattice and let $r > 2\eta_\epsilon(\Lambda)$ where $\epsilon \leq \frac{1}{100}$. Then the probability that a set of n^2 vectors chosen from $D_{\Lambda,r}$ contains no n linearly independent vectors is exponentially small.

Proof. The algorithm for finding short linearly independent lattice vectors is the following:

- 1 Compute $\tilde{\lambda}_n$ such that $1 \leq \tilde{\lambda}_n/\lambda_n(\Lambda) \leq 2^n$
- 2 For each $k \in \{1, 2, \dots, \lceil \log p \rceil\}$ do
- 3 $r \leftarrow 2^{4n} \tilde{\lambda}_n 2^k$
- 4 Let S be n^c taken from $D_{\Lambda,r}$
- 5 While S contains n lin. indep. vectors from Λ of length $\leq r\sqrt{n}$ do
- 6 $S_k \leftarrow n$ lin. indep. vectors of length $\leq r\sqrt{n}$ from S
- 7 $r' \leftarrow r\sqrt{n}/(\alpha p)$
- 8 Let S' be n^c taken from $D_{\Lambda,r'}$
- 9 Set $r \leftarrow r'$ and $S \leftarrow S'$
- 10 Output the shortest set among $S_1, \dots, S_{\lceil \log p \rceil}$

The rough estimate of $\lambda_n(\Lambda)$ in the first step can be obtained by using LLL algorithm [8], where we can win a basis for Λ of length at most $2^n \lambda_n(\Lambda)$. In the outer loop of k , with each iteration we obtain a different candidate set S of n short linearly independent vectors until we output the shortest of these sets with respect to the length of a set, i.e. the length of the longest vector in it. The purpose of the loop is to achieve the $\tilde{O}(n/\alpha)$ approximation factor. Without it, the algorithm yields worse polynomial factor.

We can choose $c = \max\{c_1, 2\}$, where the constant c_1 comes from Lemma 3 and 2 from Lemma 4. The iterative step 8, can be explained with Lemma 3, as well. It allows us to create samples from $D_{\Lambda, r\sqrt{n}/(\alpha p)}$ by using samples from $D_{\Lambda,r}$. This distribution is narrower by a constant factor since $r\sqrt{n}/(\alpha p) \leq r/2$. Indeed, in order to obtain samples from $D_{\Lambda, r\sqrt{n}/(\alpha p)}$, the condition $r \geq 2\eta_\epsilon(\Lambda)$ must hold. Since the algorithm does not check this condition, step 8 might run for smaller values than r . In such cases, we do not assume anything about the set S' , it might contain arbitrary vectors that do not correspond to the distribution $D_{\Lambda, r\sqrt{n}/(\alpha p)}$.

Another important note is that using n^c samples from $D_{\Lambda,r}$, generates the same number of samples n^c from $D_{\Lambda, r\sqrt{n}/(\alpha p)}$. In fact, we could even generate more than n^c samples. The algorithm does not work if we generate only $n^c/2$ samples. This is because the number of iterations of step 8 is polynomial and if the number is halved in each iteration, we would have to start with an exponential number of samples.

Finally, Regev shows that the algorithm finds n linearly independent vectors of length at most $2n\eta_\epsilon(\Lambda)/\alpha$. In each of the $\lceil \log p \rceil = O(\log n)$ iterations of the outer loop, the algorithm begins with $r \leq 2^{cn} \tilde{\lambda}_n 2p \leq 2^{(c+2)n} \lambda_n(\Lambda)$. In each iteration of the inner loop, r gets smaller by a factor of at least 2. Having n linearly independent vectors of length at most $r\sqrt{n}$ implies that $r\sqrt{n} \geq \lambda_n(\Lambda)$. Hence, the number of iterations of the inner loop is at most $(c+2)n + \frac{1}{2} \log n = O(n)$. \square

3 Regev's Cryptosystem

A public cryptosystem serves an unlimited number of participants. Each of them publishes a key called public and keeps secret another one, called private. The public key is available for everybody, but the private key is known only for its owner. The usual scenario is the following: Alice and Bob want to exchange information through a secure channel. If Alice wants to send a message to Bob, she gets Bob's public key from a directory available for everybody. (We do not assume that Alice has a public or private key.) Then, using Bob's private key, she encodes the message and sends it to Bob through an open channel. Bob using his private key is able to decode the message, but without this private key the message cannot be decode.

3.1 Example

Let us consider a simple example, but keep in mind that we want to lay stress on explaining the cryptosystem. We do not care about the parameters in this part and the following has little to do with the actual parameters, it is just for illustration:

The private key is a vector $s = (\mathbf{0} \ \mathbf{1} \ \mathbf{2})^T$. It is chosen uniformly at random from \mathbb{Z}_p^n . The public key is given by the pair (A, b) , where A is a 6×3 matrix $A \in \mathbb{Z}_p^{m \times n}$:

$$A = \begin{pmatrix} 2 & 0 & 1 \\ 1 & 2 & 2 \\ 0 & 2 & 0 \\ 1 & 2 & 0 \\ 0 & 1 & 1 \\ 2 & 1 & 0 \end{pmatrix}$$

and $b = (\mathbf{2} \ \mathbf{0} \ \mathbf{2} \ \mathbf{2} \ \mathbf{0} \ \mathbf{1})^T \in \mathbb{Z}_p^m$ is a vector. Remember that $b = As + e \pmod p$. In this case $p = 3$, $m = 6$ and $n = 3$.

Suppose Alice wants to encrypt a message and send it to Bob. She chooses a random subset of rows from the public known matrix A . In this case, these are the first and the forth. The process continues depending on the bit she wants to encrypt. If it is 0, she calculates the sum of the first and the forth row, element by element. Then does the same with the corresponding b 's. (Do not forget that all the additions are made modulo p .) She sends the result to Bob.

$$a_1 + a_4 = 3 \cdot ? + 2 \cdot ? + 1 \cdot ? \approx 1$$

He computes $b - \langle a, s \rangle$, using his private key s . Precisely:

$$b - \langle a, s \rangle = 1 - \sum a_i b_i = 1 - (3 \cdot 0 + 2 \cdot 1 + 1 \cdot 2) = -3 \pmod 3 = 0$$

As the exact result is 0, which is obviously closer to 0 than any other number and $\lfloor \frac{p}{2} \rfloor \pmod p = 1 \pmod 3 = 1$ as well, Bob decides that the encrypted bit was 0.

If the encrypted message from Alice was:

$$a_1 + a_4 = 3 \cdot ? + 2 \cdot ? + 1 \cdot ? \approx 2$$

Bob repeats the same calculations, plugging in 2 for b . The result is 1. That will be the case of encryption on a 1-bit, because 1 is closer to $\lfloor \frac{p}{2} \rfloor \pmod p = 1 \pmod 3 = 1$ than 0.

3.2 Formal Definition

Let n be the security parameter of the cryptosystem. The other parameters are m and p both integers, and a probability distribution χ on \mathbb{Z}_p . Choose m such that $m = 5(n+1)(1+2\log n)$ and set $p \geq 2$ to be some prime number between n^2 and $2n^2$. The probability distribution χ is taken to be $\bar{\psi}_\alpha$ for $\alpha = \alpha(n) = o(1/\sqrt{n} \log n)$, i.e. we can choose $\alpha(n) = 1/(\sqrt{n} \log^2 n)$. In the following description, all additions are performed in \mathbb{Z}_p , i.e. modulo p .

Private key : The private key of Bob is a vector $s \in \mathbb{Z}_p^n$, chosen uniformly at random. More precisely, Bob picks a random vector s with uniform distribution from \mathbb{Z}_p^n .

Public key : The public key is the pair $(a_i, b_i)_{i=1}^m$ where $b_i = \sum e_i \langle a_i, s_i \rangle$ and the sequence of vectors m vectors $a_1, \dots, a_m \in \mathbb{Z}_p$ is chosen independently from the uniform distribution. The same holds for the elements $e_1, \dots, e_m \in \mathbb{Z}_p$, which were chosen again independently according to χ , for $i = 1, \dots, m$.

Encryption : Knowing the public key and assuming that Alice knows how to generate the necessarily distribution, she can start sending messages to Bob. If she wants to send a single 0 bit, then she chooses a random subset S of $[m]$. The actual encryption is $(\sum_{i \in S} a_i, \sum_{i \in S} b_i)$. Else if Alice wants to encrypt a bit 1, she sends $(\sum_{i \in S} a_i, \lfloor \frac{p}{2} \rfloor + \sum_{i \in S} b_i)$.

Decryption : Now suppose that Bob received the message, which Alice sent him. Bob computes $b - \langle a, s \rangle$, using his private key s , then checks whether the result is closer to 0 or to $\lfloor \frac{p}{2} \rfloor$ modulo p . In the first case he concludes, that the encrypted bit was 0, while in the second 1.

3.3 Correctness

Next we want to prove that the probability of decryption error is small and the choice of the parameters of the cryptosystem guarantees correctness. In order to do that we need some additional notation. For a distribution χ on \mathbb{Z}_p and an integer $k \geq 0$, we define χ^{*k} as the distribution obtained by summing together k independent samples from χ , where addition is performed in \mathbb{Z}_p , i.e. modulo p , and for $k \geq 0$, χ^{*0} is a distribution that is constantly 0.

The claim is based on the following lemma :

Lemma 5. (Correctness):

Let $\delta > 0$. Assume that for any $k \in \{0, 1, \dots, m\}$, χ^{*k} satisfies that

$$\Pr_{e \sim \chi^{*k}} [|e| < \lfloor \frac{p}{2} \rfloor / 2] > 1 - \delta$$

Then the probability of decryption error is at most δ .

Notice that for an element $e \in \mathbb{Z}_p$ we define $|e|$ as the integer e in the case of $e \in \{0, 1, \dots, \lfloor \frac{p}{2} \rfloor\}$ and as the integer $p - e$ otherwise. In other words, $|e|$ represents the distance of e from 0.

Proof. First we will have a look at the encryption of 0. It is given by the pair (a, b) where :

$$\begin{aligned}
a &= \sum_{i \in S} a_i \\
b &= \sum_{i \in S} b_i = \sum_{i \in S} \langle a_i, s \rangle + e_i = \langle a, s \rangle + \sum_{i \in S} e_i \\
b - \langle a, s \rangle &= \sum_{i \in S} e_i
\end{aligned}$$

Here the distribution of the elements e_i , which have been chosen independently, is exactly the one mentioned above, namely $\chi^{|S|}$ on \mathbb{Z}_p , where the number of samples was taken from a random subset S of $[m]$, see chapter 3.2, Encryption. Then according to our assumption, $|\sum_{i \in S} e_i|$ is less than $\lfloor \frac{p}{2} \rfloor / 2$ with probability at least $1 - \delta$. In this case we are closer to 0, the decryption was correct.

Now let us consider the encryption of 1. It is given by the pair (a,b), but this time we have:

$$\begin{aligned}
a &= \sum_{i \in S} a_i \\
b &= \sum_{i \in S} b_i + \lfloor \frac{p}{2} \rfloor = \sum_{i \in S} \langle a_i, s \rangle + e_i + \lfloor \frac{p}{2} \rfloor = \langle a, s \rangle + \sum_{i \in S} e_i + \lfloor \frac{p}{2} \rfloor \\
b - \langle a, s \rangle - \lfloor \frac{p}{2} \rfloor &= \sum_{i \in S} e_i
\end{aligned}$$

Then we can see that the following inequality holds,

$$|b - \langle a, s \rangle - \lfloor \frac{p}{2} \rfloor| < |b - \langle a, s \rangle| < \lfloor \frac{p}{2} \rfloor / 2$$

and the last inequality was true, because of the assumption. So according to the same assumption, $|\sum_{i \in S} e_i|$ is less than $\lfloor \frac{p}{2} \rfloor / 2$ with the same probability, which was at least $1 - \delta$. And in this case we are closer to $\lfloor \frac{p}{2} \rfloor$, resulting that the decryption is correct again. \square

For a probability distribution ϕ on \mathbb{R}/\mathbb{Z} we define ϕ^{*k} similarly to χ^{*k} . It is a distribution obtained by summing together k independent samples from ϕ . For $x \in \mathbb{R}/\mathbb{Z}$, we define $|x|$ to be x for $x \in [0, \frac{1}{2}]$ and $1 - x$ otherwise.

Claim 1. For our choice of parameters it holds that for any $k \in \{0, 1, \dots, m\}$,

$$\Pr_{e \sim \bar{\psi}_\alpha^{*k}} [|e| < \lfloor \frac{p}{2} \rfloor / 2] > 1 - \delta(n)$$

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

Note that ϕ can be any distribution on \mathbb{R}/\mathbb{Z} , but one can think of it as ψ_α . See the definition of the error distribution, chapter 1.4. Then a sample from $\bar{\psi}_\alpha^{*k}$ can be obtained by sampling x_1, \dots, x_k from ψ_α and outputting $\sum_{i=1}^k \lfloor px_i \rfloor \bmod p$.

3.4 Security

In a public-key cryptographic scheme, a key pair is selected so that the problem of deriving the private key from the corresponding public key is equivalent to solving a computational problem that is believed to be intractable. Number-theoretic problems whose intractability form the basis for the security of commonly used public-key schemes are:

1. The integer factorization problem, whose hardness is essential for the security of **RSA public-key encryption** and signature schemes.
2. The discrete logarithm problem, whose hardness is essential for the security of the **ElGamal public-key encryption** and signature schemes and their variants such as the Digital Signature Algorithm (**DSA**).
3. **The elliptic curve discrete logarithm problem**, whose hardness is essential for the security of all elliptic curve cryptographic schemes.

The Regev's public-key cryptosystem we explained above is based on the worst-case quantum hardness of SVP and SIVP. This means that breaking the cryptosystem implies an efficient quantum algorithm for approximating SVP. This security guarantee is incomparable to the one by Ajtai and Dwork lattice-based public key cryptosystem based on unique-SVP, which is a special case of SVP. On one hand, it is stronger as it is based on the general SVP and on the other hand, it is weaker as it only implies a quantum algorithm for a lattice problem. Since no quantum algorithm is known to outperform classical algorithms for lattice problems, it is not unreasonable to suppose that lattice problems are hard even quantumly. Still we emphasize that the cryptosystem itself is entirely classical. The new cryptosystem of Regev is also much more efficient than the previous one, as the public key is of size $\tilde{O}(n^2)$ compared to the one of Ajtai and Dwork $\tilde{O}(n^4)$ and encrypting a message increases its size by $\tilde{O}(n)$, while before the value was $\tilde{O}(n^2)$, respectively. Regev's cryptosystem is based on the following lemma:

Lemma 6. (Security):

For $m \geq 5(n+1)\log p$, if there exist a polynomial time algorithm W that distinguishes between encryptions of 0 and 1 then there exist a distinguisher Z that distinguishes between $A_{s,\chi}$ and U for a non-negligible fraction of all possible private keys s .

We already know from chapter 1.4 what type of distribution $A_{s,\chi}$ is. It was defined as the distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$ obtained by choosing a vector $a \in \mathbb{Z}_p^n$ uniformly at random, choosing $e \in \mathbb{Z}_p$ according to χ , and outputting $(\mathbf{a}, \langle \mathbf{a}, \mathbf{s} \rangle + e)$, where additions are performed in \mathbb{Z}_p , i.e. modulo p . We want to have a closer look at the distinguisher Z . According to the definition we need to consider the acceptance probabilities of W with samples from the two distributions, namely $A_{s,\chi}$ and U (the uniform distribution on $\mathbb{Z}_p^n \times \mathbb{Z}_p$). The proof starts with the hypothesis that $|\rho_0(W) - \rho_1(W)| \geq \frac{1}{n^c}$ for some $c > 0$. Here $\rho_0(W)$ denotes the acceptance probability of W on input $((\mathbf{a}_i, b_i)_{i=1}^m, (\mathbf{a}, b))$ where (\mathbf{a}, b) is an encryption of 0 with the public key $(\mathbf{a}_i, b_i)_{i=1}^m$ and the probability is taken over the randomness in the encryption algorithm. Similarly, $\rho_1(W)$ for encryptions of 1 and $\rho_u(W)$ is the acceptance probability of W on inputs $((\mathbf{a}_i, b_i)_{i=1}^m, (\mathbf{a}, b))$ with $(\mathbf{a}_i, b_i)_{i=1}^m$ chosen again according to the private and public keys distribution but (\mathbf{a}, b) is chosen uniformly from $\mathbb{Z}_p^n \times \mathbb{Z}_p$. Then construct a new distinguisher W' for which holds $|\rho_0(W') - \rho_u(W')| \geq \frac{1}{2n^c}$. Because of the hypothesis, in the case of $|\rho_0(W) - \rho_u(W)| \geq \frac{1}{2n^c}$, W' is the same as W , while for $|\rho_1(W) - \rho_u(W)| \geq \frac{1}{2n^c}$, W' calls W with $((\mathbf{a}_i, b_i)_{i=1}^m, (\mathbf{a}, \frac{p-1}{2} + b))$ on the same input $((\mathbf{a}_i, b_i)_{i=1}^m, (\mathbf{a}, b))$. In this way the distribution on encryption of 0

is mapped to the distribution on encryption of 1 and the uniform distribution to itself. Therefore, the claim is that W' is the required distribution. Then Regev shows a distinguisher Z that distinguishes between U and $A_{s,\chi}$ for any $s \in W'$ such that holds $|\rho_0(s) - \rho_u(s)| \geq \frac{1}{4n^c}$, stressing out that these are the same acceptance probabilities but for W' . Taking m samples $(\mathbf{a}_i, b_i)_{i=1}^m$ from R , which is some distribution that is either U or $A_{s,\chi}$, the distinguisher Z estimates both the acceptance probabilities $\rho_0((\mathbf{a}_i, b_i)_{i=1}^m)$ and $\rho_u((\mathbf{a}_i, b_i)_{i=1}^m)$ up to an additive error of $\frac{1}{64n^c}$. If their difference is more than $\frac{1}{16n^c}$, Z accepts, otherwise Z rejects. For further details, see [13].

3.5 Tight bounds

In this chapter we discuss the parameters of Regev's cryptosystem and give tighter bounds for some of them.

For the choice of parameters, presented in chapter 3.2, the public key size is $O(mn \log p) = \tilde{O}(n^2)$ and encryption increases the size of a message by a factor of $O(n \log p) = \tilde{O}(n)$. It is possible to reduce the size of the public key to $O(m \log p) = \tilde{O}(n)$ by the following idea of Ajtai [2]: assume all users of the cryptosystem share some fixed random choice of a_1, \dots, a_m which can be achieved by distributing these vectors as part of the encryption and decryption software. Then the public key need only consist of b_1, \dots, b_m . As an end effect of that modification the security of the cryptosystem is not affected.

Next we have a closer look at the setting of the parameters m and p that guarantees both security and correctness. Remember that $m = 5(n+1)(1+2 \log n)$. In fact m needs to be slightly bigger than $1.1n \log p$ and the factor of 5 is not really needed. One possible optimization is to replace it with 1.1. The parameter p , on the other hand which was set to be some prime $p \geq 2$ between n^2 and $2n^2$, needs to be large enough, so that not too many decryption errors occur. If one does not care about a few decryption errors, p can be certainly set to be smaller than n^2 . In fact the variance needs to be at least $2\sqrt{n}$ for the security proof to work, anything above \sqrt{n} is alright. We remind once again that only for $p > 2\sqrt{n}$ the main theorem works.

Finally, the security parameter n can be any integer bigger than 1, the fact is coming from the following lemma:

Lemma 7. Let $n \geq 1$ be some integer and $p \geq 2$ be a prime. Let ψ be some probability density function on \mathbb{R}/\mathbb{Z} and let $\bar{\psi}$ be its discretization on \mathbb{Z}_p . Then, assume there exists a distinguisher that distinguishes $A_{\mathbf{s}, \bar{\psi}}$ from U for a non-negligible fraction of all possible \mathbf{s} . Then there exists an algorithm that solves $LWE_{p,\psi}$.

In other words, Regev shows through a sequence of reductions, that all variants of the LWE problem are as hard as LWE. For more details have a look at chapter 4 in [13].

Conclusion

In the preceding chapters we have seen that lattice problems like SIVP or CVP are connected by a reduction problem on which Regev based a new lattice cryptosystem. In fact, we saw that instead of immediately finding very short vectors in a lattice, the iterative reduction of Regev suggests proceeding in steps, where in each step shorter lattice vectors are found. Thus, in each step the length of the vectors found decreases by a factor of 2, so that after a polynomial number of steps we end up with vectors that are within some polynomial factor of the shortest. Iterative reductions were so far never used in the construction of a lattice-based public-key cryptosystem. The main theorem is the first to apply such a reduction in this context. Moreover, it is the reduction that allows Regev to relate the LWE problem to lattices, and in particular to problems like SVP and SIVP as opposed to the unique-SVP.

Then we saw that, using the reduction from worst-case lattice problems such as SVP and SIVP to a certain learning problem, Regev presented a public-key cryptosystem whose hardness is based on the worst-case quantum hardness of SVP and SIVP, while previous lattice-based cryptosystems such as the one by Ajtai and Dwork were only based on unique-SVP, a special case of SVP. The advantages of this application were quite astonishing itself because the system seems to be very simple and practicable, and more with improved efficiency. Comparing it with previous one by Ajtai and Dwork, we saw that the public key size has decreased from $\tilde{O}(n^4)$ to $\tilde{O}(n^2)$ and the encryption increases the size of a message by a factor of $\tilde{O}(n)$ instead of $\tilde{O}(n^2)$. Moreover, we can reduce the public key size even to $\tilde{O}(n)$, under the assumption that all users share a random bit string of length $\tilde{O}(n^2)$. Still, this modification does not affect security of the cryptosystem.

Unfortunately, the main results of Regev have some disadvantages as well. First of all, the cryptosystem does not solve the open problem of basing a cryptosystem on the worst-case hardness of SVP and SIVP because of its quantum aspect. Thus, the main open question of Regev's work is whether his results can be made classical. The difficulty is coming from the fact that there seems to be no classical way to use an oracle that solves the closest vector problem within a small distance. Another important open question is to determine the hardness of the learning from parity with errors problem. His main theorem seems to work only for $p > 2\sqrt{n}$, while for smaller values of p is still undefined. And finally, we want to go back to the two of the main computational problems on lattices SVP and SIVP. It is presumed that there is no polynomial time algorithm that approximate them to within any polynomial factor. One might guess that the same conjecture holds in quantum world, i.e. there is no quantum polynomial time algorithm that approximates these problems to within any polynomial factor. Thus one can interpret the main theorem as saying that based on this guess, the LWE problem is hard. But still, the only evidence supporting this statement is that there are no quantum algorithms for lattice problems that are known to output classical algorithms. However we do not know if this is true. Another way of interpreting the main theorem of Regev would be: if one finds an efficient algorithm for LWE, then one also obtains a quantum algorithm of

approximating worst-case lattice problems.

These questions highlight some of the weaknesses of Regev's cryptosystem as well as the gap of knowledge regarding the complexity of lattice problems. Nonetheless, the results of Regev offer hope that the intractability of lattice reduction may provide an alternative to the assumptions that integer factorization and the discrete logarithm problem are intractable. On the other hand, the significance of his work in theory is a hope that his results will be made classical, which would make them stronger and would be of tremendous importance on its own.

References

- [1] M. Ajtai. Generating hard instances of lattice problems. *ECCCTR: Electronic Colloquium on Computational Complexity, technical reports*, 1996.
- [2] M. Ajtai. Representing hard lattices with $o(n \log n)$ bits. In *Proc. 37th Annual ACM Symp. on Theory of Computing (STOC)*, 2005.
- [3] M. Ajtai and C. Dwork. A public-key cryptosystem with worst-case/average-case equivalence. In *Proc. 29th Annual ACM Symp. on Theory of Computing (STOC)*, 1997.
- [4] L. Babai. *On Lovasz' lattice reduction and the nearest lattice point problem*. *Combinatorica*, 6(1):1-13, 1986.
- [5] A. Blum, A. Kalai, and H. Wassermann. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of ACM*, 50:506–519, 2003.
- [6] W. Ebeling. Lattices and codes. *Advanced Lectures in Mathematics. Friedr. Vieweg and Sohn, Braunschweig. A course partially based on lectures by F.Hirzebruch*, revisited edition, 2002.
- [7] R. Kumar and D. Sivakumar. On polynomial approximation to the shortest lattice vector length. *Proc.12th annual ACM-SIAM Symp. on Discrete Algorithms*, pages pages 126–127, 2002.
- [8] A. K. Lenstra, Jr. Lenstra, H. W., and L. Lovasz. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.
- [9] D. Micciancio. Improving lattice based cryptosystems using the hermite normal form. In *J.Silverman, editor, Cryptography and Lattices Conference-CaLC 2001*, volume 2146 of *Lecture Notes in Computer Science*:pages 126–145, Providence, Rhode Island, Springer Verlag, 2001.
- [10] D. Micciancio and S. Goldwasser. Complexity of lattice problems: A cryptographic perspective:. *volume 671 of The Kluwer International Series in Engineering and Computer Science. Kluwer Academic Publishers, Boston, Massachusetts*, 2002.
- [11] D. Micciancio and O. Regev. Worst-case to average-case reduction based on gaussian measures. In *Proc.45th Annual IEEE Symp. on Foundations of Computer Science(FOCS)*, 2004.
- [12] Daniele Micciancio. Lattices in cryptography and crytanalysis, lecture notes. *CSE 291*, 1999.
- [13] O. Regev. On lattices, learning with errors, random linear codes, and cryptography. In: *Proc.37th ACM Symp. on theory of Computing(STOC)*, 2005.
- [14] O. Regev and D. Aharonov. *Lattice problems in NP intersect coNP*. In *Proc, 45th Annual IEEE Symp. on Foundations of Computer Science (FOCS)*, 2004.

- [15] Oded Regev. Lattices in computer science, lecture notes. *Tel Aviv University*, 2004.