ON THE NTRU PUBLIC KEY CRYPTOSYSTEM

CANAN ÇİMEN

September 2008

ON THE NTRU PUBLIC KEY CRYPTOSYSTEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF APPLIED MATHEMATICS
OF
THE MIDDLE EAST TECHNICAL UNIVERSITY

BY

CANAN ÇİMEN

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF CRYPTOGRAPHY

September 2008

Approval of the Graduate School of Applied Mathematics

Prof. Dr. Ersan AKYILDIZ
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.

Prof. Dr. Ferruh ÖZBUDAK
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

Assoc. Prof. Dr. Emrah ÇAKÇAK
Supervisor

Examining Committee Members

Assoc. Prof. Dr. Emrah ÇAKÇAK _____

Assoc. Prof. Dr. Ali DOĞANAKSOY _____

Prof. Dr. Ali Bülent EKİN _____

Assist. Prof. Dr. Hakan ÖKTEM _____

Inst. Dr. Muhiddin UĞUZ _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last name : Canan ÇİMEN
Signature :

Abstract

ON THE NTRU PUBLIC KEY CRYPTOSYSTEM

ÇİMEN CANAN
M.Sc., Department of Cryptography
Supervisor: Assoc. Prof. Dr. Emrah Çakçak

September 2008, 51 pages

NTRU is a public key cryptosystem, which was first introduced in 1996. It is a ring-based cryptosystem and its security relies on the complexity of a well-known lattice problem, i.e. shortest vector problem (SVP). There is no efficient algorithm known to solve SVP exactly in arbitrary high dimensional lattices. However, approximate solutions to SVP can be found by lattice reduction algorithms. LLL is the first polynomial time algorithm that finds reasonable short vectors of a lattice.

The best known attacks on the NTRU cryptosystem are lattice attacks. In these attacks, the lattice constructed by the public key of the system is used to find the private key. The target vector, which includes private key of the system is one of the short vectors of the NTRU lattice.

In this thesis, we study NTRU cryptosystem and lattice attacks on NTRU. Also, we applied an attack to a small dimensional NTRU lattice.

Keywords: Lattice Based Cryptography, Lattice Problems, Lattice Reduction Algorithms, NTRU, Lattice Attacks.

ÖZ

AÇIK ANAHTAR KRİPTOSİSTEMİ NTRU ÜZERİNE

ÇİMEN Canan
Yüksek Lisans, Kriptografi Bölümü
Tez Yöneticisi: Doç. Dr. Emrah Çakçak

Eylül 2008, 51 sayfa

Açık anahtar kriptosistemi NTRU ilk olarak 1996 yılında tanıtıldı. Halka tabanlı bir kriptosistem olan NTRU'nun güvenliği, iyi bilinen bir kafes problemi olan en kısa vektör problemine dayanır. En kısa vektör problemini yüksek boyutlu kafesler için tam olarak çözen bir algoritma yoktur. Ancak, yaklaşık sonuçlar kafes indirgeme algoritmalarıyla bulunabilir. LLL algoritması, makul kısalıkta kafes vektörlerini bulan, polinom zamanlı ilk algoritmadır.

NTRU kriptosistemi üzerine düzenlenen, bilinen en iyi saldırılar kafes saldırılarıdır. Bu saldırılarda, gizli anahtarı bulmak için sistemin açık anahtarıyla oluşturulan kafes kullanılır. Gizli anahtarı içeren hedef vektör, kafesin kısa vektörlerinden biridir. NTRU kafesine, hedef vektörü bulmak için, kafes indirgeme algoritması uygulanır.

Bu tezde, NTRU kriptosistemi ve NTRU'ya uygulanan kafes saldırıları üzerinde çalıştık. Ayrıca, küçük boyutlu bir NTRU kafesine saldırı uyguladık.

Anahtar Kelimeler: Kafes Tabanlı Kriptografi, Kafes problemleri, Kafes İndirgeme Algoritmaları, NTRU, Kafes saldırıları

To Onur and my family

Acknowledgments

I would like to express my sincere appreciation to my supervisor, Assoc. Prof. Dr. Emrah Çakçak for his interest and guidance throughout the work. I also thank to him for his great advices in the writing period of the thesis.

To my dear Onur, I offer special thanks for his love, patience, and support. I am also greatful to my lovely parents for their unwavering support and encouragement during my whole life.

I am also so thankful to my sister Yasemin, my brother Serdar Yorulmaz and our little Defne for their support and love, even though they are far away.

I wish to thank to my friend Turgut Hanoymak for introducing this subject to me, for his cooperation and support at all times I need.

I also would like to thank to all my friends for their help and moral support during the preparation of the thesis.

Table of Contents

Chapter 1

Introduction

Until 1970's, cryptography was concerned only with message confidentiality. The security of the cryptosystems relied on the protection of the private key which was used in both encryption and decryption. Today, they are called symmetric key cryptosystems. In these systems, sender and receiver shares the same key in order to encrypt and decrypt messages. The encryption and decryption schemes are fast in practice, but key management is a big deal in the implementation of large user networks where communication flow is encrypted.

In 1976, Diffie and Hellman introduced the notion of public key cryptography in which two different keys are used [32]. This was a breaktrough in cryptography, because the key exchange problem would be avoided by using a pair of keys, a public key and a private key. The public key is accessible to everyone, but the private key is only known by the sender. This remarkable idea provided cryptography to be used by large networks of users in an unsecure channel. After this idea, several public key cryptosystems have been presented. In 1978, RSA was introduced by Rivest, Shamir and Adleman which is still the most widely used public key cryptosystem [30]. Its security relies on the difficulty of factoring big integers which is one of the hardest computational problem of mathematics. The other well known public key cryptosystem is Elliptic Curve Cryptography(ECC) which was introduced independently by Neal Koblitz and Victor Miller in 1985.

The security of the ECC comes from the elliptic curve logarithm, which is the discrete logarithm problem in a group defined by points on an elliptic curve over a finite field. The idea of public key cryptography depends on the one way functions that are easy to compute but hard to invert. More explicitly, the encryption is performed easily by using the public key but the knowledge of the public key does not help to decrypt the message and the eavesdropper is encountered to solve a hard computational problem in mathematics. There are many other such problems that might be used in public key cryptography. In this thesis, we investigate lattice problems which are potentially appropriate to be the base of a public key cryptosystem.

In the past twenty years, lattices have played an extremely important role in cryptology. Prior to 1996, lattices and in particular the lattice basis reduction algorithms were used in cryptanalysis to prove the insecurity of some cryptosystems. By Ajtai's paper which was published in 1996 [22], some progress was made on the complexity of lattice problems. In [22], Ajtai showed a connection between the worst case complexity and the average case complexity of some well known lattice problems. Then in 1997, he proved the NP-hardness of the most famous lattice problem, the shortest vector problem [23]. These results attracted cryptographers and after Ajtai's ideas, several cryptographic schemes based on the lattice problems were proposed.

We are interested in a relatively new public key cryptosystem, NTRU whose security relies on the complexity of the shortest vector problem. Actually, NTRU is a ring-based cryptosystem that was first introduced in a rump session at Crypto' 96 [13]. Its encryption and decryption schemes are constructed on the algebraic structures of certain polynomial rings. Recently, it has received quite atten-

tion since it appears to be more efficient than the current and more widely used public-key cryptosystems. The advantages of NTRU than other public key cryptosystems are that the generation of public and private key pairs is easy and quick; encryption and decryption are very fast and key sizes are relatively small. As already noted, the underlying hard problem that the security of NTRU relies on is the shortest vector problem. Finding the shortest vector of a lattice is a problem that is proved to be NP-hard and lattice reduction algorithms aim to find very short vectors in a lattice. The best attack known against NTRU cryptosytem is lattice attack which basically uses lattice reduction algorithms. The most famous reduction algorithm LLL is a polynomial time algorithm that was presented by Lenstra, Lenstra and Lovasz in 1982 [3]. Therefore LLL and its versions constitute a significant part in the security of NTRU.

In this study, we shall analyze the lattice attacks on NTRU and give an example in small dimensional lattices. We will use LLL reduction and BKZ reduction which is an improvement of LLL and we will compare these two reduction algorithms.

The thesis is organized as follows:

In chapter 2, we give a brief introduction to lattice theory. We state the basic definitions and notations related to lattices. We present the lattice reduction algorithms LLL and BKZ.

Chapter 3 is devoted to NTRU and we describe the classical NTRU cryptosystem. There have been several improvements of NTRU after the succesful attacks, but the main principles remain the same. Therefore we discuss the first version presented in the original paper. In the last section, we give a comparison of NTRU with the other public key cryptosystems.

In Chapter 4, we introduce two important attacks that have been applied on NTRU, i.e. meet in the middle attack and lattice attacks. We describe how lattices can be used to attack NTRU and different versions of lattice attacks.

Finally, in chapter 5 we analyze the complexities of the attacks given in the previous chapter. We give an example of lattice attack in small dimensional lattice by using two different lattice reduction algorithms and compare the results.

Chapter 2

Lattice Theory

2.1 Introduction

The basic idea underlying the public key cryptography relies on the complexity of some computational problems in number theory. Factorizing big integers and computing discrete logarithms in certain groups are problems that are most commonly used in public key cryptography. In 1996, certain lattice problems are proved as NP-hard by Ajtai [23]. This paper provided lattices to attract considerable attention to be used in public key cryptosystems. Afterwards, new public key cryptosystems, i.e. lattice based cryptosystems started to be introduced.

In order to study the lattice based cryptosystems and NTRU, we shall introduce basic properties of lattices, the important lattice problems and lattice basis reduction algorithms in this chapter. For a detailed study about lattice theory, we recommend the lecture notes of Cyntia Dwork [6].

2.2 Basic definitions and properties

In this section, we introduce basic definitions and results of lattice theory. We first give the definition of the inner product of two vectors.

Let $v = (v_1, v_2, ..., v_m)$ and $w = (w_1, w_2, ..., w_m)$ be two vectors in $\mathbb{R}^m$. The expression

$$(v, w) = \sum_{i=1}^{m} v_i w_i$$

is called the inner product of v and w. These are called *orthogonal* if $(v, w) = 0$. For $v = (v_1, v_2, ..., v_m)$, we denote by

$$||v|| = \sqrt{(v, v)} = \sqrt{\sum_{i=1}^{m} v_i^2}$$

the *length* or *Euclidean norm* of v.

Definition 2.2.1. Let B be a set of linearly independent vectors $v_1, v_2, ..., v_n \in \mathbb{R}^m$. The *lattice* generated by the set of vectors B is the set

$$L(B) = \left\{ \sum_{i=1}^{n} a_i v_i \mid a_i \in \mathbb{Z} \right\} \tag{2.2.1}$$

We say that the set B is a basis of the lattice L and the dimension of the lattice is n, i.e. $dim(L(B)) = n$. If $m = n$ the lattice is called *full-dimensional*.

Alternatively, we can define B as a matrix in $\mathbb{R}^{n \times m}$ with $v_1, v_2, ..., v_n$ as rows. Then B is called the basis matrix of the lattice $L(B)$ in 2.2.1 and $\dim(L(B)) = rank(B)$.

A lattice is just like a vector space except that it is generated by all linear combinations of its basis vectors with integer coefficients, rather than real coefficients. Two different bases for a lattice L are related to each other in almost the same way that two different bases for a vector space V are related to each other. That is, if $B = \{v_1, v_2, ..., v_n\}$ is a basis for a lattice L then $B' = \{w_1, w_2, ..., w_n\}$

is another basis for L if and only if there exist $a_{i,j} \in \mathbb{Z}$ such that

$$\begin{aligned} a_{1,1}v_1 + a_{1,2}v_2 + ... + a_{1,n}v_n &= w_1 \\ a_{2,1}v_1 + a_{2,2}v_2 + ... + a_{2,n}v_n &= w_2 \\ &\vdots \\ a_{n,1}v_1 + a_{n,2}v_2 + ... + a_{n,n}v_n &= w_n \end{aligned}$$

and the determinant of the basis transformation matrix

$$\begin{pmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & a_{2,2} & \dots & a_{2,n} \\ & & \vdots & \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{pmatrix}$$

is equal to 1 or -1. This matrix is a unimodular matrix in $\mathbb{R}^{n\times m}$, i.e. an integer matrix with determinant ∓ 1.

Definition 2.2.2. Let L be a lattice of dimension n with basis $B = \{v_1, v_2, ..., v_n\}$. A *fundamental domain* or *fundamental parallelepiped* associated with B is the set of points:

$$P(B) = \left\{ \sum_{i=1}^{n} t_i v_i \mid 0 \leq t_i < 1 \right\}$$

We observe that $v + P(B)$ where $v \in L(B)$ forms a partition of the vector space spanned by B. In other words, for any $x \in \text{span}(B)$ there exists a unique lattice point $v \in L(B)$ such that $x \in v + P(B)$.

Definition 2.2.3. Let $B \in R^{m\times n}$ be the basis matrix of a lattice L. The volume of the fundamental domain associated to the basis B is called the *determinant* of the lattice L and denoted by $\det(L)$.

For a full dimensional lattice, the determinant of the lattice is equal to the absolute value of the determinant of the basis matrix, i.e. $\det(L(B)) = |det(B)|$.

Remark 2.2.4. *The determinant of a lattice is an invariant of the lattice, i.e. independent of the choice of basis. In the case of a full dimensional lattice, this can be easily proven by using the unimodularity of the basis transformation matrix.*

2.3 Lattice Problems

In this section, we shall introduce the main computational problems on lattices. The most famous problem is the shortest vector problem which has been studied by mathematicians for centuries since the time of Gauss and it has remained one of the most important open problems in this area. Another long standing open problem is closest vector problem which is a version of the shortest vector problem.

Shortest Vector Problem(SVP): Given a lattice L, find the shortest nonzero vector v in L.

Closest Vector Problem (CVP): Given a vector $w \in \mathbb{R}^n$, find a vector $v \in L$ that is closest to w.

If the dimension of the lattice L is large, both SVP and CVP are NP- hard problems. Ajtai [23] recently proved that SVP is NP-hard under randomized reductions. CVP seems to be a more difficult problem. In [28] it is proven that CVP cannot be easier than SVP. We refer to Cai [12] for a detailed survey of complexity results of these problems. There is also one more important problem in lattice theory which is called minimum basis problem.

Minimum Basis Problem: Given a lattice L, find a basis $B = (v_1, v_2, ..., v_n)$

such that the product of the lengths of its vectors, which we call the weigth of the basis B,

$$\prod_{i=1}^{n} ||v_i||$$

is minimum.

This problem is also an NP-hard problem that was proved by Lovasz [21]. There is a general lower bound for the weight of any basis of a lattice and in a given lattice, the weight of all bases are at most a constant multiple of a general upper bound. These bounds are given as follows:

Hadamard's Inequality: Let $v_1, v_2, ..., v_n$ be any basis for L. Then

$$\det(L) \leq \prod_{i=1}^{n} ||v_i|| \tag{2.3.2}$$

In general, Hadamard's inequality amounts to say that the volume of a parallelepiped can never be greater than the product of the lengths of its sides. In this case, it means that the determinant of the lattice constitutes a lower bound for the product of the lengths of any basis vectors. We have equality in 2.3.2 if and only if the basis vectors are pairwise orthogonal. On the other hand, the following result which was introduced by Hermite in 1850, gives an upper bound for the weight of a basis of a lattice.

Theorem 2.3.1. *Let L be the lattice with basis $v_1, v_2, ..., v_n$. There is a number γ_n, depending only on n such that*

$$\prod_{i=1}^{n} ||v_i|| \leq \gamma_n^{n/2} \det(L)$$

The value γ_n is called Hermite's constant. The exact value of γ_n is known

only for $n \leq 8$ and for large n, it is proved that

$$\sqrt{\frac{n}{2\pi e}} \leq \gamma_n \leq \sqrt{\frac{n}{\pi e}}$$

Hadamard's inequality and Hermite's constant are given as lower and upper bounds for the weight of a basis for a lattice which are relevant to the minimum basis problem. Now, we will introduce a theorem that gives an upper bound for the shortest vector of a lattice.

Theorem 2.3.2. (*Minkowski*) *Let L be a lattice of dimension n. Let $\sigma(L)$ denote the length of the expected shortest nonzero vector in the lattice L. Then*

$$\sigma(L) \leq \gamma_n \det(L)^{1/n} \tag{2.3.3}$$

where γ_n is the Hermite constant.

The exact bounds for the size of the shortest vector of a lattice are unknown for large n in 2.3.3. There is also a result which gives an estimate for the length of a shortest vector in a random lattice. The *Gaussian Heuristic* says that the length of the shortest nonzero vector in a random lattice with dimension n is approximately

$$\sigma(L) \approx \sqrt{\frac{n}{2\pi e}} det(L)^{1/n} \tag{2.3.4}$$

Up to now, we gave some important definitions and theorems about lattices and lattice problems. Now we will introduce the lattice reduction methods which are useful to solve certain instances of lattice problems.

2.4 Lattice Basis Reduction

A lattice has infinitely many bases, but some are more significant than others, e.g. the bases that contain reasonably short and almost orthogonal vectors. A basis where the basis vectors are short is considered an interesting representation of a lattice. The goal of lattice basis reduction algorithms is to find this representation of a given lattice and the shortest vector in that basis.

When the lattice dimension is sufficiently low, the shortest vector of a lattice can be found by using exhaustive search by enumeration techniques [9]. However, in high dimensional lattices, e.g. beyond dimension 100, the running time of exhaustive search is exponential. Until 1982, no algorithm were known to find reasonable short vectors in high dimensional lattices. In 1982, by the paper of Lenstra, Lenstra and Lovasz the best lattice reduction method *LLL Algorithm* was introduced [3]. LLL is still the only algorithm that finds reasonable short lattice vectors in polynomial time.

One approach to finding short vectors in lattices is to obtain a basis that is close to orthogonal. Besides, if a lattice has an orthogonal basis, then its shortest vector is one of the basis vectors.

Proposition 2.4.1. *Let L be a lattice with an orthogonal basis $B = \{v_1, v_2, ..., v_n\}$ then the shortest vector of the lattice is one of the basis vectors.*

Proof. Let $v \in L$ be any given vector and $B = \{v_1, v_2, ..., v_n\}$ be an orthogonal basis of L. Then $v = c_1v_1 + ... + c_nv_n$ where $c_i \in \mathbb{Z}$. Therefore

$$||v|| = ||c_1v_1 + ... + c_nv_n||$$

Since the basis B is orthogonal, we deduce:

$$\begin{aligned} ||v||^2 &= c_1^2||v_1||^2 + ... + c_n^2||v_n||^2 \\ ||v||^2 &\geq ||v_i||^2 \ \ for \ \ any \ \ i \in \{1, 2, .., n\} \\ ||v|| &\geq ||v_i|| \geq min_{1 \leq i \leq n}||v_i|| \end{aligned}$$

□

We have shown that a shortest lattice vector is always contained by orthogonal bases. Any basis of a vector space can be transformed into an orthogonal basis for the same vector space using the Gram-Schmidt orthogonalization process.

Definition 2.4.2. Let $B = \{v_1, v_2, ..., v_n\}$ be any set of vectors in R^n. We can calculate the *Gram-Schmidt* orthogonalization:

$$\begin{aligned} v_1^* &= v_1 \\ v_i^* &= v_i - \sum_{j=1}^{i-1} \mu_{i,j} v_j^*, \quad i = 2, ..., n \end{aligned}$$

$$\mu_{i,j} = \frac{(v_i, v_j^*)}{(v_j^*, v_j^*)}$$

Let's denote the set of the vectors $\{v_1^*, v_2^*, .., v_n^*\}$ by B^*. Here, each v_i^* is explained as the projection of v_i onto the orthogonal complement of the subspace spanned by $\{v_1, .., v_{i-1}\}$. Now, let $P(B)$ be the parallellepiped associated with B. When $n = 2$, $P(B)$ is a parallelogram and the area of $P(B)$ is $||v_1^*||||v_2^*||$. It

follows by induction on n that, for any $n \geq 1$, the volume of $P(B)$ is:

$$V(P(B)) = \prod_{i=1}^{n} ||v_i^*||$$

which is also the determinant value of the lattice $L(B)$.

Although B^* is a basis of the vector space spanned by B, in general it will not be a basis for the lattice $L(B)$, because some $\mu_{i,j}$ may be non-integral. Now, we give the definition of the LLL reduced basis.

Definition 2.4.3. A basis $B = (v_1, v_2, .., v_n)$ of a lattice L is said to be ***LLL Reduced*** if:

1. it is **size reduced**, i.e.

$$|\mu_{i,j}| \leq \frac{1}{2} \quad 1 \leq i < j \leq n \tag{2.4.5}$$

2. for all $1 < i \leq n$, the following holds

$$||v_{i+1}^* + \mu_{i+1,i} v_i^*||^2 \geq \frac{3}{4}||v_i^*||^2 \tag{2.4.6}$$

The constant $\frac{3}{4}$ may be replaced with any δ satisfying $\frac{1}{4} < \delta < 1$, in which case the basis is said to be LLL reduced with respect to the parameter δ. If δ is close to 1, the resulting basis may have shorter vectors but it will take more swaps and time to find them.

To better understand the properties of an LLL reduced basis, we give the following theorem which has a detailed proof in [11].

Theorem 2.4.4. *Let $B = (v_1, v_2, .., v_n)$ be an LLL-reduced basis of a lattice L*

with a δ-value 3/4. This basis will satisfy

1. $$detL \leq \prod_{i=1}^{n} ||v_i|| \leq 2^{n(n-1)/4}|detL|$$

2. $$||v_i|| \leq 2^{(j-1)/2}||v_j^*|| \quad for \;\; all \;\; 1 \leq i \leq j \leq n$$

3. $$||v_1|| \leq 2^{(n-1)/4}|detL|^{1/n}$$

In property 1, the first inequality is the Hadamard's inequality that does not require the basis to be reduced, but in the second inequality the Hermite constant replaces with a valid constant for LLL reduced basis. By property 3, we can deduce that finding an LLL reduced basis of a lattice provides us an approximate solution of the shortest vector problem.

2.5 LLL Algorithm

The history of lattice reduction goes back to the 19th century. There were many reduction algorithms introduced by different mathematicians like Lagrange[20], Gauss[7], Hermite, Korkine and Zolotarev [4]. However, the breaktrough on this subject was LLL reduction algorithm. LLL algorithm is a polynomial time algorithm which takes a basis of a lattice as input and returns an LLL reduced basis [3]. As already noted, the first vector of the basis returned by LLL is within a factor of the shortest vector. In practice, LLL often performs much better than the theoretical bound.

Definition 2.5.1. LLL Algorithm

Input A lattice basis $B = (v_1, v_2, .., v_n)$, $\delta \in (\frac{1}{4}, 1)$

Output An LLL-reduced basis of $L(B)$.

$k = 2$

While $k \leq n$ **do**

size reduce the vector v_k and recompute μ_{kj} for $j = 1, .., k-1$

If $||v_k^* + \mu_{k,k-1} v_{k-1}^*||^2 < \delta ||v_{k-1}^*||^2$

Then swap (v_k, v_{k-1})

$k = max(k-1, 2)$

Else $k = k + 1$

Return $v_1, ..., v_n$

The first part of the algorithm is performed to get a size reduced basis. If $\mu_{k,j}$ where $1 \leq j < k$ does not satisfy the inequality 2.4.5 then the size of v_k is reduced as follows:

$$\hat{v}_k = v_k - \sum_{j<k} \lfloor \mu_{k,j} \rceil \ \hat{v}_j$$

where $\lfloor \mu_{k,j} \rceil$ denotes the integer closest to $\mu_{k,j}$. Second part of the algorithm tests if the inequality 2.4.6 is satisfied. If it does not hold then the basis vectors v_{k-1} and v_k are swapped. This changes the $\mu_{k-1,j}$ and $\mu_{k,j}$ values, so first part is performed again. These iterations continues untill $k = n + 1$. It is obvious that if the algorithm terminates the output basis will be LLL reduced. We now show that indeed the algorithm terminates.

There is a bound for the number of iterations performed by the algorithm. The termination of the LLL algorithm will be proven by using a positive integer associated to the determinant of the lattice and by showing that this integer

decreases at least by a factor at each time we swap two vectors.

Definition 2.5.2. Let L be a lattice with a basis $(v_1, ..., v_n)$. Then

$$D = \prod_{i=1}^{n} det(L(v_1, .., v_i))^2$$

where $det(L(v_1, .., v_i)) = \prod_{k=1}^{i} ||v_k^*||$.

We want to show that D decreases at least by a factor δ at each iteration. During the size reducing step D does not change, because new basis $B' = (v_1', ..., v_n')$ obtained by size reducing and B are bases of the same lattice. Also they have the same orthogonalized basis B^* and lastly their determinants and D values are equal. So we should show that swap step affects D, actually it decreases D at least by δ.

First we look at the change in a single swap between v_{k-1} and v_k. Let D be the integer before the swap, D' be the integer after the swap.Then

$$\begin{aligned}
\frac{D}{D'} &= \frac{\prod_{i=1}^{k-1} det(L(v_1, .., v_i))^2}{(\prod_{i=1}^{k-2} det(L(v_1, .., v_i))^2) det(L(v_1, .., v_{k-2}, v_k))^2} \\
&= \frac{det(L(v_1, .., v_{k-2}, v_{k-1}))^2}{det(L(v_1, .., v_{k-2}, v_k))^2} \\
&= \frac{\prod_{j=1}^{k-1} ||v_j^*||^2}{\prod_{j=1}^{k-2} ||v_j^*||^2 ||\mu_{k,k-1} v_{k-1}^* + v_k^*||^2} \\
&= \frac{||v_{k-1}^*||^2}{||\mu_{k,k-1} v_{k-1}^* + v_k^*||^2} \geq \delta^{-1}
\end{aligned}$$

The last result comes from the second condition of the LLL reduced basis with

respect to the parameter δ, (see equation 2.4.6). Notice that the inequality does not satisfy, since we are performing a swap between v_{k-1} and v_k. Then the proof follows as

$$D' \leq \delta D$$

As shown here, in each iteration D decreases by a factor δ. After l swaps, $1 \leq D^{(l)} \leq \delta^l D$. So $l \leq \log_{1/\delta} D$

This proves that there is an upper bound for the number of iterations, so theoretically we have shown that LLL algorithm terminates. In order to show that the algorithm is polynomial time, we have to prove that each iteration takes polynomial time. For the running time analysis of the algorithm, see the lecture notes of Micciancio [25].

We have mentioned that for high dimensional lattices, lattice reduction algorithms cannot find the exact shortest vector of the lattice which is denoted by $\sigma(L)$. Actually, they find a lattice vector whose length is smaller or equal to an α factor of the shortest vector, where α is called approximation factor and $\alpha \geq 1$. Therefore such algorithms find reduced basis $B = \{v_1, v_2, .., v_n\}$ with the approximate shortest vector v_1 such that $||v_1|| \leq \alpha \cdot \sigma(L)$. LLL [3] provably finds a vector in an n dimensional lattice at most $(4/3)^{(n-1)/2}$ times the length of the shortest vector.

Several improvements of LLL have been proposed over the years. Schnorr [8] has extended the LLL reduction to a hierarchy of polynomial time reduction algorithms that finds a nonzero vector of a lattice with a smaller approximation factor. Later, Schnorr and Euchner presented practical algorithms [9]. One of them is called LLL with deep insertions. It is a variant of LLL with poten-

tially superexponential complexity that improves again the approximation factor. Another important algorithm is BKZ reduction algorithm which is a blockwise generalization of LLL [9]. The basis that BKZ algorithm gives as an output is block Korkin-Zolotarev reduced basis. Before introducing the algorithm, we give some useful definitions.

With a basis $v_1, v_2, .., v_n$ of the lattice L we associate the orthogonal projections:

$$\pi_i : span(v_1, ..., v_n) \rightarrow span(v_1, ..., v_{i-1})^{\perp} \quad i = 1, ..., n$$

where $span(v_1, ..., v_{i-1})^{\perp}$ is the orthogonal complement of $span(v_1, ..., v_{i-1})$. We let L_i denote the lattice

$$L_i = \pi_i(L)$$

which has dimension $n - i + 1$ and basis $\pi_i(v_i), \pi_i(v_{i+1}), ..., \pi_i(v_n)$, where $\pi_i(v_j)$ is defined as

$$\pi_i(v_j) = \sum_{k=i}^{n} \mu_{j,k} v_k^*$$

so that $\pi_i(v_j)$ is the part of v_j perpendicular to $v_1, v_2, ..., v_{i-1}$.

Hence, we can also write $L_i = L(\pi_i(v_i), \pi_i(v_{i+1}), ..., \pi_i(v_n))$. Then we define the lattice $L_i(k) = L(\pi_i(v_i), \pi_i(v_{i+1}), ..., \pi_i(v_{i+k-1}))$.

Definition 2.5.3. A basis $B = \{v_1, v_2, .., v_n\}$ is *Korkin-Zolotarev reduced (KZ-reduced)*, if:

1. it is size reduced (see 2.4.5)

2. v_i^* is the shortest nonzero vector of the lattice L_i for $1 \leq i \leq n$.

This definition is given by Korkin and Zolotarev in 1873. Schnorr [8] introduced the following type of the KZ-reduced bases.

Definition 2.5.4. Let k be an integer, $2 \leq k \leq n$. A basis B is block Korkin-Zolotarev reduced with block size k or in other words *k-BKZ reduced* if:

1. it is size reduced (see 2.4.5),

2. $L_i(k)$ is KZ-reduced, for $1 \leq k \leq n - k + 1$.

Definition 2.5.5. Block Reduction Algorithm(BKZ-LLL)

Input A lattice basis $B = (v_1, v_2, .., v_n)$ and an integer k, $2 \leq k \leq n$

Output A k-BKZ reduced basis of $L(B)$.

size reduce B

if there exists an i such that $L_i(k)$ is not KZ-reduced **then**

KZ-reduce $L_i(k)$, applying any vector operations made in $L_i(k)$

to $v_i, .., v_{i+k-1}$

Return B

As we have seen, in LLL algorithm the basis vectors are considered pairwise and in swapping step each v_i is required to be shorter than each vector in the two-dimensional projected lattice generated by v_i and v_{i+1}. But BKZ reduction replaces the swapping step with a more complicated procedure. Therefore BKZ runs in time exponential in k. However it works well in practice and finds shorter vectors than LLL algoritm.

We have used these two algorithms experimentally in our examples about lattice attacks to NTRU. The results will be given in the last chapter. Now, we introduce the NTRU cryptosytem which is the main subject of the thesis.

Chapter 3

NTRU

3.1 Lattice Based Cryptography

As already mentioned, Ajtai presented some ideas about the complexity of the lattice problems and proved the NP-hardness of the shortest vector problem. After Ajtai's discoveries, several cryptographic schemes were proposed.

Ajtai and Dwork proposed a cryptosystem related to Ajtai's work [24]. It is not related to lattices explicitly, but in the security proof of the system there are some connections to lattice problems. Its security is based on a variant of SVP. The system is secure unless the worst case of a particular lattice problem can be solved in polynomial time. Nguyen and Stern [29] presented an attack by approximating CVP sufficiently well and showed the insecurity of the system.

Another system based on the complexity of lattice problems is presented by Goldreich, Goldwasser and Halevi [27]. The security of GGH rests on the difficulty of solving CVP using highly nonorthogonal bases. GGH is currently secure, when the key size is very large. So it is proven impractical.

In this thesis, we work on the NTRU cryptosystem which is another important lattice based cryptosystem. Actually NTRU is based on the algebraic structures of certain polynomial rings. It does not work explicitly with lattices but its security relies on the fact that for most lattices it is very difficult to find short

vectors. So, we shall make a connection with lattices in the security analysis of NTRU in the next chapter. Now, we introduce NTRU cryptosystem.

3.2 NTRU Cryptosystem

NTRU is a ring based public key cryptosystem which was first presented at the rump session of Crypto'96 by J.Hoffstein, J.Piper and J.H.Silverman [13]. It has been changed and optimized several times after its first version but the underlying principles remain the same. Here we introduce the encryption and decryption schemes as in the original paper [13].

By the invention of NTRU, it has been a promising alternative to the widely used public key cryptosystems. It has advantages on the subjects including speed, key generation and system requirements. We will give the comparison results of NTRU with other sytems in the next sections. Now, we introduce notations and definitions that will be used to describe the system.

3.2.1 Notation

The NTRU cryptosytem is built on polynomial algebra. The principal objects used by the system are polynomials with integer coefficients in the ring $R = \mathbb{Z}[X]/(X^N - 1)$ where N is an integer parameter. The basic tool of the system is the reduction of the polynomials with respect to two relatively prime integers. These integer parameters that are denoted by p and q need not be prime, but q is always considerably larger than p.

The computations in NTRU are made in the ring R. An element $a \in R$ will

be represented as a polynomial or a vector

$$a = \sum_{i=0}^{N-1} a_i x^i = [a_0, a_1, ..., a_{N-1}]$$

The multiplication in R is called *cyclic convolution product* and defined as

$$a * b = c \;\; with \; c_k = \sum_{i=0}^{k} a_i b_{k-i} + \sum_{i=k+1}^{N-1} a_i b_{N+k-i} = \sum_{i+j \equiv k (mod N)} a_i b_j$$

When this multiplication is done modulo q, it means to reduce the coefficients of the polynomials modulo q. The ring that the operations are done modulo q will be denoted by $\mathbb{Z}[\mathbb{X}]/(q, X^N - 1)$. The computation of a product of two polynomials with degree N requires N^2 multiplication. Therefore NTRU uses polynomials with small coefficients to compute the multiplications rapidly.

Another operation used in encryption scheme is taking inverse of a polynomial. The *inverse* of a polynomial a in $\mathbb{Z}[X]/(q, X^N - 1)$ is a polynomial $a^{-1} \in \mathbb{Z}[X]/(q, X^N - 1)$ such that $a * a^{-1} = a^{-1} * a = 1$. Note that some polynomials are not invertible in the ring $\mathbb{Z}[X]/(q, X^N - 1)$. But it is proven that with high probability a randomly chosen polynomial $f \in \mathbb{Z}[X]/(q, X^N - 1)$ which is required to satisfy $f(1) = 1$ is invertible in $\mathbb{Z}[X]/(q, X^N - 1)$ [15].

3.2.2 Key Generation

To create the public key, a private key f is chosen randomly from the set L_f which is a subset of R. The polynomial f should have inverses modulo q and modulo p which are denoted by f_q^{-1} and f_p^{-1}, respectively. As we said, with high probabilty f has inverses [15] and the actual computation of these inverses is

easy [17]. If the inverses of f does not exist, we choose a new f and try again. Then we choose a random polynomial g from the set L_g which is a subset of R and the corresponding public key h is computed as:

$$h \equiv f_q^{-1} * g \ (mod\ q) \tag{3.2.1}$$

Thus, we have h as the public key and f as the private key, also the polynomials f_p^{-1}, f_q^{-1}, g are kept secret.

3.2.3 Encryption

A message m is choosen from a set of plaintexts L_m, a subset of R. Then a random polynomial r is picked from a set L_r which is again a subset of R and the message is encrypted as follows:

$$e \equiv pr * h + m \ (mod\ q) \tag{3.2.2}$$

where e is the ciphertext of the message. Since a random polynomial is used in encryption, the message m can be encrypted into many different ciphertexts.

3.2.4 Decryption

After receiving the ciphertext, we use the private key f and f_p^{-1} to recover the message:

$$a \equiv f * e \ (mod\ q) \tag{3.2.3}$$

If the coefficients of a are not in the interval $(-q/2, q/2]$ then we change each of them as to be in this interval. Now, we compute

$$c \equiv f_p^{-1} * a \ (mod\ p)$$

For appropriate parameters, there is a high probability that decryption works and c will be equal to the message m [13]. The reason of the change of the coefficients a_i's after the equation 3.2.3 is to prevent the decryption failures. The designers of NTRU have noticed these decryption failures and recommended some sets of parameters that we give later.

Why Decryption Works

If we elaborate the computations above, we see that:

$$\begin{aligned} a &\equiv f * e \quad (mod\ \ q) \\ &\equiv f * (pr * h + m) \quad (mod\ \ q) \\ &\equiv f * [pr * (f_q^{-1} * g) + m] \quad (mod\ \ q) \\ &\equiv pr * g + f * m \quad (mod\ \ q) \end{aligned}$$

Now consider the last equivalence. If the appropriate polynomials, that will be explained soon, are choosen in the encryption, the coefficients of a will lie in an interval of length less than q, so the coefficients will not change when they are reduced modulo q. Hence a will be exactly equal to $(pr * g + f * m)$. When we

reduce a modulo p we obtain:

$$a \equiv pr * g + f * m \equiv f * m \ \ (mod\ p)$$

Lastly, multiplication of $f * m$ with f_p^{-1} in modulo p retrieves the message m (mod p).

The designers of NTRU defined sample spaces, that already denoted as L_f,L_g,L_m and L_r, for the corresponding polynomials to avoid the failures in decryption [13]. First, the space of messages consisting of polynomials modulo q is defined as:

$$L_m = \left\{ m \in R : \text{m has coefficients lying between } \frac{-1}{2}(p-1) \text{ and } \frac{1}{2}(p-1) \right\}$$

To describe the other sample spaces, the following representation of the sets is used:

$$L(d_1, d_2) = \{ f \in R : f \text{ has } d_1 \text{ coefficients equal } 1, d_2 \text{ coefficients equal } -1, \text{the rest } 0 \}$$

With this notation, the sample spaces of f,g and r are defined by choosing three integers d_f, d_g and d_r respectively.

$$L_f = L(d_f, d_f - 1),\ L_g = L(d_g, d_g) \text{ and } L_r = L(d_r, d_r)$$

The polynomial f do not have the same number of coefficients equal to 1 and -1, because a polynomial satisfying $f(1) = 0$ can never be invertible.

When we choose the polynomials to be used in encryption from sample spaces defined as above, the coefficients of the polynomial a will be small enough, so that they will lie between $-q/2$ and $q/2$. In other words the polynomial a will be

exactly equal to the polynomial $pr * g + f * m$. This will thwart the decryption failures.

3.2.5 Parameter selection

As we have seen, NTRU has three integer parameters N, p, q and four sets L_f, L_g, L_r, L_m which depend on integers d_f, d_g, d_r and p. The norms of f and g or the parameters d_f and d_g related to norms are chosen by considering decryption failures. If proper parameters are chosen, the probability of decryption failures is less than 10^{-5} [16]. The suggested parameters for different security levels are in the following table [13].

Table 3.1: Parameter sets of the first version of NTRU.

	N	q	p	d_f	d_g	d_r
Moderate	167	128	3	61	20	18
Standart	263	128	3	50	24	16
Highest	503	256	3	216	72	55

3.2.6 Comparison with other PKCS's

According to the results of the experiments performed by the NTRU company, the following table gives the comparison of NTRU with other public key cryptosystems, RSA and ECC. The security level $N = 251$ is comparable to RSA security level 1024 and ECC security level 163. Key lengths, encryption and decryption speeds of the systems are considered.

Due to the results in Table 3.2, we can observe that the encryption of NTRU is faster than RSA and ECC, because the underlying operation used by NTRU

Table 3.2: Comparison of NTRU with RSA and ECC.

		$NTRU251$	$RSA1024$	$ECC163$
public key	bits	2008	1024	164
secret key	bits	251	1024	163
plaintext block	bits	160	702	163
ciphertext block	bits	2008	1024	163
encrypt speed	blocks/sec	22727	1280	458
decrypt speed	blocks/sec	10869	110	702

can be performed much more rapidly than the underlying operations of RSA and ECC. Also NTRU involves small numbers where others use large numbers with hundred of digits.

The key sizes of NTRU are about the same as with RSA and ECC. However the public key of NTRU is twice the length of the public key of RSA. When we consider the size of plaintext and ciphertext blocks, NTRU has a big difference between them. Its ciphertext size is much more greater than the plaintext size, because the ciphertext polynomial has coefficients as large as q where the message representative is a binary string.

Chapter 4

Attacks on NTRU

There have been several attacks based on different aspects of NTRU since its presentation. NTRU has been modified and improved many times due to these attacks, but the main principals remained the same. The security parameters are determined after experiments performed by NTRU company to defend the system from the most effective attacks which we shall present in this chapter. At the moment, NTRU is proven as secure against these attacks and it is standardized as Efficient Embedded Security Standards(EESS#1) [1]. This standard includes relevant information to assist in the development and interoperable implementation of NTRUEncrypt and NTRUSign, including security considerations. See [1] for supported parameter choices for the current version of NTRU.

Now we introduce the brute force attack which is the most general attack for a cryptosytem.

4.1 Brute Force Attack

As given in chapter 3, the key generation of NTRU is performed by choosing two polynomials f and g from the sets L_f and L_g with parameters d_f and d_g which are known publicly. An attacker apply brute force attack on an NTRU private key by enumerating all possible $f \in L_f$ and computing $f * h \bmod q$ which gives

the polynomial g. Since he knows the parameter d_g, he tests if the polynomial retrieved by the multiplication has small entries. In other way, he may try all possible $g \in L_g$ and check if the coefficients of $g * h^{-1} \ (mod\ q)$ are small. When we look into the parameter sets of NTRU, we notice that d_g is smaller than d_f(see Table 3.1). The search space L_g will be smaller than L_f. Therefore, in practice it is more reasonable to test all $g \in L_g$.

If we enumerate all g's and compute $g * h^{-1}$ then the search time will be computed as $|L_g| = \binom{N}{d_g}$.

Similarly, one can also apply the brute force attack on the message. As we know, the message is equal to $e - r * h \bmod q$. By trying all possible $r \in L_r$ and testing if the given equation has small coefficients, we can get the message. The search time will be equal to $|L_r| = \binom{N}{d_r}$.

When N is chosen as a large number, this attack will not be efficient because of the large search spaces. Now, we describe the meet in the middle attack which decreases the search time of the brute force attack by taking its square root.

4.2 Meet in the Middle Attacks

First Andrew Odylzko pointed out that a meet in the middle attack can be applied by using the random polynomial r of NTRU. Then the designers of NTRU observed that this attack can be also used against the private key f [14]. Before explaining the attack, let us give these useful remarks.

Remark 4.2.1. *In section 3.2.1, we have represented an element $a \in R$ as a polynomial $a_0 + a_1x + ... + a_{N-1}x^{N-1}$ and a vector $[a_0, a_1, ..., a_{N-1}]$. Then, the*

i^{th}-rotation of a in R is given as $a * x^i$ and

$$a^i = [a_{N-i}\ ,\ a_{N-i+1}\ ,...,\ a_{N-1}\ ,\ a_0\ ,\ a_1\ ,...,\ a_{N-i-1}]$$

Remark 4.2.2. *In the key generation of NTRU, if $f * h \equiv g \bmod q$ (it is equal to $h \equiv f_q^{-1} * g \bmod q$), then the i^{th}-rotations of f and g also satisfies the equivalence, i.e. $f * x^i * h \equiv g * x^i \bmod q$. So, if we find an (f, g) pair satisfying $f * h \equiv g \bmod q$ and including the same number of d_f and d_g with the original key, then it can be one of the rotations of the original key pair. In other words, once we have found one rotation of f, we have only N possibilities left for the private key f.*

We shall describe the attack in the case f and g contains binary coefficients and N and d_f are even. The modifications for odd values are easy. The idea is to divide f into two pieces $f_1||f_2$, where the symbol $||$ denotes concatenation. Each f_1 and f_2 has length $N/2$ with $d/2$ ones. An important point here is the fact that the number of ones of f may not spread into two parts equally, but it is easy to show that at least one of the rotations of f has this property [14]. Then, the attack continues as follows:

$$\begin{aligned} f * h &\equiv g \pmod{q} \\ (f_1||f_2) * h &\equiv g \pmod{q} \\ f_1 * h + f_2 * h &\equiv g \pmod{q} \end{aligned}$$

As the i^{th} coordinate of g is 0 or 1, we have the following equivalence at the i^{th}

coordinates of $f_1 * h$ and $f_2 * h$

$$(f_1 * h)_i \equiv \{0, 1\} - (f_2 * h)_i \pmod{q}$$

The attacker first enumerates f_1's. There will be $\binom{N/2}{d/2}$ different choices for the polynomial f_1. Then he computes $f_1 * h \pmod{q}$ and uses first k coordinates of this polynomial to store each f_1 into an array, which will be called a bin, labelled by binary numbers based on these coordinates. For each entry of the k coordinates, the following is done: If the value of the coefficient is between 0 and $q/2$ then the corresponding entry of the bin will be 0, if the value of the coefficient is between $q/2$ and q then the corresponding entry will be 1. Once all the k bits are computed, the polynomial f_1 is stored in the bin labelled by this k-bit string. There can be 2^k bins at most and the attacker has $\binom{N/2}{d/2}$ possible f_1's. Hence, at the beginning he should choose k such that $\binom{N/2}{d/2} < 2^k$ for a sufficient number of bins to store maximum possible polynomials.

In the second step, the attacker enumerates f_2's and computes $f_2 * h$. There are again $\binom{N/2}{d/2}$ different f_2's. The attacker will search the correct (f_1, f_2) pair satisfying $(f_1 * h)_i \equiv \{0, 1\} - (f_2 * h)_i \pmod{q}$ for $0 \leq i < k$. But the polynomials f_1's are stored in the bins, so he searches for the correct bins. For any polynomial f_2, he computes the bin value of $-f_2 * h$ by using the rule that is used to determine the entries of the bin in the first step and then goes to the same bin that contains f_1's. But for each coefficient of $-f_2 * h$ there is a possibility of being added by 1. So if there is a critical coefficient that the bin value can be changed by adding 1 to the corresponding coefficient of $-(f_2 * h) \pmod{q}$, then the attacker also goes to that possible bin. Finally he takes f_1's from those bins and checks if

$(f_1||f_2) * h \ mod \ q$ is binary or not. If it is a polynomial with binary cefficients then the private key f is found. This is repeated for each f_1 contained in the bin. The attacker searches over all f_2's untill he finds the private key.

The search space of this attack is smaller than the search space of brute force attack, since f is divided into two parts. By storing f_1's in the bins, the attacker need not to check all f_1's, he checks only the ones in the corresponding bins. As we have mentioned, at least one rotation of f has $d/2$ ones in the first $N/2$ entries and $d/2$ ones in the second half. By the experiments it is shown that there are more than $\sqrt{N}$ rotations of f having this property. By this observation and some improvements, the running time and the storage requirement of the meet in the middle attack is given as follows:

$$\frac{\binom{N/2}{d/2}}{\sqrt{N}}$$

See the technical report [14] for the detailed computation of the running time.

4.3 Lattice Attacks

As we mentioned before, NTRU is a lattice based cryptosytem and its security relies on the lattice problem, SVP(see chapter 2). The designers of NTRU showed that the secret key could not be found by computing the shortest vector of the lattice with the LLL algorithm, since it was surrounded by a cloud of exponentially many unrelated lattice vectors. However, after the presentation of NTRU, Coppersmith and Shamir presented another lattice-based attack, which either finds the original secret key f or an alternative key f' which can be used

in place of f to decrypt ciphertexts [10]. The goal of the attack is to find the private key by constructing a lattice using public key h and applying lattice reduction algorithms to this lattice. After this attack, NTRU changed the security parameters and improved the system against lattice attacks. We will analyze the lattice security of the system in the next chapter. Now, we give the construction of the NTRU lattice.

4.3.1 The Standart NTRU Lattice

One knows the public informations N, p, q, d_f, d_g and h. By using public key h, a set of vectors L can be described as follows:

$$L = \left\{(a, b) \in Z^{2N} : a * h = b \ \ mod \ \ q\right\}$$

It is obvious that L is a lattice and the vector (f, g) is contained in L. The length of this vector due to the proposed parameters of NTRU will be $\sqrt{2d_f - 1 + 2d_g}$ which is relatively small compared to N. Therefore this vector will be a short vector of the lattice L.

Coppersmith and Shamir found a basis which is shown by the rows of a $2N \times 2N$ matrix composed of four blocks and constructed a lattice generated by this basis [10]. The lattice is called NTRU lattice and the basis matrix generating the

lattice is denoted by L^{NT}.

$$\mathbf{L^{NT}} = \left(\begin{array}{cccc|cccc} \lambda & 0 & \dots & 0 & h_0 & h_1 & \dots & h_n - 1 \\ 0 & \lambda & \dots & 0 & h_n - 1 & h_0 & \dots & h_n - 2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda & h_1 & h_2 & \dots & h_0 \\ \hline 0 & 0 & \dots & 0 & q & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & q \end{array}\right)$$

In this matrix, h_i's are the ith coefficients of the polynomial h. λ is a small constant that is chosen to make the lattice reduction more efficient. It is obvious that NTRU lattice contains all the elements of the set

$$L = \left\{(\lambda a, b) \in Z^{2N} : a * h = b \ \ mod \ \ q\right\}$$

and so $(\lambda f, g)$ is in the NTRU lattice. The length of this vector is $\sqrt{(2d_f - 1)\lambda^2 + 2d_g}$. Since d_f, d_g and λ are small numbers, then $(\lambda f, g)$ is a short vector of the NTRU lattice. Now, a natural question is what happens if there are short vectors other than the private key in the NTRU lattice. Is it true that a sufficiently short vector (a, b) in the NTRU lattice, i.e. $a * h \equiv b \ mod \ q$ can be used to decrypt the messages? Coppersmith and Shamir showed that it is not necessary to find the private key. If we find a vector (a, b) of NTRU lattice which is not longer than 2.5 times the length of the private key, we can also use it to decrypt the messages.

However, the designers of the NTRU performed experiments and implemented lattice basis reductions to the NTRU lattice and they never found any vector possible to be used for decryption other than the private key [13]. Therefore lattice reduction algorithm has no chance except finding the actual private key.

As we discussed in section two, Gaussian heuristic gives the shortest vector of a lattice as approximately $\sqrt{n/2\pi e}(detL)^{1/n}$. In our NTRU lattice, the dimension is $2N$ and the determinant is $q^N\lambda^N$. If we substitute these into the equation, the length of the shortest vector of the NTRU lattice will be approximately

$$\sigma(L) = \sqrt{\frac{N\lambda q}{\pi e}} \tag{4.3.1}$$

where λ is the constant chosen when constructing the NTRU lattice, so the attacker should decide the value of λ.

Now, let τ denote the vector $(\lambda f, g)$, which will be called the target vector and we try to find it by using the lattice reduction. We define a lattice constant c_h as the ratio of the length of the expected shortest vector to the length of the target vector, i.e. $c_h = ||\sigma(L)||/||\tau||$. By substituting lengths of the vectors in NTRU lattice, the constant c_h is

$$c_h = \frac{\sqrt{\frac{N\lambda q}{\pi e}}}{\sqrt{\lambda^2||f||^2 + ||g||^2}} \tag{4.3.2}$$

Practical experiments have shown that as this constant gets smaller, lattice reduction methods are observed to work best. Therefore the attacker tries to make this constant as small as possible by choosing λ. It is easily checked that to maximize the search efficiency for τ, the choice for this constant shold be $\lambda = ||g||/||f||$.

Since the attacker knows d_f and d_g values, he can compute λ and makes the lattice reduction to find the target vector easier [19].

Although reducing the lattice constant makes LLL work easier, the time required to find a short vector is still exponential in N. Therefore after the ideas of Coppersmith and Shamir, other attacks have been proposed. Now we will give the remarkable ones.

4.3.2 Zero Run Lattice

Since we know d_f and d_g, we can make a guess about the components of target vector $(\lambda f, g)$. Alexander May introduced new results in the cryptanalysis of NTRU by lattice reduction [5]. He observed that there are many zero coefficients in polynomials f and g and this can be used as an advantage in lattice attack. He assumes that the polynomial g has r consecutive coefficients which are equal to zero, in other words he defines this as r length zero run. The places of these zeros are not important, since all rotations of the private key are in the lattice. The idea of May is basicly to multiply r consecutive columns of L^{NT} by some large number θ. This lattice is called zero run lattice and the basis matrix is denoted

as L_θ^r.

$$\mathbf{L}_\theta^{\mathbf{r}} = \left(\begin{array}{cccc|cccccc} \lambda & 0 & \dots & 0 & h_0\theta & \dots & h_{r-1}\theta & h_r & \dots & h_{n-1} \\ 0 & \lambda & \dots & 0 & h_{n-1}\theta & \dots & h_{r-2}\theta & h_{r-1} & \dots & h_{n-2} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \lambda & h_1\theta & \dots & h_r\theta & h_{r+1} & \dots & h_0 \\ \hline 0 & 0 & \dots & 0 & q\theta & \dots & 0 & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & \dots & 0 & 0 & \dots & q \end{array}\right)$$

The effect of multiplying the columns by θ is to force the lattice reduction algorithm to find the shortest vector which has zero coefficients in those r coordinates. Otherwise in a lattice vector if any of those r coordinates is nonzero than then in the length of the vector there will be a θ multiple, so the vector cannot be a short vector of the lattice. This idea reduces the search space of short vectors.

After this idea, by a technical report Silverman suggested that rather than multiplying r consecutive columns with a θ multiple, multiplying r random columns by θ will be more adventageous [18].

4.3.3 Zero Forced Lattice

Silverman [18] generalized May's idea and introduced zero-forced lattices that has better performance than zero-run lattices. He basically aims encouraging the lattice reduction to find short vectors with zeros in choosen coordinates like May. But he tries to create a smaller dimensional lattice instead of multiplying

coordinates with a large number.

In this attack, the first step is to choose a random set of indices for coefficients of g. Let $J = \{j_1, ..., j_r\}$ denote the set of indices of the r randomly chosen coefficients in g. Then these randomly chosen coefficients are assumed to be zero and corresponding r congruences in $f*h \equiv g \bmod q$, involving only the coefficients of f and h, will be forced to equal zero. But, initially we write the N original congruences hold for the vectors in L^{NT}

$$g_j \equiv \sum_{i+k\equiv j} h_k f_i \pmod{q} \text{ for } 0 \leq j < N,$$

By assuming $g_j = 0$, where $j \in J$, we get r linear relations modulo q. We solve these r congruences for $f_{N-r}, ..., f_{N-1}$ in terms of $f_0, ..., f_{N-r-1}$ and substitute back in the remaining $N - r$ congruences. We get the following system.

$$a_{0j}f_0 + a_{1j}f_1 + ... + a_{N-1-r,j}f_{N-1-r} \equiv g_j \pmod{q} \text{ for } 0 \leq j < N,\ j \notin J$$

Here the a_{ij}'s are known quantities, and the $f_0, ..., f_{N-r-1}$'s and g_j's are the unknown quantities. This is a system of $N-r$ congruences in $2(N-r)$ unknowns, so we can construct a lattice defined as *Zero-Forced Lattice* with dimension $2(N-$

r). The basis matrix of the lattice which is denoted by L_J^{ZF} is in the following.

$$\mathbf{L_J^{ZF}} = \left(\begin{array}{cccc|cccc} \lambda & 0 & \dots & 0 & & & & \\ 0 & \lambda & \dots & 0 & & & & \\ \vdots & \vdots & \ddots & \vdots & & & a_{ij} & \\ 0 & 0 & \dots & \lambda & & & & \\ \hline 0 & 0 & \dots & 0 & q & 0 & \dots & 0 \\ 0 & 0 & \dots & 0 & 0 & q & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 0 & 0 & 0 & \dots & q \end{array}\right)$$

The last step of the attack is applying the lattice reduction algorithm to this lattice and trying to find the target vector (f, g) where the $j \in J$ coordinates of g are equal zero. Since the dimension of this lattice is smaller than the standart NTRU lattice, the running time of the lattice reduction algorithm will decrease.

4.3.4 Dimension Reduced Lattice

Alternative to zero forced lattices, May also suggested discarding a number of columns from the basis matrix [5]. This reduces the dimension of the lattice and speeds up the lattice reduction. In this attack the coordinates are again chosen randomly and a new lattice is constructed from the lattice L^{NT} by discarding those randomly chosen coordinates. By applying lattice reduction to the new lattice, we don't get the whole target vector since the coordinates that are discarded at the beginning are unknown. Hence the length of the target vector will be smaller than its actual length. Silverman [18] claims that by this attack the

length of the target vector gets closer to the expected shortest vector's length, theoretically lattice constant c_h gets closer to 1. This means that the lattice reduction algorithm will have difficulty to pick out the target vector from other short vectors which has a length approximately expected shortest vector. In the experiments performed by May [5], the speed of reduction algorithm increases in low dimensions, but as N exceeds 100 the speed of the reduction algorithm is not substential enough to break the system by applying this attack.

These were the methods of improving the NTRU lattice to speed up the lattice reduction algorithms to find the target vector. These methods works properly in low dimensional lattices. By the experiments performed, it is shown that when N exceeds 100, the lattice reduction algorithms get slower and the breaking of the system gets harder. In the next chapter, we analyze the security of NTRU and investigate the running times of the two primary attacks given in this chapter,i.e.meet in the middle attack and lattice attacks.

Chapter 5

Analyzing the Security of NTRU

To determine the security of a public key cryptosystem, first the trapdoor function underlying the system should be investigated. For instance in RSA [30], the security of the system relies on the complexity of the integer factorization problem and the modulus of RSA is chosen as an integer big enough that can not be factorized in the amount of time required by the desired level of security. In the case of NTRU, there are two main attacks that try to invert the trapdoor function: meet in the middle attack and lattice attacks. So the NTRU parameters should be chosen in such a way that neither of these attacks is successful for the security level determined before. In 2005, NTRU company published a paper that presents an algorithm to generate all required parameters for the last version of NTRU [26]. This paper specifies the parameter bounds implied by the effectiveness of the attacks and generates an algorithm by considering these bounds for different levels of security. Now, we introduce two security types of NTRU that are defined after the attacks given in the previous chapter.

5.1 Combinatorial Security

This is the security of the cryptosystem due to the combinatorial attacks that we have presented as brute force and meet in the middle attacks for NTRU. In chapter 3, it is given that private key polynomials are drawn from a known space L_f. So an attacker can use a combinatorial technique to search this space like brute force attack or meet in the middle attack. As already noted, the meet in the middle attack has a better running time than the brute force attack. Hence, the combinatorial security of NTRU is defined as greater than or equal to the running time of the meet in the middle attack which is given as $\binom{N/2}{d/2}/\sqrt{N}$ in the chapter 4.

5.2 Lattice Security

In chapter 4, we introduced the NTRU lattice and we showed how the lattice reduction algorithms can be used to find the private key. When analyzing the lattice security, it is indispensable to say that these algorithms often behave better than the theoretically proven results. Therefore, to see the running time of lattice reduction algorithms applied to NTRU lattices, a series of tests and experiments have been done by NTRU company. These are published as a technical report in NTRU website [33] and the details of the results can be found in [19]. Now we will give a brief summary about the lattice security.

There are two lattice constants defined, which are relevant to the shortest vector problem. These are:

$$a = \frac{N}{q}$$

$$c = \sqrt{\frac{4\pi e||f||||g||}{q}}$$

where the parameters N, q and the polynomials f, g are as given in chapter 3. After introducing the NTRU lattice, we have given a constant c_h which is the ratio of the length of the expected shortest vector to the length of the target vector and given by equation 4.3.2. By experiments, it is seen to be used in determining the performance of LLL reduction. Recall that as it gets smaller, LLL finds the target vector easier. Now we will write c in terms of c_h and explain the observations about these constants.

$$c = \frac{\sqrt{2N}}{c_h}$$

By the various experiments about the NTRU lattice done over the years, some results related to a and c are found. If we hold a and c constant, while increasing N then the logarithm of the time for LLL to find the target vector grows at least linearly with N. Let T denote the time and A, B be constants then

$$\log T \geq AN + B$$

It is not quantified yet but by the experiments it is enounced that A and B depend on c and a such that as either c or a or both increase the constant A increase. This means that if these constants increase, the time to find the target vector will increase. Thus, c, a and N provide a measure of the difficulty of finding the target vector.

These are all experimental results and accurate mathematical models for esti-

mating the time of the lattice reduction algorithms are still not found. Therefore, this analysis has been accepted for the lattice security of the NTRU. As a result, by using suggested parameters, NTRU seems secure against the lattice attacks unless there are new ideas in lattice reduction algorithms.

5.3 An Example on Lattice Security of NTRU

In this section, our aim is to perform some implementations on the small dimensional NTRU lattices by using LLL and BKZ reduction algorithms. We shall try to find out the target vector $\tau = (\lambda f, g)$ of an NTRU lattice when N is not too big. Also, we shall give the estimated times of the algorithms to break the NTRU lattice. We have used Maple computer algebra package [2] to compute NTRU polynomials. Then to perform the reduction algorithms, we have worked on C++ that includes Victor Shoup's NTL(Number Theory Library) package [31].

We first choose the NTRU parameteres $\{N, p, q, d_f, d_g\}$ and the polynomials $\{f, g\}$ from the sets L_f and L_g. Then we compute the public key h and try to find the private key by using the polynomial h. We will use the vector representations of the polynomials in the following example.

Example 5.3.1. Let us use the following parameters:

$N = 51$

$q = 64$

$p = 3$

$d_f = 9$

$d_g = 8$

We choose randomly a polynomial f from $L(9, 8)$ as the private key and a

polynomial g from $L(8,8)$. When we are choosing the polynomial f, we should beware of it to have inverses in modulo 64 and modulo 3.

$f = [0,-1,0,1,0,0,0,-1,0,1,0,0,0,-1,0,1,0,0,0,1,-1,0,1,0,0,0,-1,0,0,$
$1,0,0,-1,0,0,1,0,0,0,0,-1,0,1,0,0,1,0,0,0,0,-1]$

$g = [0,1,-1,0,0,0,0,0,1,0,-1,0,0,-1,0,1,0,0,0,-1,1,0,-1,0,0,0,0,-1,$
$0,1,0,-1,0,0,0,0,0,0,0,1,0,1,0,0,0,0,0,1,-1,0,0]$

The public key polynomial is computed as $h = f_q^{-1} * g \ \ (mod\ q)$ and this gives the following coefficient vector:

$h = [53,9,0,10,27,2,44,30,62,23,45,2,35,53,55,21,0,5,60,38,52,40,14,2,$
$21,23,50,2,46,1,54,27,5,60,16,40,62,49,42,46,8,0,4,36,26,11,27,52,30,36,16]$

Now, suppose we do not know polynomials f and g. We only have the public information h, p, q, d_f and d_g. To find the private key, we first construct the NTRU lattice by using the coefficients of h as given in the matrix L^{NT} in chapter 4. In our case, the dimension of the matrix is 102 and we take the λ value as 1 since $||g||/||f||$ is approximately 1.

Now we try to find the target vector of the lattice, then we can get the private key, i.e. $\tau = (f||g)$. We have run the lattice reduction algorithms LLL and BKZ with a block size 2, to compare the results with each other. We take the basis matrix of NTRU lattice as an input and we get the following vector as output from the both reduction algorithms. Let's denote this vector as $\tau' = (f'||g')$. :

$\tau' = [0,0,0,1,0,-1,0,0,0,-1,1,0,-1,0,0,0,1,0,0,-1,0,0,1,0,0,-1,0,0,0,0,1,$
$0,-1,0,0,-1,0,0,0,0,1,0,1,0,-1,0,0,0,1,0,-1 \ ||\ 1,0,0,1,0,-1,0,0,0,1,-1,0,1,$
$0,0,0,0,1,0,-1,0,1,0,0,0,0,0,0,0,0,-1,0,-1,0,0,0,0,0,-1,1,0,0,0,-1,1,0,0,0,$
$0,0,-1,0]$

Observe that $f \neq f'$ and $g \neq g'$, but if we examine carefully, each f' and g' is

the negative of the rotation of f and g vectors where each rotates 10 components left. We checked that f' has inverses in modulo q and modulo p. This means that the attacker can use this vector as the private key of the system and decrypt the messages.

An attacker, who will apply a lattice attack to an NTRU private key, performs these computations and finds a small vector of the lattice; but he cannot verify if the vector is the exact target vector or not. However he gets some idea by checking d'_f and d'_g values. If these match with the exact values, then he tries to find the inverses of f' in modulo q and modulo p.

In our example, it did not take too much time to find the target vector, since the N value is small. We got the same results from the reduction algorithms LLL and BKZ after a running time of 3.79 and 3.92 seconds, respectively.

In this example, we have chosen N as a small value to succeed in the attack and to be able to find the target vector. According to the parameters given in chapter 3, for the moderate security, the N value should be 107 and due to the standart EESS#1 [1], the recommended parameter sets are given as ees251ep4 and ees251ep5 both of which have $N = 251$ and $q = 259$.

Chapter 6

Conclusion

In this thesis, we studied the NTRU public key cryptosytem. After the presentation of basic properties of lattices, we introduced the well known lattice problems and lattice reduction algorithms. We focused on the NTRU cryptosytem and studied key generation, encryption and decryption schemes. We investigated the most severe attacks against the system. We showed the connection between the lattice problems and the NTRU lattice.

In the implementation part of this work, we applied an attack to a small dimensional NTRU lattice and found a rotation of the target vector which can also be used as a private vector.

References

[1] Efficient Embedded Security Standard (EESS) version 2. *Consortium for Efficient Embedded Security*, June 2003.

[2] Maple 9.01. www.maplesoft.com.

[3] L.Lovasz A.K.Lenstra, H.W.Lenstra. Factoring polynomials with rational coefficients. *Math.Ann.*, 261, 1982.

[4] A.Korkine and G.Zolotarev. Sur les formes quadratiques. *Math.Ann.*, pages 336–383, 1873.

[5] A.May. Cryptanalysis of NTRU. *Unpublished preprint*, available at http://informatik.uni-frankfurt.de/ alex/ntru.ps, 1999.

[6] C.Dwork. Lattices and Their Application to Cryptography. *Lecture Notes, Stanford University*, 1998.

[7] C.F.Gauss. Disquisitiones arithmeticae. 1801.

[8] C.P.Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Therotical Computer Science*, 53:201–224, 1987.

[9] C.P.Schnorr and M.Euchner. Lattice basis reduction: Improved practical algorithms and solving subset sum problems. *Math.Prog.*, 66:181–199, 1994.

[10] D.Coppersmith and A.Shamir. Lattice attacks on NTRU. In *Proc. of Eurocrypt'97*, volume 1233 of LNCS. Springer Verlag, 1997.

[11] H.Cohen. *A Course in Computational Algebraic Number Theory.* Springer-Verlag, New York, 1995.

[12] J.-Y.Cai. The complexity of some lattice problems. In *Proc. of ANTS-IV, volume 1838 of LNCS.* Springer-Verlag, 2000.

[13] J.H.Silverman J.Hoffstein, J.Piper. NTRU: A ring based public key cryptosystem. *Lecture Notes in Computer Science, Springer-Verlag*, pages 267–288, 1998.

[14] J.H.Silverman. A meet in the middle attack on an NTRU private key. *Technical Report 4, NTRU cryptosystems*, Available at [33], 1997.

[15] J.H.Silverman. Invertibility in truncated polynomial rings. *Technical Report 9, NTRU cryptosystems*, Available at [33], 1998.

[16] J.H.Silverman. Wraps, gaps, and lattice constants. *Technical Report 11, NTRU cryptosystems*, Available at [33], 1998.

[17] J.H.Silverman. Almost inverses and fast ntru key creation. *Technical Report 14, NTRU cryptosystems*, Available at [33], 1999.

[18] J.H.Silverman. Dimension-reduced lattices, zero-forced lattices and the NTRU public key cryptosystem. *Technical Report 13, NTRU cryptosystem*, Available at [33], 1999.

[19] J.H.Silverman. Estimated breaking times for NTRU lattices. *Technical Report 12, NTRU cryptosystems*, Available at [33], 1999.

[20] L.Lagrange. Recherches d'arithmetique. *Mem. Acad*, 1773.

[21] L. Lovasz M. Grotschel and A. Schrijver. Geometric algorithms and combinatorial optimization. pages 139–156. Springer-Verlag, 1991.

[22] M.Ajtai. Generating hard instances of lattice problems. *ECCC*, TR96-007, 1996.

[23] M.Ajtai. The shortest vector problem in L_2 is NP-hard for randomized reductions. *In Proc. of 30th STOC. ACM*, 1998.

[24] C.Dwork M.Ajtai. A public key cryptosystem with worst-case/average-case equivalance. *In Proc. of 29th STOC*, pages 284–293, 1997.

[25] D. Micciancio. *Lattices in cryptography and cryptanalysis*. World Scientific, 1995.

[26] W. Whyte N. Howgrave-Graham, J. H. Silverman. Choosing parameter sets for NTRUEncrypt with NAEP and SVES-3. *Topics in cryptologyCT-RSA*, 2005.

[27] S.Halevi. O.Goldreich, S.Goldwasser. Public key cryptosystems from lattice reduction problems. In *Proc. of Crypto 97, volume 1294 of LNCS*, pages 112–131. Springer-Verlag, 1997.

[28] S.Safra O.Goldreich, D.Micciancio and J.-P.Seifert. Approximating shortest lattice vectors is not harder than approximating closest lattice vectors. Available at http://www.eccc.uni-trier.de/eccc/.

[29] J.Stern P.Nguyen. Cryptanalysis of the Ajtai-Dwork cryptosystem. In *Advances in Cryptology – Proceedings of CRYPTO '98*, pages 223–242. vol. 1462 of LNCS, Springer-Verlag, 1998.

[30] L. M. Adleman R. Rivest, A. Shamir. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM 21*, pages 120–126, 1978.

[31] V.Shoup. Number Theory C++ Library (NTL) version 5.4.2. *available at http://shoup.net/ntl/*, last visited in August, 2008.

[32] M.Hellman W.Diffie. New directions in cryptography. *IEEE Transactions on Information Theory IT-22*, pages 644–654, 1976.

[33] www.ntru.com. Last visited in August, 2008.