BACHELOR

Klepto for post-quantum signatures

Thissen, Kimberley K.A.

Award date:
2016

TU/e University of Technology Eindhoven

2WH40 Bachelor Final Project

Klepto for Post-Quantum Signatures

Author:
Kimberley Thissen
k.k.a.thissen@student.tue.nl
Supervisor:
prof.dr. Tanja Lange

August 21, 2016

Abstract

Online communication is important to digitally express thoughts, ideas and actions to others. Therefore, everybody should be able to do that in a secure, and if needed private, way. People also should have the possibility to verify the identity of the other communicating party. This is where cryptography comes in. Signature schemes provide authentication and non-repudiation. There could be a third party involved, that wants to gain secret information, like the secret signing key or the message. This party could modify the scheme in such way that it leaks the information exclusively to this party every time the user signs a document. The study of designing private backdoors such that the user does not notice them is called kleptography. In this thesis, kleptography is investigated in post-quantum signature schemes.

First a short introduction is given on the subject. After that, the basic terminology of cryptography is explained. Also the title of this project is illustrated by an overview of the most important developments. Then the necessary mathematical background is given, in order to understand the technical parts in this thesis. The next section is about lattice-based signature schemes. Two well-known schemes are described in detail. In the central part, new backdoors in NTRU Signature Schemes are given, along with an analysis. At the end, the conclusions are presented.

Contents

1 Introduction

Most of the security online is based on *cryptosystems*: combinations of cryptographic algorithms meant for securing communications. Entities like enterprises and governments need a secure way to communicate, to exchange documents and to save important files. A suitable secure cryptosystem is used for that. In some cases, they want to make sure that a certain message comes from an authenticated source. This can be done with digital signatures. The signature is attached to the message so that the other party can verify it and has enough proof that the message is indeed from the intended party. Suppose there is a third party that wants information about those messages or about the generation of the signature. *Kleptography* denotes the study of modifying an implementation of a cryptosystem in such a way that it leaks secret information exclusively to the third party and the user cannot notice this subliminal channel. The cryptosystem or the signature system itself is in general assumed secure. In this thesis, *post-quantum signature schemes* are considered. These are signature schemes which are likely secure against attacks by a quantum computer. These quantum computers are breaking several schemes in use nowadays, so it is important that post-quantum systems are developed as soon as possible.

So is it indeed possible to extract secret information from a signature, and if so, what does it take to accomplish this? How does that relate to signatures?

1.1 Motivation

The questions above trigger a lot of interest in the subject. There are some cryptosystems which have been proven to have a great probability of being resistant against quantum computers. They are called post-quantum cryptosystems. Still, it might possible to leak secret information. The leak could come from a backdoor in the system. This backdoor is implemented in a program by a (malicious) developer. Only this person knows exactly how to get information out of these leaks. When the user has installed this program, and sends his messages and signatures over it, the developer is able to retrieve the secret information using publicly known parameters and intercepted signatures. This is possible, because the signatures contain additional information that only the developer can understand.

The goal of this thesis is to elaborate on existing post-quantum signature schemes, and to explore possibilities for a backdoor in such a scheme. In the coming sections extensive research is done on several systems, with the main focus on their digital signature schemes. Eventually some backdoors are given in NTRU Signature Schemes. In detail it is shown how such a construction can be designed.

1.2 Content of this thesis

In the coming section, some basic concepts will be explained. It forms a friendly introduction to cryptography and a necessary base for this thesis. Also some history and current developments in this field will be dealt with. The next section is the mathematical background, required for the technical parts of this thesis. Section 4 is about lattice-based

signature schemes. Some examples of algorithms to generate and verify a signature are presented. In section 5 the main focus lays on the signature schemes, called NTRUSign and NSS. In detail, the algorithms are described. Also new backdoors are shown, along with an analysis. In the end there will be the conclusions and some final remarks about the content of this thesis.

2 The world of cryptography

In this section the necessary information is given so that the reader is fully prepared for understanding the concepts of this thesis. First a couple of definitions are explained. Then a short introduction is given about post-quantum signature schemes and kleptography. Finally some current applications of signature schemes are described.

2.1 Terminology

In cryptography, there are some standard terms used when explaining how a cryptosystem works. These terms are described below.

2.1.1 Once upon a time... Alice, Bob, Eve and the crypto world

In this world, people want to have a secure way to communicate with each other. But what is this security exactly? It includes a couple of aspects of secure communication. These are called security properties [8]:

- *Confidentiality*: the message is and stays a secret to anyone else but the sender and the receiver.
- *Integrity*: the content of the message stays correct and cannot be changed during transmission.
- *Authentication*: the message is of undisputed origin.
- *Non-repudiation*: the message is undeniable by the sender.

Cryptosystems will cover some of these properties. A cryptosystem is a series of algorithms that provides security services. Alice and Bob are frequently used to represent the communicating parties. They are sending messages to each other, using the same cryptosystem. Then Eve is the third party interested in this communication. Eve can do a lot of things to jeopardize the security properties.

- She can *eavesdrop*, i.e. intercept and read the messages.
- She can perform a *modification*, i.e. she can change the message when it is in transmission to Bob.
- She can put on a *masquerade*, i.e. she can pretend to be Alice and send messages to Bob, using Alice's identity.

Therefore, Alice and Bob better use a good cryptosystem. A cryptosystem should be able to prevent Eve from listening in by encrypting the messages in a way that Eve cannot understand the message, it should be able to prevent changes from the outside and it should be able to authenticate Alice as a source. The science of making a cryptosystem like this is called *cryptography*.

2.1.2 Inside a cryptosystem

Every cryptosystem has the following components. Alice sends message m, which is the plain text, to Bob. This message is commonly represented by integers or bits. The message gets encrypted by the cryptosystem into a different form of text, which is called the ciphertext c. In a secure system, the ciphertext is made unreadable for everyone. The encryption is done by an encryption key k_1, unknown to third parties. Encryption is the process of encoding data so that one cannot recognize the original message anymore. Such a key is generated by the system. This process is denoted as:

$$E_{k_1}(m) = c. \tag{1}$$

Bob receives c, and decrypts this with decryption key k_2. Although related k_2 does not need to be the same as the encryption key. Now Bob can read Alice's message m. Decryption is the process of decoding data that has been encrypted in ciphertext, so that the data is understandable for the intended party. This process of encrypting message m under K_1 to get ciphertext c is denoted as:

$$D_{k_2}(c) = m. \tag{2}$$

When k_2=k_1 or when k_1 and k_2 are easily derived from one another, the cryptosystem is symmetric and the key is a shared secret between Alice and Bob. When they differ, the cryptosystem is asymmetric and in this case there is one private key, denoted by SK, and one public key, denoted by PK. The public key is used for encryption and the private key is used for decryption. So anyone can encrypt, but only the receiver can decrypt with his or her own private key.

The process of signing is denoted by:

$$\text{sign}_{SK}(m) = s \tag{3}$$

The process of accepting a signature by verification is denoted by:

$$\text{ver}_{PK}(m, s) = \begin{cases} 1, & \text{if } s \text{ meets requirements with input } PK \text{ and } m \\ 0, & \text{if } s \text{ does not meet requirements with input } PK \text{ and } m \end{cases} \tag{4}$$

The signature is accepted if equation (4) resulted in 1. The signature is denied if equation (4) resulted in 0.

If a signature scheme is involved, Alice can sign the message with her signature s, generated by her signing secret key. Then Bob can verify that signature, using the signing public key from Alice. Therefore, signature schemes are always asymmetric.

Note that the keys used for generating a signature are different from the keys used for encryption and decryption. Since this thesis only covers digital signature schemes from now on, the private and public keys are only used for generating signatures and verification. So SK denotes the signing private key and PK denotes the signing public key.

Using a sufficiently secure cryptosystem the messages cannot be read by third parties, this results in some kind of confidentiality. Alice can sign the message with a digital

signature, so Bob knows for sure the message comes from Alice. This results in some kind of authenticity. Since the signature depends on the message, the message cannot be modified. This results into some kind of integrity.

2.1.3 Encryption and signature schemes

Encryption schemes deal with the encryption and decryption processes. When dealing with asymmetric cryptography, anyone can encrypt a message given the public key, so the receiver cannot know for sure that a message came from a certain source. Signatures are used to authenticate the communicating sources. The verification of these signatures is a process of checking if the signature sent with the message is really from the intended source and if the signature matches the obtained message. In an encryption scheme, the parameters are set and the steps to generate keys, to perform encryption and to perform decryption are described. In a signature scheme, the parameters are set, the steps to generate keys, to generate a signature for a message and the steps for the verification are described. The security of a signature scheme depends on attack results. These results could be being able to recover the secret signing key, forging a signature for a chosen message, retrieving a valid signature for a certain message and retrieving some pairs of signature and message not already known to the attacker.

A secure signature scheme will result in secure signatures. A verified digital signature gives a reason to believe the message really came from the sender and that it is not modified. It is like a real-life signature. Everyone who knows Alice can verify messages with a signature generated by her. This also results in the fact that Alice cannot deny having sent that message to Bob, i.e. Bob can prove to other parties (who know Alice) that the message has a valid signature from Alice.

There are three things a digital signature typically consists of:

- A key generation algorithm, which selects a private key from the set of all private keys. It also produces a corresponding public key.
- A signing algorithm, that produces a signature using the message and the private key.
- A signature verifying algorithm, which is able to check whether the one who claimed to have sent the message is really the sender, or not, and is able to prove that the message is not modified given the message, public key and signature. Formally it checks whether a signature was generated with a secret key matching the given public key.

2.2 Post-quantum signature schemes

The story of cryptography goes back over 2500 years [6]. The biggest part of that time, authentication was done through a physical signature on the (encrypted) message. An example of such a signature is a wax seal. When cryptosystems got digitalized, and the popularity of public-key encryption had increased, authentication could no longer go through

a physical signature. Diffie and Hellman tackled this problem in their revolutionary paper [5] in 1976. They thought of a way to implement such a digital signature using the key exchange and the abstract construction of their encryption scheme. They based their encryption and signature scheme both on computational one-way functions with trapdoors but they didn't have such function. Instead they had a computational one-way function (discrete log) which was good enough to get key exchange. Soon after that, Rivest, Shamir and Adleman came up with a real signature scheme in 1978 [16], called the RSA signature scheme. Many signature schemes came to rise. The security of these signature schemes is based on solving all kinds of computationally hard mathematical problems. The most used signature schemes nowadays are the RSA signature scheme, the Digital Signature Algorithm (DSA) and the Elliptic Curve Digital Signature Algorithm (ECDSA). Also computers had made fast development. In the early 80's there was a thought experiment involving a quantum computer. A quantum computer is a computer built to make use of quantum mechanical effects in its computations [2]. Many attempts to built such a computer of significant size have been done, but it has not yet succeeded. Quantum computers are known to solve some computationally hard mathematical problems, like factoring integers, a lot times faster than classic computers, because algorithms for these problems can be executed much faster than on classical computers. In cryptography this fact was ignored for quite a while. In the early 2000s, Bernstein coined the term post-quantum cryptography and soon after that the focus also shifted quickly to include *post-quantum signature schemes*: signature schemes that are likely to be resistant against quantum computer attacks. Note that it only concerns asymmetric cryptography as most symmetric cryptography is less affected by the quantum computer development.

Fortunately there are still a couple of mathematical problems that are not efficiently tractable by a quantum computer. Examples are CVP, the Closest Vector Problem, and everything related to it, the decoding problem and finding the solution of multi-variate quadratic systems. Based on these kinds of problems, there are four types of post-quantum signature schemes:

1. Multivariate public-key signature schemes.
2. Code-based signature schemes.
3. Hash-based signature schemes.
4. Lattice-based signature schemes.

In section 4 several examples of lattice-based signature schemes are presented.

2.3 Kleptography

Cryptology is the science of secure communication and contains both of the areas cryptography and cryptanalysis. Cryptography is the study of designing and developing cryptosystems which results in secure communication. One can perform an analysis on the security of a certain cryptosystem. This analysis is called *cryptanalysis*. Such an analysis usually

contains attacks in order to break the cryptosystem. An attack is designed especially for the weak spot of the cryptosystem. These attacks exploit the weaknesses of the system. If such an attack easily takes 2^{128} operations, it would not be a problem. It is, however, an attempt to retrieve the plaintext without knowing the secret key or knowing the secret key from public information and with sufficiently small effort, the cryptosystem is considered broken. The cryptosystem is not secure anymore.

2.3.1 Attacks

A weakness of digital signature schemes is when an unauthorized person gains access to Alice's signing private key SK, this person can send messages with a signature from Alice. Then Alice has to revoke her old key and must generate a new private key to be able to authenticate herself to others, because with the old keys she is unable to repudiate the false messages. The fake signatures do not have any value when everyone who knows Alice gets the new keys.

The security of some digital signature scheme is as strong as the weakest part. Some schemes have general weaknesses. Such weaknesses can be found in the algorithms or in the implementation of the scheme. Many systems use randomness and therefore they use a random number generator. Such a random number generator is sometimes not entirely random as some random number generators are depending on several algorithms. So this randomness could be predicted by e.g. keeping statistics. An attacker can exploit this randomness and find a leak in it. Such a leak is considered a general weakness of the system itself. This is, because the weakness is exploitable from the outside. It is also possible for everyone to use this attack. Attackers find a way to prove that information itself is leaking. Then the cryptosystem is considered broken. This is basically conducting a cryptanalysis.

That is not what kleptography of signature schemes is about. Kleptopgraphy does not want to break the system, but instead keep universal protection and provide a private backdoor, only to be used by the attacker. Kleptography operates from the inside. It modifies the set up of the scheme in such a way that it is unnoticeable to the user and the leaked information is only understandable for the attacker. Young and Yung actually warned everybody in their paper [17] about these kleptographic backdoors in already existing black-box systems. A black-box system is one where only input and output are accessible to the user. The implementation details are completely unknown. All systems used by Young and Yung are based on pre-quantum cryptosystems. So when quantum computers are completely developed, these systems will be broken. Now, this thesis wants to show how to design a kleptographic backdoor in post-quantum signature schemes.

2.3.2 SETUP

Kleptography is the main focus of this thesis. It is the study of modifying a cryptosystem in such a way that it leaks secret information exclusively to the third party and the user cannot notice this subliminal channel. The cryptosystem itself is in general assumed secure.

The goal of kleptography is to build a cryptographic Trojan horse, called a backdoor, that is robust against reverse-engineering and only useful to the attacker. It will keep global security while providing the attacker exclusive use of insecurity. There are three desired properties of a kleptographic backdoor [17]:

1. *Exclusivity*: only the attacker gets the secret information and the attacker cannot be caught doing this.

2. *Indistinguishability*: the backdoor cannot be detected in a black-box system, except by the attacker.

3. *Forward secrecy*: if one would be able to reverse-engineer the black-box, then the leaked information remains confidential, e.g. nobody can recover the previously leaked information.

Young and Yung are the inventors of this concept. In 1996, they published their first article [17] about the dangers of black box cryptography. They introduced a so-called SETUP mechanism, which stands for Secretly Embedded Trapdoor with Universal Protection. Young and Yung later coined the term kleptography to denote such attacks.

The most simple attacks on a cryptosystem are in the form of tampering or predicting the pseudorandom generator or using a weak version of the algorithm. As mentioned before, these are ways to exploit general weaknesses of the system. Then signatures are not necessarily needed for leaking information. SETUP, on the other hand, is a lot more complex to implement, because it has many requirements. The formal definition of SETUP is given by [17]:

Definition 2.1. *Let C be a publicly known cryptosystem. A SETUP mechanism is an algorithmic modification made to C to get C' such that the following hold.*

1. *The input of C' agrees with the public specifications of the input of C.*

2. *C' computes using the attackers public encryption function E contained within C'.*

3. *The attacker's private decryption function D is not contained within C' and is known only by the attacker.*

4. *The output C' agrees with the public specifications about the output of C. At the same time, it contains published bits of the user's secret key, which are easily derivable by the attacker but otherwise hidden.*

5. *The output of C and C' are polynomially indistinguishable to everyone, except the attacker.*

6. *After discovery of the specifics of the SETUP algorithm and after discovering its presence in the implementation (by e.g. reverse-engineering), users (except the attacker) cannot recover past keys.*

It is hard to hide a backdoor if the code can be inspected, but for the definition of SETUP it is really important that the extracted information from the backdoor cannot be recovered by inspecting the code and reverse-engineering the backdoor.

Next to regular SETUP, described in definition 2.1, there is also weak SETUP where requirement 5 does not have to be satisfied. This type of SETUP works just fine, because the user of C' probably will not notice this discrepancy as the user is not aware of using a modified program and does not know what the output should be.

Young and Yung made several SETUP mechanisms in commonly used cryptosystems. For a detailed overview of these mechanisms, see [1]. In these cryptosystems the message could be leaked as well. They also made one in the ElGamal signature scheme. The mechanism needs two signatures to recover the private key. In figure 1, there is a schematic view of Alice and Bob using a modified version of a cryptosystem C.

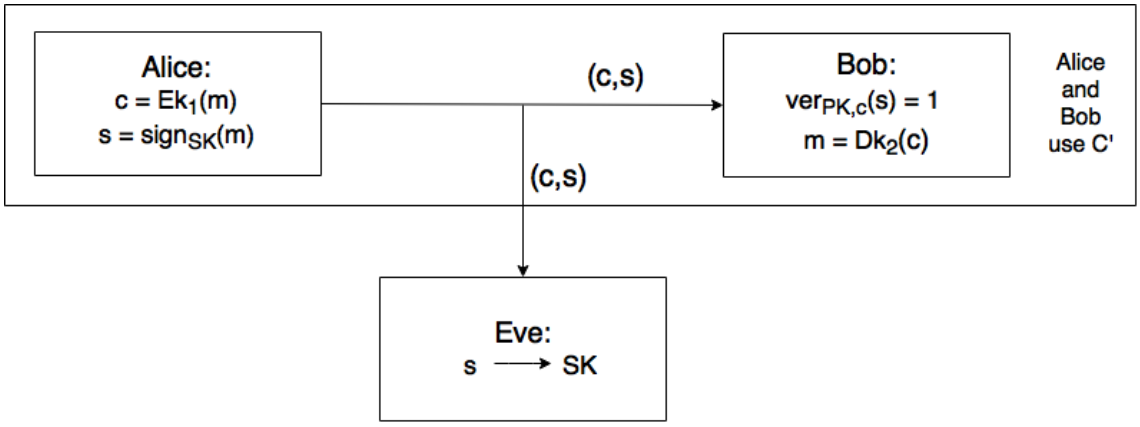

Figure 1: Back door

2.3.3 Subliminal Channel

Both of the kleptographic backdoors described in chapter 5 are applicable for transferring a secret message hidden in a signature generated in a secret way. Of course a backdoor can be used for the attacker to exfiltrate secret information. Then the user does not notice that the information is leaking to a third party, and neither does the receiver. There is also another way to use this backdoor. This is called setting up a *subliminal channel* [1]. Imagine the following situation: Alice and Bob are in prison and they are separated from each other. But the warden of the prison, Eve, is willing to pass on messages from Alice to Bob and vice versa, only if Eve is permitted to read the messages. In order to still have some secret communication, Alice and Bob can set up a subliminal channel in this communication. For this, Alice and Bob have to agree on a certain key y. Then, whenever Alice sends a message including a signature (generated by the modified signature scheme using y) to Bob, Eve can read the message, authenticate the signature and conclude that nothing is wrong with the message. Now, Bob can decrypt the secret message hidden in

the signature using y. This way Alice and Bob have an exchange of messages without the third party knowing.

2.4 Applications

Digital signature schemes are used worldwide for many applications. For the remainder of this section the focus is on two of the most well-known applications.

2.4.1 Bitcoin

Bitcoin is an alternative payment system [3]. It is one of the first widely used implementations of the concept *cryptocurrency*, i.e. cryptographic money. There is no bank or any trusted third party. The system solely relies on cryptography. The user uses digital signatures for transactions. The signature schemes provide authentication of users and integrity of transactions. Bitcoin uses cryptosystems that rely on the discrete logarithm problem in elliptic curves.

2.4.2 PGP

PGP stands for Pretty Good Privacy, and it is a program that encrypts and decrypts data, for example an e-mail. Everyone can use this program to send e-mails in a confidential way. Using signatures, it offers authenticity. In the most commonly used variant, the security relies on the RSA cryptosystem. When signing the encrypted document, the signature is sent along with it. The receiver is then able to verify this signature with the public key of the sender. This offers integrity as the signature proves that the message is not changed in transmission. It is also possible to just encrypt or just sign a document.

2.5 Summary

A cryptosystem provides a way for secure communication. To make sure that the concerned parties are actually communicating with each other, they can authenticate themselves using a signature scheme. Post-quantum signature schemes provide a secure way to authenticate, even when quantum computers are involved. Kleptography, however, studies modifications to schemes done in such a way that secret information leaks to the attacker from a backdoored implementation.

3 Background information

In the next section some number theory is explained in order to provide the reader with sufficient mathematical background. After that some technical terms often used in cryptography are described.

3.1 Mathematics

Number theory is the study of integers. The theorems and lemmas treated in this field are often used in cryptography. The set of all integers is denoted by $\mathbb{Z}$. The set of non-negative integer numbers is denoted by $\mathbb{N}$. The set of all rational numbers, i.e. the set of all numbers which can be expressed as the quotient $\frac{p}{q}$ with p and q integers and q non-zero, is denoted by $\mathbb{Q}$. The set of all real numbers is denoted by $\mathbb{R}$.

3.1.1 Primes

Cryptosystems often use primes, because of their special properties. Prime numbers in $\mathbb{N}$ are numbers whose only positive divisors are 1 and themselves. A divisor of a number a is another number b, which can divide a into an integer amount without remainder. So 4 is a divisor of 12, because $\frac{12}{4} = 3 \in \mathbb{N}$ but 8 is not a divisor of 12, because $\frac{12}{8} = \frac{3}{2} \notin \mathbb{N}$. Let gcd($a$,$b$) be the greatest common divisor of integer a and b. If gcd(a,b) = 1, then a and b are called co-prime or relatively prime. A great property of primes, and this property is also called the fundamental theorem of number theory, is the following theorem:

Theorem 3.1. *Let $a \in \mathbb{N}\backslash\{0\}$, then a has exactly one prime factorization $a = p_1^{e_1} p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}$, with p_i prime such that $p_1 < p_2 < \ldots < p_k$ and e_i positive integers*

This theorem and the proof can be read in section 2.2.2 in [1].

Let a be as in Theorem 3.1. Then the Euler Phi function $\phi(a)$ is defined as $\phi(a) = \phi(p_1^{e_1} p_2^{e_2} \cdot \ldots \cdot p_k^{e_k}) = p_1^{e_1-1}(p_1 - 1) \cdot \ldots \cdot p_k^{e_k-1}(p_k - 1)$. Note that for any $n = pq$, with p, q prime, it holds that $\phi(n) = (p-1)(q-1)$. By definition, $\phi(0) = 1$ and $\phi(1) = 1$.

3.1.2 Modulus

When pointing out the time, the hour is usually a number between 0 and 23, although much more time has past at that moment. This is an example of computing with a modulus. In this example the hour is computed modulo 24, meaning only the remainder is considered after dividing by 24. Now, when there is an integer number q for which the following holds: $a - qn = b$, then $a \equiv b \mod n$ and a and b are called congruent modulo n.

If for $a, b \in \mathbb{Z}$ it holds that $a \cdot b \equiv 1 \mod n$, then b is the modular inverse of a modulo n. Such an a has exactly one inverse in $\{0, 1, \ldots, n-1\}$ if gcd(a, n)=1, otherwise the inverse does not exist.

3.1.3 Rings and fields

The following properties are needed in order to provide formal definitions of a ring and a field. Let S be a set with elements and $a, b, c \in S$. Let $\oplus$ be an operation on S. Operation $\oplus$ is a function that maps elements of $S \times S$ to a value in S, e.g. $\oplus(a, b) = a \oplus b$, i.e. S is closed under $\oplus$. Then:

1. Associativity on $\oplus$ implies $a \oplus (b \oplus c) = (a \oplus b) \oplus c$.
2. Commutativity on $\oplus$ implies $a \oplus b = b \oplus a$.
3. S is closed under $\oplus$ if for all $a, b \in S$ it holds that $a \oplus b \in S$.
4. There exist a neutral element, i.e. a neutral element of a under $\oplus$ is an element o such that $a \oplus o = o \oplus a = a$.
5. An inverse element of a under $\oplus$ is an element c such that $a \oplus c = c \oplus a = o$, with o being the neutral element under $\oplus$.

In a ring the operations involved are addition $+$ and multiplication $\cdot$. The formal definition [4] of a ring is a set of elements R for which the following properties hold for every $x, y, z \in R$:

- Associativity holds for both operations $+$ and $\cdot$ (property 1).
- Commutativity holds for operation $+$ (property 2).
- Left-distributivity ($x \cdot (y+z) = x \cdot y + x \cdot z$) and right-distributivity ($(x+y) \cdot z = x \cdot z + y \cdot z$) hold.
- R has a neutral element for both operations $+$ and $\cdot$ for all elements (property 4.
- R has an inverse element for operation $+$ for all elements (property 5).
- R is closed under both operations $+$ and $\cdot$ (property 3).

The formal definition [4] of a field F is a ring R for which the following properties hold for every $g, h \in F$:

- Commutativity holds for operation $\cdot$ (property 2).
- F has an inverse element for operation $\cdot$ for all elements except for 0 (property 5), where 0 is the neutral element for operation $+$.

Note that the following property of a field F results from property 2 and 5: there are no zero divisors except 0, e.g. $g \cdot h = 0$ implies $g = 0$ or $h = 0$. If R is a ring, then R^* denotes the multiplicative group of R, i.e. the set of all multiplicatively invertible elements of R. Also $\mathbb{Z}_q^n$ is the ring of the integer vectors in dimension n modulo q, e.g. each element is taken modulo q.

3.1.4 Polynomials

A polynomial is a function that is a sum of products of a coefficient and variables with a non-negative exponent. In this thesis only polynomials in one variable are considered. These have the form $f(X) = \sum_{i=0}^{n-1} f_i X^i$, in which $f_i \in R$ for some ring R are called the coefficients of f and X is the variable. This polynomial has degree $n-1$ if $f_{n-1} \neq 0$. Polynomials in $\mathbb{Z}[X]$ are functions with coefficients in $\mathbb{Z}$.

In this thesis, polynomials are generated from a set $R[X]$ or $\mathbb{Z}[X]$. When reducing their coefficients modulo q, then this ring is denoted by $R_q[X]$. When the polynomials are reduced modulo $X^n - 1$, then this ring is denoted by $R[X]/(X^n - 1)$. A polynomial reduction is dividing a polynomial $f(X)$ by another polynomial $g(X)$. The result is $f(X) = h(X) + l(X)g(X)$, with $h(X)$ being the remainder after the polynomial reduction.

Let $f, g \in R = \mathbb{Z}[X]/(X^n - 1)$. This is the ring of polynomials with integer coefficients and with a reduction modulo $X^n - 1$. Then the cyclic convolution product of f and g, resulting in h, is defined as

$$h = f \circledast g = \sum_{k=0}^{n-1} \sum_{i+j \equiv k \mod n} (f_i \cdot g_j) \cdot X^k. \tag{5}$$

This is similar to the normal product of two polynomials, except h should end up in $\mathbb{Z}[X]/(X^n - 1)$ after the multiplication. Therefore h has to be reduced modulo $X^n - 1$. This is the same as substituting 1 for X^n, which results in the product given by equation (5). The centered norm of polynomial $f \in \mathcal{F}_f \subset \mathbb{Z}[X]/(X^n - 1)$ with degree $n-1$ is defined as $\|f\|_c^2 = \sum_{i=0}^{n-1} f_i^2 - \frac{1}{n}(\sum_{i=0}^{n-1} f_i)^2 = \sum_{i=0}^{n-1}(f_i - \mu_f)^2$, with $\mu_f = \frac{1}{n}\sum_{i=1}^{n-1} f_i$. The maximum norm of f is $\|f\|_\infty$, defined as $\|f\|_\infty = \max_i |f_i|$.

3.2 Matrices and vectors

An $n \times m$ matrix A is a rectangle block of size n by m with numbers, called entries, in it as a representation of n row vectors or m column vectors. The entry on the i-th row and the j-th column is denoted as $a_{i,j}$. For example if $A = \left(\begin{smallmatrix} 1 & 2 \\ 3 & 4 \end{smallmatrix}\right)$, then $\left(\begin{smallmatrix} 1 & 2 \end{smallmatrix}\right)$ is the first row vector, $\left(\begin{smallmatrix} 2 \\ 4 \end{smallmatrix}\right)$ is the second column vector and $a_{2,1} = 3$. In this thesis, a vector is always a row vector, unless mentioned otherwise. Vectors are underlined, e.g. $\underline{a}$. The determinant of a matrix A is denoted by $\det(A)$. The determinant can only be computed for square matrices. For a vector $\underline{a} \in \mathbb{R}^n$, $\|\underline{a}\| = \|\underline{a}\|_2$ means the Euclidean length, e.g. $\|\underline{a}\|_2 = \sqrt{a_1^2 + a_2^2 + ... + a_n^2}$ for $\underline{a} = (a_1, ..., a_n) \in \mathbb{R}^n$.

3.2.1 Special Matrices

The $n \times m$ null matrix O is the matrix in which all the entries are zero. The unity matrix, denoted by I_n is the $n \times n$ matrix such that for any $n \times n$ matrix A it holds

that $AI = IA = A$. The inverse of the $n \times n$ matrix A, denoted by A^{-1}, is such that $AA^{-1} = A^{-1}A = I_n$. This matrix does not exist for every matrix.

The transposed matrix of $n \times m$ matrix A, denoted by A^T, is the matrix where the i-th row vector of A is the i-th column vector of A^T.

3.2.2 Linearly independent

Vectors $\underline{a}_1, \underline{a}_2, ..., \underline{a}_n$ are linearly independent if and only if $\lambda_1\underline{a}_1 + \lambda_2\underline{a}_2 + ... + \lambda_n\underline{a}_n = 0$ only holds for $\lambda_i = 0\ \forall i \in \{1, ..., n\}$. Matrices are used for solving these kind of equations. When dealing with solving $Ax = b$, for some known matrix A and vector b and unknown vector x, x is the only unique solution to $(A|b)$ if after row reduction there are the same number of non-zero rows as there are variables and the matrix is not contradictory. There is also a specific kind of matrices, called unimodular matrices, often denoted as U. Such a matrix has integer entries and has $\det(U) = \pm 1$.

When there is a set of linearly *independent* vectors, it spans a certain space and it serves as the basis of that space. The number of vectors in the basis is called the dimension of the space.

When there is a set of linearly *dependent* vectors, they can be put into a matrix. Then the Gaussian elimination algorithm can be applied to the matrix. The result is the set of linearly independent vectors that serves as basis of the subspace they generate.

The inner product of two vectors $\underline{a} = (a_1, a_2, ..., a_n)$ and $\underline{b} = (b_1, b_2, ..., b_n)$ is defined as $\langle \underline{a}, \underline{b} \rangle = a_1b_1 + ... + a_nb_n$.

3.2.3 Gram-Schmidt Orthogonalization

The Gram-Schmidt Orthogonalization is an algorithm that results in an orthogonal basis $B^* = (\underline{b}_1^*, ..., \underline{b}_n^*)$ from a random $n \times n$ basis $B = (\underline{b}_1, ..., \underline{b}_n)$, in which the vectors are perpendicular to each other. In matrix notation B, the matrix of basis vectors, can be expressed in the product of B^*, the matrix of orthogonal basis vectors, and G, where $G = \begin{pmatrix} 1 & \mu_{2,1} & \cdots & \mu_{n,1} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \mu_{n,n-1} \\ 0 & \dots & 0 & 1 \end{pmatrix}$ and $\mu_{i,j} = \frac{<\underline{b}_i, \underline{b}_j^*>}{\|\underline{b}_j^*\|_2^2}$. Because of the divisions in μ, the resulting matrix is usually over Q, even if the input was a matrix over Z.

3.3 Lattices

Lattices are discrete subgroups of $\mathbb{R}^m$. Their use for cryptography is divided in two main purposes:

- Lattices are the base of several signature schemes, like GGH and NTRU. These will be described in detail later on.

- Lattices are used for cryptanalyses. Many attacks on cryptosystems are based on lattices.

3.3.1 Definition

The formal definition of a lattice is as follows:

Definition 3.1. *Let $\{\underline{b}_1, ..., \underline{b}_n\}$ be a linearly independent set of row vectors in $\mathbb{R}^m$ ($n \leq m$). The lattice generated by $\{\underline{b}_1, ..., \underline{b}_n\}$ is the set $L = \left\{ \sum_{i=1}^{n} \ell_i \underline{b}_i \middle| \ell_i \in \mathbb{Z} \right\}$ of integer linear combinations of $\underline{b}_i$.*

The vectors $\{\underline{b}_1, ..., \underline{b}_n\}$ are called a lattice basis. Every lattice has a basis. The lattice rank is n and the dimension of the space is m. If $n = m$, then L is a full rank lattice. A basis matrix B of a lattice L is an $n \times m$ matrix such that $L = \{\underline{x}B | \underline{x} \in \mathbb{Z}^n\}$. Note that $B_{i,j} \in B$ is the j-th entry of the basis vector $\underline{b}_i$. Specifically for lattices the words points and vectors are used for the same thing: an element of the lattice.

3.3.2 Important remarks

The following definitions, lemmas and remarks are used in research later in this thesis.

Lemma 3.2. *Two $n \times m$ matrices B and B' generate the same lattice L if and only if B and B' are related by a unimodular matrix. i.e. $B' = UB$.*

Remark 3.1. *Let $\underline{a} \in \mathbb{Q}^m$. Then the notation $\lfloor a \rceil$ is used to denote rounding off the entries in the vector to the nearest integer.*

For example: $\lfloor (\pi, 1/2, -5/8) \rceil = (3, 1, -1)$

Remark 3.2. *The parallelepiped of a basis $B = \{\underline{b}_1, ..., \underline{b}_n\}$ is defined as*
$P(B) = \left\{ \sum_{i=1}^{n} \alpha_i \underline{b}_i \middle| -\frac{1}{2} \leq \alpha_i \leq \frac{1}{2} \right\}$.

In figure 2 the grey area represents an example of a paralellepiped. A good basis results in a rather orthogonal parallelepiped and a bad basis results in a skinny parallelepiped.

Definition 3.2. *Let L in $\mathbb{R}^m$ be a lattice of rank n. The successive minima of L are $\lambda_1, ..., \lambda_n \in \mathbb{R}$ such that, for $1 \leq i \leq n$, λ_i is minimal, so that there exist i linearly independent vectors $\underline{v}_1, ..., \underline{v}_i \in L$ with $\|\underline{v}_j\| \leq \lambda_i$ for $1 \leq j \leq i$.*
So $\lambda_i = \min\{\max\{\|\underline{x}_1\|, ..., \|\underline{x}_i\| | \underline{x}_1, ..., \underline{x}_i \in L \text{ are linearly independent}\}\}$.

It follows that $0 < \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$.
The lemmas and their proofs can be read in Chapter 16 and 17 of [8].

3.3.3 Computational problems in lattices

A couple of problems in lattices are for example checking if a point is in a certain lattice, or finding a basis for a lattice. These problems can be solved efficiently by using linear algebra. However, there are also some problems which cannot be solved that easily [8].

Let L be a lattice in $\mathbb{Z}^m$. There are six problems that are considered hard to compute:

- The closest vector problem (CVP): given a basis matrix B for L and a vector $\underline{w} \in \mathbb{R}^m$, compute $\underline{v} \in L$ such that $\|\underline{w} - \underline{v}\|$ is minimal.
- The shortest vector problem (SVP): given a basis matrix B for L, compute a non-zero vector $\underline{v} \in L$ such that $\|\underline{v}\|$ is minimal.
- The decision closest vector problem: given a basis matrix B for L, a vector $\underline{w} \in \mathbb{R}^m$ and a real number $\tau > 0$, decide whether or not there exists a vector $\underline{v} \in L$ such that $\|\underline{w} - \underline{v}\| \leq \tau$.
- The decision shortest vector problem: given a basis matrix B for L and a real number $\tau > 0$, decide whether or not there exists a non-zero vector $\underline{v} \in L$ such that $\|\underline{v}\| \leq \tau$.
- The approximate closest vector problem: Let $\tau > 1$. Given a basis matrix B for L and a vector $\underline{w} \in \mathbb{R}^m$, compute a vector $\underline{v} \in L$ such that $\|\underline{w} - \underline{v}\| \leq \tau\|\underline{w} - \underline{x}B\|$ for all $\underline{x} \in \mathbb{Z}^n$.
- The approximate shortest vector problem: Let $\tau > 1$. Given a basis matrix B for L, compute a non-zero vector $\underline{v} \in L$ such that $\|\underline{v}\| \leq \tau\lambda_1$.

3.3.4 Lattice basis reduction

With lattice problems, there are good and bad bases. A good basis means that the problems are relatively easy to solve, whereas a bad basis means that the problems are basically intractable. In this section two figures are shown as examples of a good and bad basis in dimension 2. With lattice basis reduction it is possible to transform a given (bad) lattice basis into a good basis with vectors that are preferably short and close to orthogonal. This requires a good definition of a good basis and an efficient way to compute it, e.g. by an algorithm. For lattices in dimension two, we have the algorithm of Lagrange and Gauss [8]. For lattices with higher dimensions we have the algorithm of Lenstra, Lenstra and Lovász, the LLL algorithm [8]. Basically the LLL algorithm generalizes the Lagrange-Gauss algorithm. The steps of both of the algorithms are not explained in detail here, but can be found in the mentioned book.

For a good basis, the goal for the vectors in this basis is to be as short as possible, so in this case the vectors should be equal to the successive minima. In dimension 2, this goal is achieved by using the Lagrange-Gauss algorithm. Let B be a basis for lattice L, with $\underline{b}_1, \underline{b}_2$ as row vectors. The basis $\{\underline{b}_1,\underline{b}_2\}$ is Lagrange-Gauss reduced if $\|\underline{b}_1\| \leq \|\underline{b}_2\| \leq \|\underline{b}_2 + k\underline{b}_1\|$ for all $k \in \mathbb{Z}$. A result of this reduced basis is that the norm on both of the row vectors

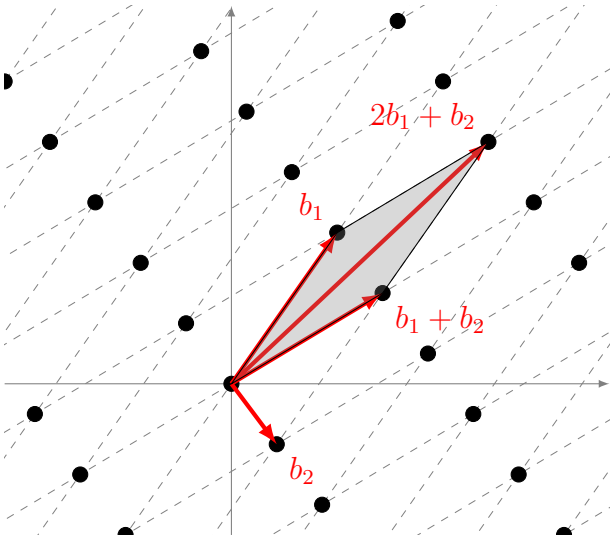

Figure 2: Example of a bad basis

matches the successive minima, i.e. $\|\underline{b}_i\| = \lambda_i$ for $i = 1, 2$ [8]. The Lagrange-Gauss algorithm can be used to achieve this, because for any basis of L the algorithm terminates and provides the reduced basis as output. However, this gets harder when working in large dimensions. The retrieved matrices from this algorithm are still very bad and this is why cryptosystems can rely on the computationally hard problems. Another property of a good basis is orthogonality. The vectors should be orthogonal, or as close to orthogonal as possible. For any set of vectors, the Gram-Schmidt orthogonalization process results in orthogonal vectors. However, this transformation does not use integer coefficients and therefore does not produce lattice vectors. So when rounding the entries off to the nearest integer, for example, the lattice loses orthogonality.

Let $\{\underline{b}_1, ..., \underline{b}_n\}$ be a basis for lattice L and $\{\underline{b}_1^*, ..., \underline{b}_n^*\}$ an orthogonal basis of L, being the output of the Gram-Schmidt algorithm. Recall that $\underline{b}_i^* = \underline{b}_i - \sum_{j=1}^{i-1} \mu_{i,j} \underline{b}_j^*$, where $\mu_{i,j} = \frac{<\underline{b}_i, \underline{b}_j^*>}{\|\underline{b}_j^*\|^2}$. Then the basis $B = \{\underline{b}_1, ..., \underline{b}_n\}$ is called size reduced if $|\mu_{i,j}| \leq \frac{1}{2}$. This means that all $\mu_{i,j}$ will be rounded to 0. A basis B is called LLL-reduced with parameter δ if it is size reduced and for $i = 1, 2, ..., n-1$ the Lovásc condition $\|\underline{b}_{i+1} + \mu_{i+1}\underline{b}_i\|^2 \geq \delta \|\underline{b}_i\|^2$ holds. The LLL algorithm runs the Gram-Schmidt algorithm with rounding and just swaps two consecutive basis vectors when the Lovásc condition does not hold.

In theory the reduced bases retrieved by the algorithms have been very useful for computing the computationally hard problems mentioned before. For finding lattice vectors of short length (SVP), just take the reduced basis vectors. For finding lattice vectors close to any other vector v outside the lattice, there is an algorithm, called Babai's rounding technique, to compute a lattice point close to v. Babai's rounding technique is as follows: let B be the reduced basis of a lattice, then $B\lfloor B^{-1}v \rceil$ is a lattice point close to v. This algorithm gives better results when B is close to orthogonal. Unfortunately, the reduced

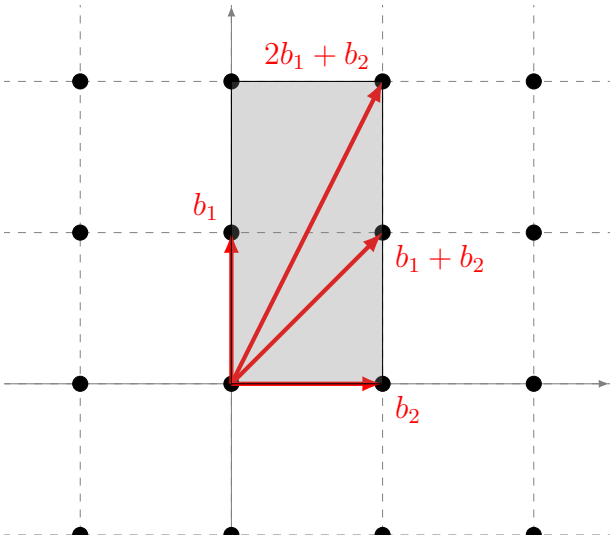

Figure 3: Example of a good basis

version of the basis does not result in the optimal solution for the computationally hard problems.

3.4 Tools for cryptography

Two kinds of functions are frequently used in cryptography. Below, the formal definitions are given.

3.4.1 Cryptographic hash functions

One type of function is a hash function. A hash function is a function that maps (binary) data of any size to a fixed size value, often called the hash value. It has the special property that the hash value of a certain input is easily evaluated, but it is hard to invert given a random image, where the terms easy and hard are used here in the sense of computational complexity. Now, a cryptographic hash function is given as the following definition [8]:

Definition 3.3. *Let $H : \{0,1\}^* \to \{0,1\}^k$ denote a hash function. H is called a cryptographic hash function if the following holds:*

- *Given hash value $y \in \{0,1\}^k$ of H, it is computationally infeasible to find a preimage $x \in \{0,1\}^*$ such that $H(x) = y$.*
- *Given a preimage $x \in \{0,1\}^*$, it is computationally infeasible to find a second preimage $x' \in \{0,1\}^*$ with $x' \neq x$, such that $H(x) = H(x')$.*
- *It is computationally infeasible to find a collision, i.e. a pair, (x, x') with $x, x' \in \{0,1\}^*$ and $x \neq x'$, such that $H(x) = H(x')$.*

Their one-way property is useful for having relatively short fingerprints for data.

3.4.2 Trapdoor one-way function

Trapdoor one-way functions are like public-key cryptosystems. One can easily compute the value of the output (in case of a public-key cryptosystem: compute the ciphertext from a message), and with certain information one can compute the inverse (in case of the public-key cryptosystem: with the secret key the ciphertext can be decrypted back to the message). The formal definition is given by:

Definition 3.4. *Let $f : A \to B$ be an function, so $f(a) \in B$ for all $a \in A$. Let f^{-1} be the inverse of this function on $f(A)$, such that $f^{-1}(b) \in A$ for all $b \in f(A)$. Then f is a trapdoor one-way function if the following holds:*

- *$f(a)$ is easy to compute for all $a \in A$.*
- *$f^{-1}(b)$ is difficult to compute for almost all $b \in f(A)$.*
- *$f^{-1}(b)$ is easy to compute for all $b \in f(A)$ given some information t.*

3.5 Summary

The nice properties of lattices are used in lattice-based cryptosystems. Because there are still computationally hard problems concerning lattices, the security of some cryptosystems is relying on (one of) them. This holds also for signature schemes in particular.

4 Lattice-based Signature schemes

Lattice-based signature schemes are based on the hardness of solving lattice problems. They are fairly new in comparison with the other schemes. Also, there is still active research in lattices and their properties. Unfortunately most of the older schemes are broken. Below, ideas of several schemes are given. Note that there are many other versions of the schemes, as the schemes below are just simplified schoolbook versions.

4.1 GGH Signature Scheme

Goldreich, Goldwasser and Halevi proposed in 1996 [10] a new trapdoor one-way function, from which they derived a public-key encryption and digital signature scheme. The signature scheme is called the GGH signature scheme. The security of their new public key encryption algorithm and digital signature scheme was assumed to rely on the computational difficulty of lattice reduction problems, in particular CVP.

4.1.1 The underlying idea

The underlying idea of the GGH signature scheme is that, given any basis for a lattice L, it is easy to generate a vector which is close to a lattice point, e.g. by taking this lattice point and adding a small error vector. The hardest part is to return from this new vector to its original given an arbitrary basis of the lattice. This can be seen as a one-way computation. In order to introduce a trapdoor mechanism into this one-way computation, which allows very good approximation of the closest lattice point problem, the basis B of L can be seen as the trapdoor information. This basis B is the good basis. Now there is another basis needed, for computing the function, but not inverting it. This bad basis B' could be derived from using a randomized uni-modular transformation. Let a lattice point be determined by an integral linear combination of the columns of B' and let τ be an error bound. To invert the value, one of Babai's nearest-vector approximation algorithms using B can be chosen to find a lattice point which is at most τ away from the given vector. The signature scheme generates signatures for messages, which are vectors of $\mathbb{R}^n$. The signature of the message is a lattice point which is close to the message (closeness is defined by τ). Verifying correctness is done by checking that a signature is indeed a lattice point and that the message is close to the signature. Messages which are close enough to each other will have the same signature. To prevent this from easy message modification attacks, the message could be hashed first before signing it.

4.1.2 The scheme

Key generation: The private key SK is the good basis B. B is used to find lattice points which are close to some given vectors in $\mathbb{R}^n$. The public key PK is a basis B', which is a bad basis for L, and the parameter τ. There are a couple of ways to make a basis bad. In this case a random uni-modular matrix U is used. Then $B' = UB$.

Creating signature s: hash message m and interpret this as a vector in $\mathbb{R}^n$. This is done by some hash function H. The result is called m', such that $m' = H(m)$ with $H : \{0,1\}^* \to \mathbb{R}^n$. Then s is defined by

$$s = \lfloor m'B^{-1} \rceil B. \tag{6}$$

Verification: Check that the signature s is indeed a lattice point in L by using public key B'. Check also if the Euclidean distance between s and m' is less than τ.

This works, because it is easy to compute a good approximation of CVP with a good basis but difficult to do so with a bad basis. So by choosing $s = \lfloor m'B^{-1} \rceil B$, s is a good approximation of the closest vector to m' such that $\|s - m'\| \leq \tau$. Then $s - m'$ is in the fundamental parallelepiped $P(L)$.

4.1.3 The weakness

Every time the user signs and sends the signature over to the other party, a bit of information is leaking about the secret key. Once sufficiently many signatures have been obtained, the paralellepiped of the lattice can be retrieved. The reason a signature is leaking this information is because of the fact that GGH signatures are not zero-knowledge [15]: for a given message, a certain set of valid signatures are possible, and the one selected by the secret key says something about the secret key itself. This information leakage does not necessarily prove that such schemes are insecure. Also note that this is a weakness of the scheme, and not a kleptographic backdoor.

So the attacker needs to recover the hidden paralellepiped. This problem is also known as the HPP (Hidden Paralellepiped Problem). A simple description of the problem is as follows: given points sampled uniformly from an n-dimensional centered parallelepiped, recover the parallelepiped. The attack is described by Nguyen and Regev [15] and follows, in short, the following steps:

1. Using the GGH signatures, the covariance matrix defined as $(B')^T B$, is approximated, in which $P(B')$ is the given parallelepiped and B' the lattice basis matrix.

2. This approximation is exploited such that the hidden paralellepiped $P(B')$ is reduced to a hypercube. This is because working with a hypercube is easier than working with a paralellepiped.

3. Now from that hypercube the actual paralellepiped is retrieved.

4. One can easily compute the basis from the parallepiped.

In dimension 2, the paralellepiped is clearly visisble, as shown below in figure 4.

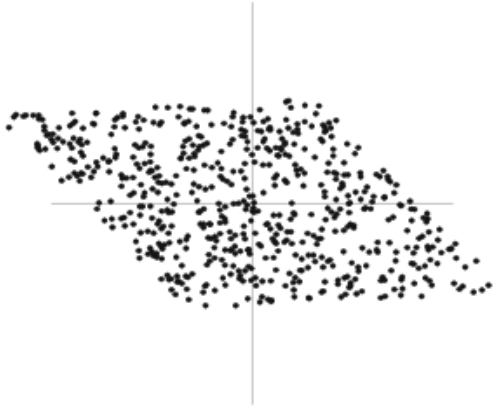

Figure 4: Hidden paralellepiped [15]

4.2 BLISS

BLISS [3] is the Bimodal Lattice Signature Scheme. It was published in 2013 by Ducas, Durmus, Lepoint, and Lyubashevsky. BLISS is considered provably secure and it has a good performance. Unlike GGH, this signature scheme does not rely on CVP. There are two other problems with lattices, namely the Short Integer Solution (SIS) and Learning With Errors (LWE) problems. Their hardness is related to the hardness of the other lattice problems.

4.2.1 The Gaussian distribution

The discrete Gaussian distribution D_σ^n is a distribution defined over the integers. The values of the distribution are visualized in figure 5. D_σ^n denotes the centered discrete Gaussian distribution for vectors, in which σ is the standard deviation and n is the desired dimension of the vector. Now, vectors can be sampled from this distribution. The choice of the parameters has to happen in a specific way in order to ensure the hardness of BLISS. Only integer values are sampled. For more details, read [7]. The reason a vector is sampled from this distribution, is to hide the secret key better against attackers.

4.2.2 SIS and LWE

SIS and LWE problems live in $\mathbb{Z}_q^n = \{\underline{x} | x_i \in \mathbb{Z}_q \forall i \in \{1, ..., n\}\}$. This space is a lattice itself, because every combination of vectors is in that space. Let L be a lattice in which $L = \{A\underline{x} \mod q | \underline{x} \in \mathbb{Z}^n\}$, spanned by a matrix $A \in \mathbb{Z}_q^{n \times n}$.

SIS concerns finding a short non-zero vector $\underline{x} \in \mathbb{Z}_q^n$, given matrix A, such that $A\underline{x} \equiv 0 \mod q$. Finding this linear combination such that the equation holds is considered hard.

LWE uses error vectors $\underline{e} \in \mathbb{Z}_q^n$ with a certain distribution D, e.g. the Discrete Gaussian Distribution. Given public matrix A, and public vector $\underline{b}$, for which the following holds:

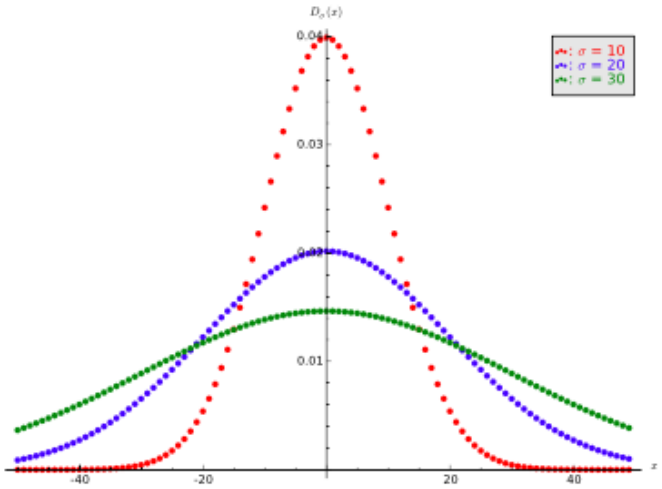

Figure 5: Discrete Gaussian Distribution [3]

$\underline{b}^T \equiv \underline{s}^T A + \underline{e}^T \in \mathbb{Z}_q^n$, find secret vector $s \in \mathbb{Z}_q^n$. Because the error vectors are unknown, this problem is considered hard.

BLISS depends on the hardness of both SIS and LWE. In section 4.2.4, a simplified version of BLISS is used from [3]. The real scheme is to be found in [7].

4.2.3 Lyubashevsky signature scheme

Lyubashevsky published a signature scheme in 2012 [14]. BLISS is actually an enhancement of this scheme. Below the scheme is described in a short overview to provide an idea of BLISS:

Key generation: The private key is a uniformly random $n_2 \times k$ matrix S with coefficients in $\{-d, ..., 0, ..., d\}$. Here, d is a chosen parameter. The public key consists of a uniformly random matrix $A \in \mathbb{Z}_q^{n_1 \times n_2}$ and matrix $T = AS$, computed modulo q.

Creating signature s: Generate $\underline{y} \in D_\sigma^{n_2}$, then $\underline{c} = H(A\underline{y} \| m)$, with m being the message and some hash function $H : \mathbb{Z}_q^{n_1} \times \{0,1\}^* \to \mathbb{Z}_q^k$. Now, $\underline{z} = S\underline{c} + \underline{y}$. Signature s is generated with $s = (\underline{z}, \underline{c})$, used with a certain probability.

Verification: to verify signature s, check if $H(A\underline{z} - T\underline{c} \| m) = \underline{c}$ and $\|\underline{z}\| \leq \tau\sigma\sqrt{n_2}$, for some chosen τ.

This works, because $H(A\underline{z} - T\underline{c} \| m) = H(A(S\underline{c} + \underline{y}) - T\underline{c} \| m) = H(AS\underline{c} + A\underline{y} - T\underline{c} \| m) = H(T\underline{c} + A\underline{y} - T\underline{c} \| m) = H(A\underline{y} \| m) = \underline{c}$

4.2.4 The scheme

The following algorithms describe the essential parts of BLISS:

The key generation: Generate prime q and dimension n. Divide n in two parts: $n = n_1 + n_2$, with $n_1 > n_2$. Generate a random sparse matrix $S' \in \mathbb{Z}_{2q}^{n_1 \times n_2}$ with coefficients

$S'_{i,j} \in \{-1, 0, 1\}$. Let $S = \binom{S'}{I_{n_2}} \in \mathbb{Z}_{2q}^{n \times n_2}$. Generate a random matrix $A' \in \mathbb{Z}_q^{n_2 \times n_1}$. Put $A = (2A'|qI_{n_2} - 2A'S') \in \mathbb{Z}_{2q}^{n_2 \times n}$. The secret key SK is S and the public key PK is A. Note that $AS \equiv qI_{n_2} \mod 2q$, resulting in $AS \equiv O \mod q$. Given that S' is sparse, this is basically solving a set of instances of the SIS problem as the columns of S are solutions to SIS on A.

Creating signature s: generate a vector $\underline{y} \in D_\sigma^n$. Now, compute $\underline{c} = H(A\underline{y} \mod 2q \| m)$. $\underline{c}$ should be a vector, consisting of $n_2 - k$ 0's and k 1's, in which $k \ll n$. Choose a random $b \in \{0, 1\}$. Set $\underline{z} = \underline{y} + (-1)^b S\underline{c}$. The whole signature procedure starts over with a certain probability. This was also seen in Lyubashevsky's signature scheme in section 4.2.3. The whole procedure starts over with a certain probability depending on z and c. This is, because of the fact that $\underline{z}$ is distributed according to $D_{\sigma, S\underline{c}}$, and so the more signatures are generated, the more information we get about the secret key. By throwing away some generated signatures changes the distribution of the signature pairs to hide S better. When the signature is not rejected, we have signature $s = (\underline{z}, \underline{c})$.

Verification: for a certain threshold τ, check if $\|\underline{z}\|_2 \leq \tau$ and $\|\underline{z}\|_\infty = \max\limits_{1 \leq i \leq n} |x_i| \leq q/4$, otherwise reject. Then accept when $\underline{c} = H(A\underline{z} + q\underline{c} \mod 2q \| m)$.

This works, because $A\underline{z} + q\underline{c} = A(\underline{y} + (-1)^b S\underline{c}) + q\underline{c} = A\underline{y} + (-1)^b q\underline{c} + q\underline{c} \equiv A\underline{y} \mod 2q$, using the definitions of A and S.

4.2.5 The Gaussian sample

In the signature generation, $\underline{y}$ is sampled from the discrete Gaussian distribution. As mentioned before, it is used to hide the secret key, in particular the secret relation $S\underline{c}$. By knowing y, the relation $\underline{z} - \underline{y} = (-1)^b S\underline{c}$ where only bit b and secret key S are unknown. Then by brute forcing, the secret key can be retrieved easily. That is why the sampling from the discrete Gaussian distribution is important.

4.3 Summary

Lattice-based signature schemes could be the solution to the coming post-quantum future. Therefore research in this area is necessary as this area is still very young.

5 NTRU Signature Schemes

In this section NTRUSign and NSS are described in detail. Also sketches of new kleptographic backdoors in both of the schemes are presented. At the end of this section a detailed analysis is given.

5.1 A brief history

The first signature scheme based on the hardness of lattice problems was the GGH signature scheme, described in section 4.1, published in 1996 [10]. Soon after that there were some flaws discovered in the scheme and the attention for lattice-based signature schemes disappeared a bit. Then, in 2002, NTRU published a signature scheme, called NTRUSign [12], which has a lot of resemblances with GGH. NTRUSign, however, was a very efficient scheme, because they used NTRU parameters. But what went wrong with both schemes, is that enough signatures leaked the form of the shape of the parallelepiped which in turn gave the secret key away, see section 4.1.3. In dimension $n = 502$, only 400 NTRUSign signatures were needed to retrieve the hidden paralellepiped. Because of the efficiency NTRUSign had, there was still interest in developing countermeasures to the attacks. Unfortunately this was not an immediate success. The most efficient known lattice-based digital signature scheme provably secure is BLISS [7], see section 4.2. NTRUSign had a predecessor, called NNS [13], introduced in 2001. The scheme got broken in 2006 by Nguyen and Regev in [15].

5.2 NTRUSign

NTRUSign is essentially a very efficient implementation of the GGH signature scheme. The signature length for the proposed parameters is only 1757 bits and signing and verification are faster than RSA-based methods. Still, NTRUSign had also some trouble with the flaw earlier discovered in GGH. The private key was recovered by analyzing signatures. For a better understanding of the scheme, the description of [9] and the original paper [11] is used for the following description.

The key generation: NTRUSign works with the following parameters (n, q, d_f, d_g, τ). The suggestions from [11] are $(n, q, d_f, d_g, \tau) = (251, 128, 73, 71, 300)$. The private key is a good basis for the NTRU lattice L, with short basis vectors. Start with generating short polynomials f and g in $\mathbb{Z}_q[X]/(X^n - 1)$, in which $\|f\|_c = \|g\|_c = O(\sqrt{n})$, f must be invertible in this ring and f, g have resp. d_f, d_g coefficients being 1 and the rest being 0. Then compute $h = f^{-1} \circledast g \mod q$. Let F and G be such that $f \circledast G - g \circledast F = q$, $\|F\|_c = \|G\|_c = O(n)$ and $F, G \in \mathbb{Z}_q[X]/(X^n - 1)$. Now the private key is good basis $B = \begin{pmatrix} M(f) & M(g) \\ M(F) & M(G) \end{pmatrix}$ and the public key is bad basis $B' = \begin{pmatrix} I_n & M(h) \\ 0 & qI_n \end{pmatrix}$ in which

$M(f) = \begin{pmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_{n-1} & f_0 & \cdots & f_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ f_1 & f_2 & \cdots & f_0 \end{pmatrix}$. Note that the rotations of pairs (f, g) and (F, G) form a basis for the lattice L, see section 5 of [11]. To show that these matrices represent both the same lattice L, B' is denoted as $\begin{pmatrix} 1 & h \\ 0 & q \end{pmatrix}$ and B is denoted as $\begin{pmatrix} f & g \\ F & G \end{pmatrix}$, for simplicity's sake. Then B' is obtained by the product of a unimodular matrix U and B, i.e. $UB' = B$, in which $U = \begin{pmatrix} f & (g - f \circledast h)/q \\ F & (G - F \circledast h)/q \end{pmatrix}$.

Creating signature s: message m is hashed to create a random vector (m_1, m_2) with $m_1, m_2 \in \mathbb{Z}_q[X]/(X^n - 1)$. Then the signer computes C and c using the following equations:

$$G \circledast m_1 - F \circledast m_2 = A + qC \tag{7}$$

and

$$-g \circledast m_1 + f \circledast m_2 = a + qc \tag{8}$$

where the coefficients of A and a are in $(-q/2, q/2]$. This is to be interpreted as taking the remainder of the left-hand side modulo q. Signature s is then defined as the following equation:

$$s \equiv f \circledast C + F \circledast c \mod q. \tag{9}$$

Verification is as follows: $t \equiv s \circledast h \mod q$ is computed by the receiver. Then two checks must be done in order to verify the signature:

1. check if (s, t) is indeed a lattice point in L.
2. check if (s, t) is close enough to (m_1, m_2), i.e. $\|s - m_1\|_c^2 + \|t - m_2\|_c^2 \leq \tau^2$, for some system parameter τ.

This works, because of the following: t can be written as $t \equiv s \circledast h \equiv (f \circledast C + F \circledast c) \circledast h \equiv g \circledast C + G \circledast c \mod q$. Because of this result and equation (9) the vector (s, t) can be written as $C \circledast (f, g) + c \circledast (G, H) \mod q$ and that is why (s, t) is a lattice point. For the second check, equations (7) and (8) are substituted in equation (9). The result is the following: $s \equiv f \circledast C + F \circledast c \equiv f \circledast \left(\frac{G \circledast m_1 - F \circledast m_2 - A}{q}\right) + F \circledast \left(\frac{-g \circledast m_1 + f \circledast m_2 - a}{q}\right) \equiv$
$m_1 \circledast (\frac{f \circledast G - F \circledast g}{q}) - \frac{A \circledast f + a \circledast F}{q} \mod q$.
Because F and G were chosen such that $f \circledast G - g \circledast F = q$, this results in the following equation for s:

$$s \equiv m_1 - \frac{A \circledast f + a \circledast F}{q}. \tag{10}$$

Then $\frac{\|A \circledast f + a \circledast F\|_c}{q}$ will be small enough, because $\|F\|_c = O(n)$, $\|f\|_c = O(\sqrt{n})$ and $\frac{\|a\|_c}{q}$, $\frac{\|A\|_c}{q}$ are small by construction. With the same analogy, equations (7) and (8) are substituted in $t \equiv g \circledast C + G \circledast c \mod q$, resulting in

$$t \equiv m_2 - \frac{A \circledast g + a \circledast G}{q} \tag{11}$$

and $\frac{\|A \circledast g + a \circledast G\|_c}{q}$ will also be small enough. The signature will likely pass. For details, the reader is referred to section 3 of [11].

5.3 Sketch of a kleptographic backdoor in NTRUSign

In this section, a sketch of a kleptographic backdoor in NTRUSign, described in section 5.2, is presented.

The main idea is to figure out how to select the signature in such a way that it does not stand out to the user while an encryption of a secondary message m' is leaked. Here, m' is public-key encrypted such that its content is exclusively available for the attacker. If executed right, the user should not notice that the program has been modified. This is done by fixing the randomness in the scheme for example. As there is hardly anything random in NTRUSign, slightly changing the generation of s might work. Ideally, this should not influence the number of invalid signatures or computation time.

In this sketch of a backdoor, s is being modified by adding a polynomial k. This polynomial is chosen such that the bits of s could represent (a part of) the encryption of the secret key. This way the attacker can leak the secret information necessary to generate such a signature.

The construction of the backdoor is explained as follows:

1. Let $m' \in \mathbb{Z}_2[X]/(X^n - 1)$ be the encryption of the secret information the attacker wants to leak. This can be information about the private key.

2. Let $s' \equiv m' \mod 2$ so that s' differs from s by 0 or ± 1.

3. Compute k such that:

$$s' \equiv f \circledast (C + k) + F \circledast c \mod q \tag{12}$$

Now, k will be a sparse polynomial with coefficients being either 1 or 0. How k is constructed, will be explained in later sections. The coefficients of s' taken modulo 2 represent the encrypted secret message m', in which the actual message is only to be decrypted by the attacker. This is because the attacker has encrypted the secret message. In the analysis coming next, the requirements for s to pass the verification and the influence of k on s will be studied.

5.4 Analysis of the backdoor in NTRUSign

The backdoor above is an example of a backdoor placed in this signature scheme. By changing s in that way, there is a chance that the signature will not pass the requirements for verification. In this analysis, mathematical reasoning will be used to determine the influence of k, the behavior of s' will be described and how many signatures the attacker needs, to leak his secret message.

The requirements for passing the signature are described in section 5.2. In short:

- (s, t) is a lattice point.
- $\frac{\|A \circledast g + a \circledast G\|_c}{q}$ should be small enough.
- $\frac{\|A \circledast f + a \circledast F\|_c}{q}$ should be small enough.

For the first requirement $(s', t) = (f \circledast (C + k) + F \circledast c, (f \circledast (C + k) + F \circledast c) \circledast h) = (f \circledast (C + k) + F \circledast c, g \circledast (C + k) + G \circledast c)$ needs to be a lattice point. Now, (s', t) can be written as $(C + k) \circledast (f, g) + c \circledast (G, H) \mod q$, so (s', t) is indeed a lattice point.

Now, with modified s the second check and the third check are:

- $\frac{\|A \circledast g + a \circledast G + k \circledast f\|_c}{q}$ should be small enough.
- $\frac{\|A \circledast f + a \circledast F + k \circledast g\|_c}{q}$ should be small enough.

For the analysis, the triangle inequality is used. This inequality is as follows: for every $\underline{a}, \underline{b} \in \mathbb{R}^n$, $\|\underline{a} + \underline{b}\|_2 \leq \|\underline{a}\|_2 + \|\underline{b}\|_2$. This also holds for the centered norm, as for all f, g the centered norm $\|f + g\|_c^2 = (\sum_{i=0}^{n-1}(f_i + g_i - (\mu_f + \mu_g))^2) = (\sum_{i=0}^{n-1}(f_i - \mu_f + g_i - \mu_g)^2) \leq (\sum_{i=0}^{n-1}(f_i - \mu_f)^2) + (\sum_{i=0}^{n-1}(g_i - \mu_g)^2) = \|f\|_c^2 + \|g\|_c^2$ using the triangle inequality above.

By this inequality the requirements can be written as:

- $\frac{\|A \circledast g + a \circledast G + k \circledast f\|_c}{q} \leq \frac{\|A \circledast g + a \circledast G\|_c}{q} + \frac{\|k \circledast f\|_c}{q}$
- $\frac{\|A \circledast f + a \circledast F + k \circledast g\|_c}{q} \leq \frac{\|A \circledast f + a \circledast F\|_c}{q} + \frac{\|k \circledast g\|_c}{q}$

Since $\frac{\|A \circledast g + a \circledast G\|_c}{q}$ and $\frac{\|A \circledast f + a \circledast F\|_c}{q}$ are already small enough for passing a normal signature [11], the behavior of both $\frac{\|k \circledast f\|_c}{q}$ and $\frac{\|k \circledast g\|_c}{q}$ should be analyzed.

To hide a one bit message in a signature, $k = x^i, i \in \{1, ..., n\}$ or $k = 0$ can be chosen. That way, the attacker is able to read off the message from the signature as the bits of the signature will represent the message. With this approach, in order to leak an ℓ bit message, ℓ signatures are required. The behavior of $\frac{\|k \circledast f\|_c}{q} = \frac{\|x^i \circledast f\|_c}{q}$ and $\frac{\|k \circledast g\|_c}{q} = \frac{\|x^i \circledast g\|_c}{q}$ will not affect the signature much because they are both sparse so that $\|f\|_c$ and $\|g\|_c$ are likely small enough. Multiplication by x^i will shift the bits of f i places to the left, so that the message can be read of from the left-most bit of the signature. So this product will not enlarge the centered norm of the product. Under normal circumstances this signature will pass the requirements. The disadvantage of this signature is that many signatures are needed to hide the secret message.

In general, to hide an ℓ bit message m' the worst case scenario would be k having ℓ coefficients being 1 and the rest being 0. Now $\|f\|_c = \|g\|_c = O(\sqrt{n})$ and for k, the entries will be either 0 or 1. When dividing by q, as seen in equation 10 and 11, this will result in an even smaller term because then k has entries of the value 0 or $1/q$. All together,

$\frac{\|k \circledast f\|_c}{q}$ and $\frac{\|k \circledast g\|_c}{q}$ will likely be small enough. To determine k, equation (12) is rewritten to $s' \equiv f \circledast (C + k) + F \circledast c \equiv f \circledast C + f \circledast k + F \circledast c \equiv s + f \circledast k \mod q$. Let f, s and m' be $f \equiv f_0 + f_1 X + ... + f_{n-1} X^{n-1} \mod 2$, $s \equiv s_0 + s_1 X + ... + s_{n-1} X^{n-1} \mod 2$ and $m' = m'_0 m'_1 ... m'_\ell$. For $k = k_0 + k_1 X + ... + k_{n-1} X^{n-1}$, one has to solve the following matrix:

$$\begin{pmatrix} k_0 & k_1 \dots k_{n-1} \end{pmatrix} \begin{pmatrix} f_0 & f_1 & \cdots & f_{n-1} \\ f_{n-1} & f_0 & \cdots & f_{n-2} \\ \vdots & \vdots & \ddots & \vdots \\ f_1 & f_2 & \cdots & f_0 \end{pmatrix} = \begin{pmatrix} |m'_0 - s_0| \\ |m'_1 - s_1| \\ \vdots \\ |m'_{\ell-1} - s_{\ell-1}| \\ s_\ell \\ \vdots \\ s_{n-1} \end{pmatrix}.$$

The entries of the solution vector are the coefficients of k. Using this k in equation (12), one retrieves s'.

Another way to retrieve k is to compute the inversion, i.e. for k being an element of $\mathbb{Z}_q/(X^n - 1)$, one can compute $k = (s' - F \circledast c) \circledast f^{-1} - C$. This would work, because f is invertible. In $\mathbb{Z}_2/(X^n - 1)$, f does not have to be invertible. The algorithm is a black-box system, so f can be chosen to be invertible in both $\mathbb{Z}_q/(X^n - 1)$ and $\mathbb{Z}_2/(X^n - 1)$. The downside of this inversion is that the coefficients of k can get really big.

A recommendation for the secret hidden message would be to put a one bit message in the signature. When the private key would consist of ℓ bits, the attacker would need ℓ signatures to leak the secret information. Because signatures will pass like normal signatures would do in [11], this is a safe bet.

5.4.1 Example

In the following section, a simple example is given how to construct k such that a two bits message is hidden in the signature. Let f be represented as the following string: `0010100001`. Modified signature s' can be written as $s' \equiv f \circledast (C + k) + F \circledast c \equiv f \circledast C + f \circledast k + F \circledast c \equiv s + f \circledast k \mod q$. So s is computed and in this example, s is represented by the following string: `0111000010`. In order to leak a two bits message, the left-most bits of s' have to be set to the message. Then the attacker can easily read off his hidden message from the modified signature. The term $f \circledast k$ is added to s and this way s' can be selected as follows:

- Let the hidden message be `00`. The term $f \circledast k$ should be represented by the string `01********` in order to set the first two bits of s' to `00`. Choose $k = X$, then the bits of f shift one place to the left and the result is that $f \circledast k$ is represented by `0101000010`. Adding this to the representation of f results in the following representation of s':

  ```
      0111000010
  +   0101000010 .
  ----------------
      0010000000
  ```

  So the representation of s' in binary is `0010000000`.

- Let the hidden message be `01`. The representation of s already starts with `01`, so choose $k = 1$ and the representation of s' in binary is `0111000010`.

- Let the hidden message be `10`. The term $f \circledast k$ should be represented by the string `11********` in order to set the first two bits of s' to `10`. Since shifting bits in f will not do the job, k should consist of two terms: one term to set the first bit of s' and the other term to set the second bit of s'. Choose $k = X^2 + X$, then $f \circledast k = f \circledast (X^2 + X) = f \circledast X^2 + f \circledast X$ in which $f \circledast X^2$ has representation `1010000100` and $f \circledast X$ has representation `0101000010`. So the following additions are done:

  ```
      0111000010
  +   1010000100
  ---------------
      1101000110
  ```

  and

  ```
      1101000110
  +   0101000010 .
  ---------------
      1000000100
  ```

  So the representation of s' in binary is `1000000100`.

- Let the hidden message be `11`. The term $f \circledast k$ should be represented by the string `10********` in order to set the first two bits of s' to `11`. Choose $k = X^2$, then the bits of f shift two places to the left and the result is that $f \circledast k$ is represented by `1010000100`. Adding this to the representation of f results in the following representation of s':

  ```
      0111000010
  +   1010000100 .
  ---------------
      1101000110
  ```

  So the representation of s' in binary is `1101000110`.

5.5 NSS

NSS, short for NTRU Signature Scheme, was the first version published of a signature scheme based on NTRU lattices [13]. There are a couple of versions made of NSS. In this thesis the version of EUROCRYPT '01 will be used. The scheme was very fast and had small keys. But in 2006 the scheme got broken. NSS got revised a couple of times, but none of them were successful.

First the following definition is given:

Definition 5.1. *Let $a(X)$ and $b(X)$ be two polynomials in $\mathbb{Z}[X]/(X^n - 1)$ and p and q integers. First reduce their coefficients modulo q to lie between $-q/2$ and $q/2$ and consider them as integers, then reduce their coefficients modulo p to lie in the range between $-p/2$ and $p/2$. If $\bar{a}(X) = \bar{a}_0 + ... + \bar{a}_{n-1}X^{n-1}$ and $\bar{b}(X) = \bar{b}_0 + ... + \bar{b}_{n-1}X^{n-1}$ are the reductions of a and b, respectively, then the deviation of a and b is Dev(a,b)= $\#\{i : \bar{a}_i \neq \bar{b}_i\}$. Intuitively, Dev(a,b) is the number of coefficients of $a \mod q$ and $b \mod q$ that differ modulo p.*

Key generation: for the key generation there are five parameters needed:
$(n, p, q, D_{\min}, D_{\max})$. NTRU made the following suggestions about these parameters, which are $(n, p, q, D_{\min}, D_{\max}) = (251, 3, 128, 55, 87)$. Let $\mathcal{F}_f, \mathcal{F}_g, \mathcal{F}_w$ be sets of sparse polynomials and the polynomials have degree $\leq n-1$. Now, $f_1 \in \mathcal{F}_f \subset R$ and $g_1 \in \mathcal{F}_g \subset R$. Then the signer chooses f, g such that:

$$f = f_0 + pf_1 \tag{13}$$

$$g = g_0 + pg_1 \tag{14}$$

with f_0 and g_0 fixed. In the article, they pick $f_0 = 1$ and $g_0 = 1 - 2X$. Then the signer computes the inverse of $f \mod q$: f^{-1}. The public key PK is $h = f^{-1} \circledast g \mod q$ and the private key SK is (f, g).

Creating signature s: given message m, which is hashed into a polynomial of degree $< n$ modulo p (for more details of the hashing, see section 5.6), the signer chooses $w \in \mathcal{F}_w$ of the form

$$w = m + w_1 + pw_2 \tag{15}$$

in which w_1 and w_2 have a specific form described in section 2.1 in [13]. In this process w_2 is randomly chosen with a specified number of 1's and -1's, and the construction of w_1 depends on w_2, $f \circledast (m + pw_2)$ and $g \circledast (m + pw_2)$. Then signature s is defined as

$$s \equiv f \circledast w \mod q. \tag{16}$$

Verification: in order to verify (m, s), check first if $s \neq 0$.
The second check is:

$$D_{\min} \leq \mathrm{Dev}(s, f_0 \circledast m) \leq D_{\max}. \tag{17}$$

Now polynomial $t \equiv h \circledast s \mod q$ is computed. Then the third check is:

$$D_{\min} \leq \mathrm{Dev}(t, g_0 \circledast m) \leq D_{\max}. \tag{18}$$

This works, because $s \equiv f \circledast w \equiv (f_0 + pf_1) \circledast (m + w_1 + pw_2) \equiv f_0 \circledast m + f_0 \circledast w_1 + pf_0 \circledast w_2 + pf_1 \circledast w \mod q$ and $t \equiv h \circledast s \equiv (f^{-1} \circledast g) \circledast (f \circledast w) \equiv g \circledast w \mod q$. The i-th coefficients of s and $f_0 \circledast m$ are the same modulo p when the following conditions hold: the i-th coefficient of $f_0 \circledast w_1$ is zero and the i-th coefficient of $f \circledast w$ is in $(-q/2, q/2]$. The article [13] explains that given the parameter set $(n, p, q, D_{min}, D_{max}) = (251, 3, 128, 55, 87)$ the conditions will hold. In short, these conditions tell that w_2 is sampled randomly from $\mathcal{F}_w$ and w_1 is either 0 or determined in a specific way. The analogy for the second check holds as well, because $t = h \circledast s \equiv h \circledast (f \circledast w) \equiv h \circledast f \circledast w \equiv g \circledast f^{-1} \circledast f \circledast w = g \circledast w \mod q$ and then replace f in the proof of the first check with g. When $D_{max} = D_{min} = 0$, these equations must hold:

$$s \equiv f_0 \circledast m \mod p \tag{19}$$

and

$$t \equiv g_0 \circledast m \mod p. \tag{20}$$

5.6 Sketch of a kleptographic backdoor in NSS

In this section, a sketch of a kleptographic backdoor in NSS [13] is presented. The backdoor is created in NSS, which is described in section 5.5 earlier.

The main idea is to hide information about the secret key in signature s itself in such a way that the generation of the polynomials f, g and w stays the same. Now, the conditions as described earlier apply when generating f, g and w. The secret information that the attacker wants to leak is encrypted and denoted by m'. As pointed out in section 5.3, the message is only readable to the attacker.

In this sketch of a backdoor, s is being modified by adding a term kp. The construction of k is as follows:

1. Let $m' \in \mathbb{Z}_2[X]/(X^n - 1)$ be the encryption of the secret information the attacker wants to leak. This can be information about the private key.

2. Let $s' \equiv m' \mod 2$ so that s' differs from s by 0 or ± 1.

3. Compute k such that:

$$s' \equiv f \circledast w + kp \mod q, \text{ with } k \in R \text{ and } k_i \in \{0, 1\}. \tag{21}$$

In this equation the coefficients for k are chosen so that the i-th coefficient of s is congruent to the i-th coefficient of $(f \circledast w) + kp$. The result is that an n-bit message $m' \in \{0, 1\}^n$ can be stored in s, e.g. with information about the secret key. This message m' is an encryption by some chosen public-key cryptosystem. Only the attacker knows the keys of this cryptosystem, so only the attacker can decrypt m'. Assuming the parameters chosen by NTRU, $p = 3$ is the case, which is odd. So with k, the parity of the i-th coefficient of s can be changed. This is a useful property, because the parity of the coefficients will leak the m' to the attacker. The agreement of an odd coefficient being 1 and an even coefficient being 0 can be made.

5.7 Analysis of the kleptographic backdoor in NSS

The backdoor above is an example of a kleptographic backdoor placed in this signature scheme. By changing s in that way, there is a chance that the signature will not pass the requirements for verification. In this analysis, mathematical reasoning will be used to determine the influence of k, the behavior of s' will be described and how many signatures the attacker needs to leak his secret message.

Note that the secret key consists of two polynomials f and g. These polynomials both have up to n coefficients. If an attacker wants to leak them both, this cannot be done by intercepting one signature. Since s' is also a polynomial with n coefficients, $k_i p$ can only be added n times, which means that the hidden message can only be n bits long. A solution to this could be waiting till the attacker gets two signatures (in each one there is a polynomial stored).

A practical implementation is given by the article. The description of the scheme is as described in section 5.5. Polynomials f_1 and g_1 are sampled by the set of polynomials with resp. d_{f_1}, d_{g_1} coefficients being 1, resp. d_{f_1}, d_{g_1} coefficients being -1 and the rest 0. f and g are in particular sparse, i.e. d is rather small. m is in particular given in the form $\sum_{i=1}^{32} X^{e_i} - \sum_{i=33}^{64} X^{e_i}$, in which e_i are the assigned distinct integers for m with $0 \leq e_i \leq 251$ for all i. This is done by a chosen method. NTRU chooses the following settings: $d_{f_1} = 70$, $d_{g_1} = 40$, $d_{w_2} = 32$ and for w_1 the number of non-zero coefficients does not exceed 25.

According to the article, a signature generated given the parameters is valid with a probability of 79.40%. When a signature is not valid, the algorithm selects a new w_2 and generates a new signature. As described earlier, a signature is valid when it satisfies equations (17) and (18).

Now the signatures produced from the modified signature scheme are being analyzed. Let such a signature s' be generated from this scheme. If s' passes the verification, then the users of the modified program will not have a clue about the leak. If s' does not get accepted by the verification tests, then one could think there is something suspicious about the program. If generation takes much longer, it is even more suspicious.

Now if one of the following situations occurs, the signature will not be accepted (for simplicity's sake $D_{max} = D_{min} = 0$):

1. The i-th coefficient of s' does not match with the i-th coefficient of $f_0 \circledast m$ after modulo p reduction or the i-th coefficient of $t = h \circledast s$ does not match with the i-th coefficient of $g_0 \circledast m$ after modulo p reduction.

2. The coefficients of $s = f \circledast w$ are outside the range of $(-q/2, q/2]$.

3. The coefficients of $t = g \circledast w$ are outside the range of $(-q/2, q/2]$.

When $D_{\max}$ and $D_{\min}$ do have the values set by NTRU, then the first two situations are modified into equations (17) and (18). In that case there can be at most $D_{\max}$ exceptions to this rule.

Signature s', generated by the modified signature scheme, will not encounter the first situation, because the additional term kp will vanish after modulo p reduction. For t, it holds that $t = h \circledast s' \equiv h \circledast (f \circledast w + kp) \equiv h \circledast f \circledast w + h \circledast kp \equiv g \circledast w + h \circledast kp \mod q$. This shows that the i-th coefficient of the polynomial on the right hand will not contain anything from polynomial $h \circledast kp$ since there is a modulo p reduction. Note that the modulo q reduction is ignored. So under the same circumstances described earlier the first situation will not occur.

When adding 3 to coefficients, they are still in the range of $(-q/2, q/2]$ since $q = 128$. In the worst case, the width of s' increases with 6, but will not likely be greater than q. When this happens, a new signature is produced. The probability that situation two occurs for a second signature is really small, so situation two will not be a problem.

Now, the third situation may be more realistic, since the coefficients $h \circledast s'$ can get too big quite easily. f^{-1} is not sparse and the coefficients are in the full interval modulo q,

therefore h is not either sparse. Furthermore, h has coefficients between $(-q/2, q/2]$. Then $h \circledast kp$ can grow into a polynomial of any size. For further research, this behavior could be analyzed experimentally and theoretically and solutions to this exponential growth could be found.

An alternative way to leak information is to hide just one bit of the ℓ bit message in the signature by changing the first bit of the signature. Then the attacker can read of this bit from the first bit. The attacker needs ℓ signatures to leak an ℓ-bit secret message.

5.7.1 Example

In the following section, an example is given how to construct k such that a two bit message is hidden in the signature. Let s be the polynomial $s = 2 + 5X^2 - 6X^4 + X^9$. Modified signature s' can be written as $s' \equiv f \circledast w + kp \equiv s + kp \mod q$. In order to leak a two bit message, the left-most coefficients of s' have to be set to the message. Then the attacker can easily read off the hidden message from the modified signature. First, $s \mod 2$ is computed, so $s \equiv X^2 + X^9$. The term kp is added to s and this way s' can be selected as follows:

- Let the hidden message be `00`. The first two coefficients are even, so nothing has to be changed.
- Let the hidden message be `01`. Now the second coefficient has to be changed. Choose $k_1 = 1$, then the second coefficient is odd, so $s' \equiv X + X^2 + X^9 \mod 2$.
- Let the hidden message be `10`. Now the first coefficient has to be changed. Choose $k_0 = 1$, then the first coefficient is odd, so $s' \equiv 1 + X^2 + X^9 \mod 2$.
- Let the hidden message be `11`. Now the first and the second coefficient have to be changed. Choose $k_0 = 1$ and $k_1 = 1$, then the first and second coefficient are odd, so $s' \equiv 1 + X + X^2 + X^9 \mod 2$.

One can easily generalize this process to hiding an ℓ-bit message with $\ell \leq n$.

5.8 Summary

In this chapter, two ideas for new kleptographic backdoors in NTRU are presented. The backdoors could be used exclusively by the attacker to leak secret information. They could also be used to transmit unnoticeable messages from one party to the other.

6 Conclusions

This chapter starts with some final words about NTRU signature schemes. Then a short summary of the results of this thesis is presented. Finally a couple of ideas for further research are given.

6.1 Final remarks on NTRU Signature Schemes

NTRU signature schemes had been broken many times, but nevertheless doing research into these broken systems is actually useful since the field of lattice-based cryptosystems is young and there is still a lot to be explored about nice properties of lattices. The presented ideas for backdoors may be a good fundamental base for possible backdoors in other lattice-based signature schemes.

In this thesis, possibilities for leaking secret information using a signature are presented. Two backdoors in two different digital signature schemes are explored: one backdoor made in NTRUSign and one backdoor made in NSS. For both of them the results are presented:

- The backdoor in NTRUSign presented in this thesis is a sketch of a way to leak the secret information. By encrypting the secret message in signature s, the attacker can recover his secret message from s by parity checking. Verification problems arise when the centered norm of the cyclic convolution product of some polynomials gets too big.

- The backdoor in NSS encapsulates the secret message by setting the coefficients of s to an even number or an odd number. The parity of the coefficients leaks the secret message in binary. From this the attacker can recover the secret message. Verification problems arise when the cyclic convolution product of some polynomials gets too big.

6.2 Future research

During the research, there were many angles to explore for possible backdoors. Due to limited time, this thesis could not cover them all. Here are a couple of these ideas for further research:

- In this thesis a lot of digital signature schemes are given with their algorithms for keys and signatures. In particular BLISS was very interesting, because there is actually some randomness involved.

- The sketches of the backdoors presented in this thesis could be implemented in suitable programs. Then, by generating a couple of signatures, one could research how many times the signatures actually get rejected. Maybe it is possible to reset the parameters to different values such that the probability of rejecting gets smaller. Also the running time of the original signature scheme and the modified signature scheme can be compared.

7 Appendix

Table 1: Notation table

Variable	Notation
p, q	Integers
n	Integer, in most cases denotes dimension
L	Lattice
B, B'	Good resp. bad basis of lattice L
m, m'	Message
s	Signature
c	Ciphertext
i, j, k, l	Integers
τ	Real number, is usually small
$\underline{x}$	Underlining is used to denote vectors
$\mathbb{N}$	Set of all the positive integers including 0
$\mathbb{Z}$	Set of all the integers
$\mathbb{Q}$	Set of all rational numbers
$\mathbb{R}$	Set of all real numbers
K^*, L^*, B^*	Multiplicative group of K, dual lattice of L, Gram-Schmidt Orthogonalization of B
$\{0,1\}^*$	The set of arbitrary long binary expressions
$< \underline{a}, \underline{b} >$	The inner product of two vectors $\underline{a}$ and $\underline{b}$
PK, SK	Public key resp. secret key
$H(a\|\|b)$	The concatenation of a and b by hash function H
$\lceil a \rceil$	The number a, rounded to the next integer larger than or equal to a
$\lfloor a \rfloor$	The number a, rounded to the next integer smaller than or equal to a
$\lfloor a \rceil$	The number a, rounded to the closest integer
$\mathcal{D}_\sigma^n$	The centered discrete Gaussian distribution with parameter σ
$\mathcal{F}_f$	Subset of $\mathbb{Z}[X]/(x^n - 1)$, set of polynomials with small coefficients
$\det(A)$	The determinant of square matrix A
$\mu_{i,j}$	Gram-Schmidt coefficient
$\|\underline{a}\|_2, \|f\|_c$	Euclidean length of vector $\underline{a}$, centered norm on polynomial f
I_n	$n \times n$ identity matrix
O	null matrix
A^T, U	The transposed matrix of A, unimodular matrix U

References

[1] Milou Antheunisse. Kleptography cryptography with backdoors. Master's thesis, Eindhoven University of Technology, 2015. `http://repository.tue.nl/801620`.

[2] Marco A. Barreno. The future of cryptography under quantum computers. `http://aiweb.techfak.uni-bielefeld.de/content/bworld-robot-control-software/`, 2002.

[3] Leon Groot Bruinderink. Towards post-quantum bitcoin. Master's thesis, Eindhoven University of Technology, 2016. `http://repository.tue.nl/844305`.

[4] Hans Cuypers, Hans Sterk, and Arjeh M. Cohen. *Algebra-Interactive*. Springer-Verlag Berlin Heidelberg, 1999.

[5] Whitfield Diffie and Martin E. Hellman. New directions in cryptography. *IEEE Trans. Information Theory*, 22(6):644–654, 1976.

[6] John F Dooley. *A Brief History of Cryptology and Cryptographic Algorithms.* Springer, 2013.

[7] Léo Ducas, Alain Durmus, Tancrède Lepoint, and Vadim Lyubashevsky. Lattice signatures and bimodal gaussians. In *CRYPTO (1)*, volume 8042 of *Lecture Notes in Computer Science*, pages 40–56. Springer, 2013.

[8] Steven D. Galbraith. *Mathematics of Public Key Cryptography*. Cambridge University Press, 2012.

[9] Craig Gentry and Michael Szydlo. Cryptanalysis of the revised NTRU signature scheme. In *EUROCRYPT*, volume 2332 of *Lecture Notes in Computer Science*, pages 299–320. Springer, 2002.

[10] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. *Electronic Colloquium on Computational Complexity (ECCC)*, 3(56), 1996.

[11] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN:: digital signatures using the NTRU lattice: Preliminary draft 2. `http://www.math.brown.edu/~jpipher/NTRUSign-preV2.pdf`, 2002.

[12] Jeffrey Hoffstein, Nick Howgrave-Graham, Jill Pipher, Joseph H. Silverman, and William Whyte. NTRUSIGN: digital signatures using the NTRU lattice. In *CT-RSA*, volume 2612 of *Lecture Notes in Computer Science*, pages 122–140. Springer, 2003.

[13] Jeffrey Hoffstein, Jill Pipher, and Joseph H. Silverman. NSS: an NTRU lattice-based signature scheme. In *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 211–228. Springer, 2001.

[14] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *EUROCRYPT*, volume 7237 of *Lecture Notes in Computer Science*, pages 738–755. Springer, 2012.

[15] Phong Q. Nguyen and Oded Regev. Learning a parallelepiped: Cryptanalysis of GGH and NTRU signatures. *Journal of Cryptology*, 22(2):139–160, 2009.

[16] Ron L. Rivest, Adi Shamir, and Len Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[17] Adam L. Young and Moti Yung. The dark side of "black-box" cryptography, or: Should we trust capstone? In *CRYPTO*, volume 1109 of *Lecture Notes in Computer Science*, pages 89–103. Springer, 1996.