THE UNIVERSITY OF THE SOUTH PACIFIC
LIBRARY

Author Statement of Accessibility

Name of Candidate : Ravinesh Chand

Degree : Master of Science in Mathematics

Department/School : School of Computing, Information & Mathematical Scien ~~Science~~

Institution/University : The University of the South Pacific

Title of Thesis : Digital Signature Scheme Over Lattices

Date of award : April, 2019
(Month, year)

Please tick ✓
This thesis may be:

1. consulted in the Library without the author's permission **YES** ✓ **NO** ☐
2. cited without the author's permission provided it is properly acknowledged **YES** ✓ **NO** ☐
3. photocopied in whole without the author's written permission **YES** ✓ **NO** ☐
   If you answered '**No**', select the percentage that may be photocopied

Under 20% ☐ **20-40%** ☐ **40-60%** ☐ **60-80%** ☐ **Over 80%** ☐

I authorize The University of the South Pacific to:

4. produce a microfilm or microfiche copy for preservation purposes **YES** ✓ **NO** ☐
5. retain a copy in electronic format for archival and preservation purposes **YES** ✓ **NO** ☐
6. make this thesis available *on the Internet*
   YES ✓
   NO ☐ **Go to Question 7**
7. make this thesis available on the USP Intranet
   YES ✓
   NO ☐ **Give reasons** ______

Signed : [illegible] Date : 4/4/19

Contact Information

Email address : ravinesh.c@fnu.ac.fj

Phone
Mobile : 9960667
Home :
Work : 3381044 ext 1554

Permanent Residential Address
Lot 49 Mukta Ben Road
Vatuwaqa

DIGITAL SIGNATURE SCHEME OVER LATTICES

by

Ravinesh Chand

A thesis submitted in fulfillment of the requirements for the degree of Master of Science in Mathematics

School of Computing, Information and Mathematical Sciences
Faculty of Science, Technology and Environmental Sciences
The University of the South Pacific

March, 2019

Declaration of Originality

I hereby declare that this thesis is original to the best of my knowledge, citations and references have been acknowledged and the main result formulated has not been previously submitted for a university degree either in whole or in part elsewhere.

Signature: Chand Date: 10/04/2019

Name: **Ravinesh Chand** Student ID: S99006990

Declaration by Supervisors

This is to certify that the thesis titled "Digital Signature Scheme over Latices" by Mr. Ravinesh Chand, under our direct supervision represents the original research work of the candidate.

[illegible] Date: 10. 4. 2019

Associate Professor Dr. MGM Khan

Principal Supervisor

[illegible] Date: 03/04/2019

Associate Professor Dr. Maheswara Rao Valluri

Co-supervisor

Dedication

This thesis is dedicated to my wife Mrs. Saileshni Chand

Acknowledgments

I wish to express my appreciation to my thesis advisors, Dr. Maheswara Rao Valluri, of the Fiji National University, and Dr. MGM Khan, of the School of Computing, Information and Mathematical Sciences at the University of the South Pacific. I deeply express my sincere appreciation for their time, valuable suggestions, motivation, and encouragements.

I would also wish to show my appreciation to my son Ronav Chand and my daughter Sanvi Chand for their continuous support.

I would also like to thank Mr. Sandeep Ameet Kumar for providing technical assistance.

Abbreviation

The table below shows the list of abbreviations used throughout this thesis.

Abbreviation	Meaning
CAPTCHA	Completely Automated Public Turing test to tell Computers and Humans Apart
CVP	Closest Vector Problem
DLP	Discrete Logarithmic Problem
DSS	Digital Signature Scheme
ECC	Elliptic Curve Cryptography
ECDLP	Elliptic Curve Discrete Logarithmic Problem
IFP	Integer Factorization Problem
LWE	Learning With Errors problem
NTRU	N-th degree Truncated Polynomial Ring Unit
PQC	Post Quantum Cryptography
RSA	Rivest, Shamir and Adleman scheme
R-LWE	Ring - Learning With Errors problem
R-SIS	Ring - Short Integer Solution problem
SVP	Shortest Vector Problem

Abstract

With the rapid growth and developments in information technology, data security has become an integral component. Cryptography plays a very important role in establishing information security. Computational problems have been utilized aggressively by cryptographers to provide strong and secure signature schemes. Digital signature schemes consist of algorithms that ensures confidentiality, authenticity, integrity and non-repudiation of a message. In this thesis, we develop a digital signature scheme which is secure under chosen message attack based on the hardness of lattice problems such as Learning With Errors and Short Integer Solution over lattices. Our proposed scheme is potentially practical. Signing and verifying our signature seems reasonably fast, and the size of the signature seems compact and yet to be verified.

Preface

This research project titled "Digital Signature Scheme over Lattices" is submitted to the University of the South Pacific, Suva, Fiji in fulfillment of the requirements to acquire a degree in Master of Science in Mathematics.

Cryptography contributes immensely towards data security. Signature schemes play a key role in safe data transmission and authentication. Most of the existing signature schemes based on the number theoretic cryptographic assumptions are not secure due to the Shor's algorithm on a quantum computer. The threat of quantum computer attacks and the relevance of signature schemes for information technology security make the development of post-quantum signature schemes necessary. Lattice-based schemes have become very popular, as well as leading the way towards post-quantum cryptography. Our proposed digital signature scheme is based on the hardness of lattice problems.

Chapter 1 gives an overview of the literature review and studies on the similar work done by researches in the field of cryptography. It elaborates on the contribution of number theory hard problems, and the importance of lattice based cryptography in the post quantum world.

Chapter 2 discusses the background resources required for the construction of our scheme. It highlights on the role of a digital signature scheme and the opportunities it provides to enhance data security.

Chapter 3 provides information on lattices. It discusses the sampling technique used in this thesis, and looks at some highly effective hardness problems on lattices.

Chapter 4 presents the proposed signature scheme, its security and performance analysis, and a brief summary of how the scheme works.

Chapter 5 gives conclusion and provides key findings of this research with recommendations for further research.

Contents

Chapter 1

Introduction

Cryptography is a method of protecting information and communications through the use of codes so that only those for whom the information is intended can read and process it. Cryptographic algorithms are employed to enhance information security. Cryptography is classified as the symmetric key (private or secret key cryptography) and the asymmetric key cryptography or public key cryptography.

In the symmetric key cryptography, a sender and a receiver have a common key, called private key, which can be used for encryption and decryption of data or message, whereas in the asymmetric key cryptosystem, a third party authority generates a pair of private key and public key for the sender and receiver. This private and public key pair is mathematically associated. As the name suggests, the public key is made available to everyone, while the private key must remain confidential to its respective owner.

Cryptographic primitives consist of well-established mathematical algorithms that form the foundation of secure signature schemes. Most of the notable public key cryptographic primitives are encryption, key exchange protocol, authentication protocol, and digital signatures such as blind signature, proxy signature, ring signature, group signature, and identity based signature schemes.

In the modern day internet era, signature schemes play a crucial role to provide confidentiality, authenticity, data integrity and non-repudiation [5]. The typical applications of signature schemes are used in smart cards (like debit and credit cards), networking, e-mails, web applications, and server applications.

Public key cryptosystem was first introduced by Diffie and Hellman in 1976 in their seminal paper [11] in which they proposed a key exchange protocol called DH-Key Exchange Protocol based on the Discrete Logarithmic Problem (DLP) over a finite field. Subsequently, Rivest, Shamir, and Adleman invented a public key encryption scheme based on the Integer Factorization Problem (IFP) known as RSA in 1978 [45]. Clearly, the three researchers named their scheme as RSA from the initials of their last names - Rivest, Shamir, and Adleman. Furthermore, ElGamal introduced public key encryption scheme and digital signatures in early 1980s [15]. Elliptic curve cryptosystem was introduced independently by Miller in 1985 [35], and later, by Koblitz in 1987 [25] using the elliptic curve discrete logarithmic problem (ECDLP) over finite fields. Most of these problems (integer factorisation problem, discrete logarithmic problem and elliptic version of discrete logarithmic problem) are based on number theory. These problems are tractable due to the Peter Shor's algorithm over a quantum computer [49].

The Short Integer Solution (SIS) problem was introduced by Ajitai in 1996, where he proposed the first worst-case to average-case reduction for a lattice problem. In addition, he introduced a subset-sum problem in his paper, also referred to as knapsack problem [2] for cryptographic primitives. The encryption schemes that heavily rely on SIS enjoy the property of being provably as secure as worst-case problems which are strongly suspected to be extremely hard to solve.

However, the applications of cryptography that are related to SIS are inherently inefficient due to the size of the associated key or public data. This means that such schemes are essentially unable to achieve its maximum potential. With the inspiration of Ajtai's problem, Hoffstein, Pipher and Silverman developed the N-th degree Truncated Polynomial Ring Unit (NTRU) cryptosystem in 1998 [23].

From 2005 onwards, numerous applications of the Learning With Errors problem (LWE) were developed by researchers. However, as claimed in [37], these applications are still inefficient and further studies are still being carried out on attaining an efficient and provable secure signature scheme. To evade this inherent inefficiency, Micciancio, inspired from the efficient NTRU encryption scheme that can itself be interpreted in terms of lattices, introduced a scheme in [32] that consists of changing the SIS problems to modifications involving structured matrices. The scheme initiated by Micciancio in 2007 was later replaced by a more powerful alternative in [38], now commonly referred to as SIS problem over Rings, or R-SIS. Peikert and Rosen observed that solving R-SIS exactly consists of finding a short nonzero vector in a module lattice [38]. In 2010, Lyubashevsky et al. [30] constructed an efficient ring counterpart to Regev's 2005 learning with errors problem. In their work, they demonstrated that this problem can be reduced to the worst case hardness of short-vector problems on ideal lattices. Thus, ring LWE problem was introduced. R-LWE is a ring version of the LWE problem. R-LWE is a computational hard problem which provides a very healthy foundation for upcoming cryptographic algorithms to defend against quantum computer attacks [29].

In 2011, Lyubashevsky used the R-LWE version of the classic Feige-Fiat-Shamir identification protocol and converted it to a digital signature [28]. The details of this signature were thoroughly discussed in 2012 by Guneysu et al. [21].

In recent years there has been tremendous growth in lattice-based cryptography as a research field. As a result, concepts such as encryption [5], identity-based encryption [1, 13], attribute-based encryption [6], group signature schemes [20], and fully homomorphic encryption [18] are now available.

In today's world of information technology, the provenance of quantum computers is ever impending. In the near future, there is a possibility of a transit from classical computers to quantum computers. The computational aptitude it could provide would cause immediate insecurity to majority of the cryptographic schemes which are in use today. This implies that existing signature schemes that are based on number theory, including integer factorization problem, discrete logarithmic problem, and the elliptic version of discrete-logarithm problem, would become sensitive to violation. In fact, these problems have been broken due to the Shor's algorithm over a quantum computer [49]. As a result, this has motivated the era of post-quantum cryptography (PQC), which refers to the study and construction of cryptographic algorithms to withstand quantum subjugation.

Post-quantum cryptography algorithms will be resistent to attacks by quantum computers. There are five interesting candidates for post-quantum cryptography, which are multivariate cryptosystem that is based on polynomials over defined finite fields [10], code-based cryptosystem which is based on the use of an error correcting code [44], hash-based cryptosystem that is based on the security of crptographic hash functions [8], super singular elliptic curve isogeny that uses conventional elliptic curve operations [25], and lattice based cryptosystem that involves lattices, either in construction or security proof of the scheme [21, 18]. Amongst these, the most prominent candidate in terms of design and efficiency is lattice based cryptography [18]. Its main advantage over other post-quantum cryptosystems is that it allows for improved performance and is, at the same time, more flexible for the basic requirements of public-key encryption and digital signature schemes. There are various computational problems that are existing nowadays within the lattice environment. These include problems such as finding the shortest vector (SVP), or finding a closest vector (CVP), which are thought to be resilient to quantum-computer attacks [27]. Such properties show tremendous potential with respect to security and achievability for substituting the existing asymmetric schemes that would be vulnerable to attacks in a post-quantum world.

This thesis was inspired by the latest advances and developments of signature schemes, and their contribution towards enhanced electronic security. Post-quantum signature schemes are very essential component of a cryptosystem. In terms of security, they have a very significant purpose, more specific than the conventional encryption prototype, and are used in a variety of areas; from legal issues such as document integrity to those that support the world's economy through electronic commerce.

Chapter 2

Background

2.1 Notations

In this section, we will define the notations used throughout the work in order to avoid ambiguity.

- For a distribution S, we write $S \longleftarrow \chi$ to denote that S is chosen uniformly at random from χ.
- pk means public key and sk means private key or secret key.
- M is a message and μ is sampled from M such that $\mu \in M$.
- x and I are random positive integers.
- E represents an elliptic curve.
- $\mathbb{F}_p$ is a finite field.
- $\mathbb{R}$ is the set of real numbers.
- $\mathbb{Z}$ is the set of integers.
- $\mathbb{N}$ is the set of natural numbers.
- $\mathcal{L}$ is used to symbolize a lattice.

- χ and $\mathbf{D}$ are uniform distributions.
- ϕ is used for Gaussian distribution.
- Ω is used to denote a random variable in $\mathbb{R}$.
- c represents the hash value.
- σ means standard deviation.

In our proposed algorithm, the secret keys are $\mathbf{S}$, $\mathbf{E}$, and $\mathbf{T}$, while the public keys are $\mathbf{T}_1$ and $\mathbf{A}$. The parameters that dictate rejection probability in sampling are δ and β. A fixed relatively small integer, τ, is to be chosen such that $\parallel c.\mathbf{S} \parallel_\infty \leq \tau$ over the choices of the hash value c and the secret key $\mathbf{S}$. We use ξ to represent the signature.

Column vectors are represented by bold lower-case letters, while matrices are represented by bold upper-case letters.

2.2 Digital Signatures

Digital signature is an electronic signature that is used to authenticate digitally transferred data and to ensure that the content of the message or the document sent has not been altered. It uses asymmetric cryptography to encrypt the data, thus providing reasons to believe that the data was sent by the claimed sender [48]. The four main characteristics of digital signatures are:

- Confidentiality: Upon encryption of the data, it is confidential.
- Authentication: The receiver is able to verify the identity of the sender.
- Integrity: The data is not tampered during the transfer.
- Non-repudiation: The sender can not deny that he/she signed that particular document.

A digital signature has the ability to validate the authenticity and integrity of a message, software or digital document. This is done using complicated computational problems

[29]. It is the digital version of a handwritten signature. It offers far more intrinsic security, and possesses the ability to identify tampering and impersonation in digital communications.

To create a digital signature, the data to be signed is used to construct a one-way hash by a signing software, for example, an email program. Then, the hash is encrypted with the private key. Thus, a digital signature is formed with the encrypted hash and the hashing algorithm. Hashing plays a major role in digital signature schemes. It can convert an arbitrary input into a fixed length value, that is usually much shorter as well as faster to implement. Hashing is a function that condenses a message into an irreversible fixed-length value, or hash. It is designed to be a one-way function, that is, a function which is impossible to invert. A hash function is defined as follows:

Definition 2.2.1. A hash function is defined by $c = H : \{0,1\}^* \longrightarrow \{0,1\}^k$. This takes a message as an input and returns a fixed-size bit string. This string is called the *hash value* or *message digest*.

The hash function has three main properties:

1. It is extremely easy to calculate a hash for any given data.
2. It is extremely computationally difficult to calculate an alphanumeric text that has a given hash.
3. It is extremely unlikely that two slightly different messages will have the same hash value.

The hash value is always unique to the hashed data. If the data is altered or influenced in any little way, then a different hash value is created. This property is very beneficial, as it enables others to validate the integrity of the data by using the signer's public key to decrypt the hash. If the hash value obtained after decryption matches a second computed hash of the same data, then this means that the data was not altered since it was signed. If the two hash values do not match, then it is obvious that the message has either been tampered with in some way (integrity) or the signature was created with a private key that doesn't correspond to the public key presented by the signer (authentication).

An important feature of digital signature schemes is its flexibility. It can be applied with any kind of message, whether it is encrypted or not as shown in [8]. Digital signatures make it arduous for a signer to deny having signed something (non-repudiation), assuming their private key has not been negotiated. The digital signature binds both the document and the signer, hence becomes unique in the process.

The concept of digital signatures is an application of public key cryptography. One can generate two keys that are mathematically connected - a public key, *pk*, and a private key, *sk*. The key *sk* is used to generate signatures, while, the key *pk* is used to verify signatures. Since *pk* is a public key, it is made available for everyone, meaning, the attacker knows it too. So while only a signer in possession of the private key can generate signatures, anyone in possession of the corresponding public key can verify the signature.

Definition 2.2.2. A *Digital Signature Scheme DSS* = (*KeyGen, Sign, Verify*) consists of three algorithms, as follows:

- *KeyGen* - From a distribution of possible private keys, a private key is generated uniformly at random using a key generation algorithm. The algorithm outputs the private key *sk* and a corresponding public key *pk*.
- *Sign* - The signing algorithm *Sign* takes the private key *sk* and a message *M* to produce a signature ξ. We write $\xi \longleftarrow Sign_{sk}(M)$ or $\xi \longleftarrow Sign(sk, M)$ for the algorithm of running *Sign* on inputs *sk, M* and letting ξ be the signature returned.
- *Verify* - The deterministic verification algorithm takes a public key *pk*, a message *M*, and a candidate signature ξ for *M* to return a bit. We write $Verify(pk, M, \xi)$ to denote the algorithm of running *Verify* on inputs *pk, M*, and ξ. This algorithm either accepts or rejects the signature.

The scheme works in the following manner. Let *S* be an entity that wants to have a digital signature capability. Firstly, the algorithm *KeyGen* is run by *S* to generate a pair of keys (*pk, sk*) for itself. Secondly, $Sign_{sk}(M)$ is run by *S* to produce a digital signature on some document $M \in$ Messages(*pk*), which is the set of all messages, to produce a signature ξ. The pair (*M*, ξ) is then the authenticated version of the document. Thirdly, after receiving a document M' and tag ξ' supposedly to be from *S*, a receiver who has possession of *pk* verifies the authenticity of the signature by applying the specified verification procedure.

This depends on the message, signature, and public key. Specifically, the receiver verifies the message obtained by computing $Verify_{pk}(M', \xi')$, whose value is a bit. If this value is 1, it is read as saying the signature is authentic, and so the receiver accepts it as coming from S. Else it discards the signature as unauthentic.

It is necessary that any receiver willing to verify S's signatures must be in possession of S's public key *pk*, and must assure that the public key is authentic, meaning really is S's key and not someone else's key. Finally, the *Verify* algorithm checks that the signatures that were correctly generated will pass the verification test. This check ensures that authentic signature will be accepted by the receiver.

Majority of the modern email programs in practice today support the use of digital signatures, treating it as an easy mechanism to sign any outgoing emails and validate digitally signed incoming messages. Digital signatures are also utilized extensively to guarantee proof of authenticity, data integrity and non-repudiation of communications and transactions conducted over the internet.

2.3 Computational Hard Problems and Elliptic Curves

Number theory consists of concepts that present many problems that are hard to solve [17]. Thus, over the past years, number theory has become a major contributor to cryptographic algorithms. It has contributed some popular and widely used problems in the construction of secure signature schemes.

2.3.1 Integer Factorization Problem

In the field of number theory, the integer factorization problem refers to the decomposition of a composite number into it's product, usually in the form of smaller integers. To make this concept hard enough to be considered in cryptographic algorithms, the method of prime factorizations is employed, where the integers are restricted to prime numbers. The underlying difficulty of this problem plays a major role in many of the practically used algorithms in cryptography such as RSA [17].

Definition 2.3.1. *[17]* The *Integer Factorization Problem (IFP)* is as follows: given a positive integer I, compute its decomposition into prime number components e_i and d_i as $I = \prod_{d_i}^{e_i}$(unique up to reordering).

2.3.2 Discrete Logarithmic Problem

Discrete logarithms are logarithms defined with regard to multiplicative cyclic groups. The hardness of finding discrete logarithms depends on the cyclic group, which is is a group that is generated by a single element.

Definition 2.3.2. *[17]* Let G be a cyclic group of order N generated by P. Assuming that there exists a polynomial time algorithm for computing the group law in G, and that P is a given integer, the *Discrete Logarithm Problem (DLP)* in G is defined as follows: given any element y in G, compute an integer $x \in G$ such that $y = g^x$.

2.3.3 Elliptic Curve Discrete Logarithmic Problem (ECDLP)

Elliptic Curve Cryptography (ECC) is a very effective technique based on public key encryption. It employs the concept of elliptic curve theory and optimizes its advantages to create faster, smaller, and more efficient cryptographic keys. ECC generates keys using the characteristics of the elliptic curve equation rather than the commonly used method of generation as the product of very large prime numbers.

Basically, an eliptic curve is defined as follows:

Definition 2.3.3. *[22]* An *elliptic curve* is a nonsingular plane algebraic curve over some infinite field, resulting in a smooth plane cubic curve with the point at infinity, and we can describe the curve as points satisfying the equation: $y^2 = x^3 + ax + b$, with a and b such that the discriminant, $\nabla = -16(4a^3 + 27b^2)$, is nonzero (which will give the desired nonsingularity).

ECC has been widely studied by researchers over the past years and has been combined with various hard mathematical problems to enhance the security of signature schemes. One such concept is the Elliptic Curve Discrete Logarithmic Problem (ECDLP).

We now define the ECDLP.

Definition 2.3.4. *[46]* Let E be an elliptic curve over some defined finite field, $\mathbb{F}$, and let P and Q be points in $\mathbb{F}$. The *Elliptic Curve Discrete Logarithm Problem* is the problem of computing an integer x, where $x \in [0, p-1]$, such that $Q = xP$. We denote this integer x by $x = \log_P(Q)$ and we call x the elliptic curve discrete logarithm of Q with respect to P.

The ECDLP is the mathematical trapdoor function underpinning elliptic curve. This problem is the fundamental building block for elliptic curve cryptography and pairing based cryptography, and has been a major area of research in computational number theory, geometry, and cryptography for several decades.

This computational problem is assumed to be hard and resistent to attacks. ECDLP was widely believed to be one of the hardest computational number theory problem used in cryptography. In [39, 46, 41], the authors provide new index calculus algorithms to solve ECDLP over fields of prime cardinality. Furthermore, ECDLP has also suffered from index calculus attacks over finite fields of sub-exponential complexity using recommended key sizes [16].

2.4 Existing Signature Schemes

In this section, we briefly highlight the workability and the mathematical background of some existing digital signature schemes. Throughout this section, the name Bob is used as the sender and Alice is used as the receiver of the message.

2.4.1 RSA Digital Signature Algorithm

As discussed in detail in [24], this algorithm employs the integer factorization problem. This method contemplates the public key of Bob and hash function is publically known. Bob performs the following to commence:

1. Selects two prime numbers p and q and computes $N = p.q$.
2. Computes Euler's Totient Function $\phi(N) = (p-1).(q-1)$.

3. Computes the secret key sk such that $sk.pk$ = 1 mod $[\phi(N)]$.

   Thus, the public key set of Bob contains N and pk, and private key contains N and sk. Bob uses the private keys to generate the signature of the message.

4. Bob hashes the message M to obtain c, i.e $c = H(M)$.

5. Bob can now generate the digital signature sign = c^{sk} mod N, where sign is the signature.

   Once the signature has been produced, Bob sends (M, sign) to Alice.

6. Alice uses the $H()$ to obtain the $c' = H(M')$, i.e hash′ function.

7. Alice decrypts the signature to retrieve its hash $c =$sign^{pk} mod N.

8. Alice finally checks if $c = c'$.

If the match is attained in the hash value retrieved and the hash value calculated, then Alice confirms the validity of the signature, otherwise it is rejected.

2.4.2 ElGamal DSS

According to [15], ElGamal digital signature is the asymmetric approach of verification technique based on discrete logarithm problem over a finite field. This mechanism has the following parameters:

- g - a random number that serves as the generator;
- q - a prime number;
- $H()$ - is the hash function.

The algorithm is summarized as follows:

1. In the *KeyGen* algorithm, Bob randomly chooses a secret key sk with $1 < sk < q - 2$, and then computes the public key pk using sk, i.e. $pk = g^{sk}(\text{mod } q)$;

   Bob performs the following steps in order to sign a message M;

2. Bob chooses a random integer x such that $1 < x < q - 2$ and *gcd*$(x, q - 1)$ = 1, where *gcd* refers to the greatest common divisor;

3. Thus, at this stage, Bob computes $K_1 = g^x (\text{mod } q)$;

4. Bob computes $K_2 = (H(M) - sk.K_1)^{-1} (\text{mod } q - 1)$;

   The pair (K_1, K_2) is the digital signature of M. This signature is verified by Alice as follows:

5. Alice chooses $0 < K_1 < q$ and $0 < K_2 < q - 1$;

6. Then, Alice computes $g^{H(M)} = pk^{K_1} K_1^{K_2}$ (mod q).

Alice accepts the signature if all the conditions are satisfied, and rejects it otherwise.

2.4.3 Digital Signature Algorithm (DSA)

Davtyan et al. [9] describe this algorithm thoroughly in their work. The algorithm is generated using the parameters as follows:

p - a prime modulus;

q - a prime divisor of $(p - 1)$;

g - a generator of the sub group of order q (mod p);

x - the private key;

y - the public-key obtained through $y = g^x$ (mod p);

k - the per message secret key obtained randomly from the range $1 < k < q$.

In the *KeyGen* algorithm, Bob chooses a secret key x, where $1 < x < q$. Bob then calculates the public key $y = g^x$ (mod p).

The signature of message M consists of a pair of numbers r and s determined using:

$r = g^k$ (mod p) mod q;

$s = (k^{-1}(H(M) + xr))$ (mod q).

Thus, the signature (r, s) is produced. Bob transmits message M and (r, s) to Alice, who verifies the signature by performing the following.

1. Alice checks that $1 < r < q$ and $1 < s < q$; the signature is rejected if one of these fails.
2. If both the conditions above are satisfied, Alice computes:

   $w = s^{-1}$(mod q);

   $u1 = H(M) \cdot w$ (mod q);

   $u2 = r \cdot w$ (mod q);

   $v = ((g)^{u1}(y)^{u2})$ (mod p) mod q.
3. If $v = r$, then the signature is accepted otherwise it is rejected.

2.4.4 Elliptic Curve ElGamal (EC ElGamal) Digital Signature Algorithm

The concept of Elliptic Curve Cryptography has been embedded into the ElGamal Digital signature algorithm to produce EC ElGamal Digital Signature Scheme as shown in [7]. The sender Bob, denoted here as *B*, selects a random integer k_B from the interval $[0, q-1]$ as the private key and computes the public key, *B* = $k_B G$. The signing procedure is given in the following steps.

1. Choose a random integer k from the interval $[0, q-1]$.
2. Compute $R = kG = (x_R, y_R)$, where $r = x_R(\text{mod } q)$ if *r* = 0 goto step 1.
3. Compute $e = H(M)$, where H is the hash function $\{0, 1\}^* \longrightarrow \mathbb{F}_q$.
4. Compute $s = k^{-1}(e + rk_B)\text{mod } q$; if then goto step 1.

The signature of message M is (R, s). Bob now sends the signature and the message to Alice for verification. It must be verified that s is an integer in $[0, q-1]$ and $R = (x_R, y_R) \in E(\mathbb{F}_q)$. Alice performs the following to verify the signature.

1. Compute $V_1 = sR$.
2. Compute $V_2 = h(M)G + rA$, where $r = x_R$.
3. If $V_1 = V_2$, then the signature is accepted by Alice, else declared as invalid.

Due to Shor's algorithm [49], the IFP and the DLP over a finite field are broken on a quantum computer. Furthermore, ECDLP is also broken on a quantum computer [49]. The design of quantum computers and its implementation to commercial use is fast approaching in the very near future. Hence, there is a necessity to develop a system which can sustain on a classical computer but it should be quantum attack resistent.

Chapter 3

Lattices

3.1 Background of Lattices

Lattices have been widely studied in the past two decades by various cryptologists. The concept of lattices was first studied in the late 18th century by the mathematicians Joseph Louis Lagrange, Carl Friedrich Gauss, and later Minkowski as mentioned in [37]. The concept of lattice is applied in various fields but its essential application to cryptography was initiated by the work of Ajtai in [2], whose work involved the construction of lattice based cryptographic system which constitutes mathematical problems which are hard to solve.

Lattice-based cryptography is based on the construction of complicated mathematical algorithms using cryptographic parameters involving lattices. The main objective here is to enhance data security. Recently, lattice-based schemes have become very popular, as well as leading the way towards the post-quantum cryptography as revealed from studies in [37]. Unlike other famous and widely used public-key cryptographic schemes such as the RSA or Diffie-Hellman cryptosystems, which are vulnerable and can be easily attacked by quantum computer based algorithms, some lattice-based constructions appear to be resistant to attacks by both classical and quantum computers. Furthermore, many such lattice-based constructions are proven to be secure under the assumption that certain well-studied computational lattice problems cannot be solved efficiently.

A Lattice, $\mathcal{L}$, is defined as follows.

Definition 3.1.1. The m-dimensional Euclidean space is denoted $\mathbb{R}^m$. A $lattice$ in $\mathbb{R}^m$ is the set of all integer combinations $\mathcal{L} = \{\sum_{i=1}^{n} x_i b : x_i \in \mathbb{Z}\}$ of n linearly independent vectors $b_1, ..., b_n$ in $\mathbb{R}^m (m \geq n)$. The set of vectors $b_1, ..., b_n$ is said to form a basis of the lattice, and the integer n is called the $rank$ of the lattice. A basis can be compactly represented by the matrix $\mathbf{B} = [b_1|...|b_n] \in \mathbb{R}^{m \times n}$ having the basis vectors as columns. The lattice generated by **B** is denoted $\mathcal{L}(\mathbf{B})$.

The product **Bv** is the usual matrix-vector multiplication. Thus, we have $\mathcal{L}(\mathbf{B}) = \{\mathbf{Bv} : \mathbf{v} \in \mathbb{Z}^n\}$. $\mathbb{Z}^n$ is a lattice, which is generated by the standard orthonormal basis for $\mathbb{R}^n$. In addition, the basis for a lattice is not unique. The value of the norm of a vector is basis dependent. The Euclidean norm depends on vector components, which further rely on the choice of basis.

Definition 3.1.2. For any $j \geq 1$, the *norm of a vector* $v \in \mathbb{R}^n$ is defined as

$$\parallel \boldsymbol{v} \parallel_j = \sqrt[j]{\sum \boldsymbol{v}_i}$$

The minimum distance of a lattice, $D(\mathcal{L})$, is the minimum distance between any two distinct lattice points and is equivalent to the length of the shortest nonzero lattice vector, as given by the following definition:

Definition 3.1.3. The *minimum distance of a lattice*, $D(\mathcal{L})$, is defined as

$$D(\mathcal{L}) = min \parallel x - y \parallel : x \neq y \in \mathcal{L} = min \parallel x \parallel : x \in \mathcal{L}, x \neq 0.$$

When discussing computational issues related to lattices, it is usually assumed that the lattices are represented by a basis matrix B whereby B has entries which are integers.

Lattices are the objects that are more geometric in nature. Hence, it can be more thoroughly described as the set of intersection points of an infinite, regular n-dimensional grid as shown in [31]. The numerous flexible characteristics of lattices makes them quite a valuable concept that can be used to attack many significant problems in mathematics and computer science. In particular, lattices have been used to solve integer programming with finitely many variables [33], factorization of polynomials over the integers [49], low density subset-sum problems [26], and many other cryptanalytic problems [37]. Lattices are discrete additive subgroups in vector spaces. They are important for us, since the embedding of all rings and ideals that we consider are actual n-dimensional lattices in $\mathbb{R}^n$.

Definition 3.1.4. An *n-dimensional lattice* $\mathcal{L}$ is any subset of $\mathbb{R}^n$ that is both:

1. an additive subgroup: $0 \in \mathcal{L}$, and $-x, x + y \in \mathcal{L}$ for every $x, y \in \mathcal{L}$; and

2. discrete: every $x \in \mathcal{L}$ has a neighborhood in $\mathbb{R}^n$ in which x is the only lattice point.

There have been many successful and effective applications of lattice problems in cryptography. As discussed by Peikert et al. in [37], the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP) are amongst the most popular versions of lattice problems. These problems are thought to be hard to solve efficiently, even with a quantum computer. In cryptographic constructions, a hard problem is usually given due respect, and is always considered to be secure.

Lattice-based cryptographic constructions have tremendous potential as researchers keep on exploring cryptographic algorithms and its contribution towards secured signature schemes. The main purpose of lattice cryptography is to protect data transmitted in the likely presence of an attacker. This is usually done by employing hard cryptographic algorithms in order to strengthen security. If there exists an algorithm that can efficiently break the cryptographic scheme with non-negligible probability, then there exists an efficient algorithm that solves a certain lattice problem on any input.

Cryptographic algorithms that are complicated rely on sophisticated techniques from which inputs and entries can be retrieved. Thus, we need to employ a feasible sampling technique that will guarantee security of our scheme. Precaution must be taken that the signatures will not leak the private keys.

3.2 Sampling

We will use the technique of rejection sampling [10], which is one of the many techniques to generate samples from a distribution. Using this technique, given one distribution, appropriate samples can be evidently drawn from another distribution by applying some rejection/ acceptance rule to the sample.

Rejection sampling is employed in our scheme mainly because it guarantees that the uniform distribution $\mathbb{Z}_q^{m \times n}$ is independent of the secret.

Definition 3.2.1. Let $X : \Omega \longrightarrow \mathbb{R}$ be a random variable. The *probability density function* of X, denoted by $f_X(x)$, is defined to be $f_X(x) = Pr(X = x)$.

In other words, $f_X(x)$ is the probability that X takes on the value x.

Lemma 3.2.1. *Let $f : \mathbb{Z}^n \longrightarrow \mathbb{R}$ be a probability distribution. Given a subset $V \subseteq \mathbb{Z}^n$, let $h : V \longrightarrow \mathbb{R}$ be a probability distribution defined on V. Let $g_v : \mathbb{Z}^n \longrightarrow \mathbb{R}$ be a family of probability distributions indexed by $v \in V$ such that for almost all v's from h there exists a universal upper bound $U \in \mathbb{R}$ such that the success probability is $Pr[Ug_v(T) \geq f(T); T \longleftarrow f] \leq 1$.*

Then the output distributions of the following two algorithms have negligible statistical difference:

1. $v \longleftarrow h, T \longleftarrow g_v, \mathrm{output}(T, v)$ *with probability* $\min(\frac{f(T)}{Ug_vT}, 1)$. *Else fail.*
2. $v \longleftarrow h, T \longleftarrow f, \mathrm{output}(T, v)$ *with probability* $\frac{1}{U}$.

In the following section, we recall the hardness assumption problem over lattices.

3.3 Hardness Problems of Lattices

In cryptography, a notable objective is to make cryptographic primitives with provable security. Provable security refers to any type of security that can be proved. This normally refers to mathematical proof which is quite common in cryptography. However, as information theoretic security cannot always be achieved, cryptographers have to rely on computational security to be assured that these systems are generally secure. Because hardness of a problem is difficult to prove, in practice, certain problems are "assumed" to be difficult [40].

Over the past couple of decades, the attempts of cryptographic researchers have revealed that the hardness of the Lattice-based schemes is built upon the following problems:

1. Short Vector Problem (SVP)
2. Closest Vector Problem (CVP)
3. Learning With Errors Problem (LWE)

3.3.1 Short Vector Problem

In order to construct strong and secure post-quantum signature schemes, it is essential to establish computational problems, whose hardness can be utilized as a weapon to boost their security. Furthermore, the cryptographic algorithms used in such schemes must remain difficult even in the presence of quantum computers. One suitable contender for this concept is the problem of approximating short vectors in a lattice (SVP). The quantum-hardness of the short vector problem was analyzed by Ludwig [27] and Regev [42]. Through their studies, they both concluded that the computational advantage that will be attained with quantum computers is marginal. There are numerous cryptographic schemes whose security relies heavily on the intractability of the SVP in lattices of sufficiently large dimension [19, 23].

SVP is the most famous and widely studied computational problem on lattices. The major objective in SVP is that given a lattice, $\mathcal{L}$, which is usually represented by a basis, one has to find the shortest nonzero vector in $\mathcal{L}$. The problem can be defined with respect to any norm, however the Euclidean norm is the most common. As discussed in [47], a variant of SVP was commonly studied in computational complexity theory, where it only asks to compute the length of the shortest nonzero vector in $\mathcal{L}$ without necessarily solving for the vector.

With regards to the Euclidean norm, the hardness of SVP was theorized by van Emde Boas in [50]. The theory remained wide open until 1998, when Ajtai proved in [2] that SVP is hard to solve exactly under randomized reductions. The strongest hardness result for SVP known to date is due to Micciancio [31].

To compute the length of the shortest vector in a lattice, no efficient algorithm in polynomial time is known to date. In spite of a number of attempts, researchers and analysts actually have not been able to find the shortest vector in a lattice [32]. Undoubtedly, failure to do this is an advantage from the cryptographic point of view, whereby the hardness of approximating SVP within certain polynomial factors can be used as an essential ingredient for the construction of provably secure cryptographic functions.

Definition 3.3.1. Let a lattice basis be given as $\mathcal{L}(B)$. Then, the *Shortest Vector Problem* is the problem of finding a vector **v** $\in \mathcal{L}(B)$ such that $\|$ **v** $\|_2$ is minimal.

In the SVP, the aim is to find the shortest nonzero vector from a given lattice. This simply refers to the intersection point in the grid closest to the origin.

3.3.2 Closest Vector Problem

The Closest Vector Problem (CVP) is a computational problem on lattices. Given a lattice $\mathcal{L}$ and a target point $\vec{x}$, the CVP asks to find the lattice point closest to the target. This problem is usually defined with respect to the Euclidean norm. Mostly in complexity theory, a less complicated version of the problem only asks to compute just the distance of the target from the lattice without actually finding the closest lattice vector.

Majority of the practical versions of the CVP only asks to find a lattice vector that is as close as possible to the target, even if not necessarily the closest. According to [4], g-approximation algorithm for the CVP finds a lattice vector within a distance at most g times the distance of the optimal solution. The best known polynomial time algorithms to solve the CVP due to Babai [4] are based on lattice reduction, and achieve approximation factors that are essentially exponential in the dimension of the lattice.

CVP is widely regarded, both in theory and in practice, as a considerably harder problem than SVP.

Definition 3.3.2. Let a lattice basis be given as $\mathcal{L}(B)$. Then, the *Closest Vector Problem* is the problem that given a vector **u** $\notin \mathcal{L}(B)$ find a vector **v** $\in \mathcal{L}(B)$ such that $\|$ **v - u** $\|_2$ is minimal.

In CVP, the objective is that, given a lattice and a target point which may not necessarily be in the lattice, find the lattice point closest to the target.

3.3.3 The Learning With Errors Problem (LWE)

Recently, the LWE, which was first introduced in [44], has established itself in the field of cryptography. Over the years, it has become a highly flexible and adaptable basis for cryptographic constructions. The LWE has attained its position amongst the highest level with regards to providing cryptographic security. As claimed in [34], just like many other Lattice-based problems, LWE problem is very hard to solve. It is as hard as worst-case lattice problems, hence furnishing all cryptographic constructions based on it secure under the assumption that worst-case lattice problems are hard.

The main parameters in this problem are the positive integers $n, q \in \mathbb{N}$. The problem is based on χ which is the uniform distribution on $\mathbb{Z}_q$ and $\phi = D_{\alpha q}$ for some fixed real number $0 < \alpha < 1$.

Definition 3.3.3. *[3]* Let $n, q \in \mathbb{N}$ and let χ and ϕ be distributions on $\mathbb{Z}$. The *LWE distribution* for a given vector $s \in \mathbb{Z}_q^n$ is the set of pairs $(a, a \cdot s + e(\text{mod } q))$ where $a \in \mathbb{Z}_q^n$ is sampled uniformly and where e is sampled from ϕ.

- The computational-LWE problem is: For a vector $s \longleftarrow \chi^n$ and given arbitrarily many samples from the LWE distribution for s, to compute s.

- The decisional-LWE problem is: Given arbitrarily many samples from $\mathbb{Z}_q^n$ to distinguish whether the samples are distributed uniformly or whether they are distributed as the LWE distribution for some fixed vector $s \longleftarrow \chi^n$.

In the computational version, to be successful, an attacker has to find $s \in \mathbb{Z}_q^n$ given $a \in \mathbb{Z}_q^{n \times m}$. This implies that any random and efficient distinguisher between LWE and uniform distributions may be utilized to recover the secret key. The essential concept of small error distribution has been vigorously studied in [14] and [34]. In these studies, it is chosen independently and identically from a Gaussian-like distribution. The standard deviation is taken as $\sigma = \alpha q$.

Moreover, a quantum reduction was shown by Regev in [43] signifying LWE problem is as hard in the average-case as approximating lattice problems in the worst-case with an approximation factor $\tilde{O}(n/\alpha)$ and $\alpha q \geq 2\sqrt{n}$.

Theorem 3.3.1. *[44] Let $n, q \in \mathbb{N}$ and $0 < \alpha < 1$ be such that $\alpha q \geq 2\sqrt{n}$. Then there exists a quantum reduction from worst-case $SVP_{\tilde{O}(n/\alpha)}$ to (n, q, α)- LWE.*

Here, a relatively small error distribution will lead to complicated situations, meaning, for all vectors that will probably be sampled as $s \longleftarrow \chi^n$, the LWE distribution is not statistically close to the uniform distribution. There exists a unique solution s that is probably used to generate samples from the LWE distribution.

Furthermore, Regev's main theorem [44] suggests that the LWE problems are as hard as worst-case assumptions in general lattices when χ is the uniform distribution and when ϕ is a discrete Gaussian with standard deviation $\sigma = \alpha q$ for some fixed real number $0 < \alpha < 1$.

Chapter 4

Proposed Signature Scheme and Analysis

4.1 Proposed Signature Scheme

The design of the proposed scheme is based on providing a variant of Dilithium technique by Ducas et al. [12]. Dilithium is a digital signature scheme that is strongly secure under chosen message attacks based on the hardness of lattice problems over module lattices.The security aspect of this scheme is quite strong, whereby an attacker who has access to a signing oracle is unable to generate the signature of a message whose signature he has not seen before, nor produced a different signature of a message that he already saw signed.

The proposed signature scheme is described in Algorithms 1, 2, and 3.

In the key generation algorithm, the secret key is notated as $(\mathbf{S}, \mathbf{E}, \mathbf{T})$ and the public key is $(\mathbf{T}_1, \mathbf{A})$. The key generation begins with choosing random keys from some uniform distribution, χ, in the LWE assumption, which is expanded into a matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$. $\mathbf{A}$ is an $m \times n$ matrix which is uniformly chosen from $\mathbb{Z}_q^{m\times n}$. $\mathbf{D}_{\sigma_S}$ and $\mathbf{D}_{\sigma_E}$ are the distributions for the secret and error, respectively, in the LWE assumption. $\mathbf{D}_{\sigma_S}$ and $\mathbf{D}_{\sigma_E}$ are discrete Gaussians with standard deviation $\sigma_S = \sigma_E \geq 2\sqrt{n}$ as shown in [42]. The scheme is designed as such that we have $m > n$ and $q > 2^d$. The main security parameter is n. This parameter is quite significant as the security of our scheme relies heavily on

Algorithm 1 *KeyGen()*

```
Input : n, m, q, $\sigma_S$, $\sigma_E$
1.    $\mathbf{A} \longleftarrow \mathbb{Z}_q^{m\times n}$
2.    $\mathbf{S} \longleftarrow \mathbf{D}_{\sigma_S}^{n\times 1}$
3.    $\mathbf{E} \longleftarrow \mathbf{D}_{\sigma_E}^{m\times 1}$
4.    $\mathbf{T} = \mathbf{AS} + \mathbf{E}$
5.    $\mathbf{T}_1 := Power2Round_q(\mathbf{T}, d)$
6.    return$pk = (\mathbf{A}, \mathbf{T}_1),\ sk = (\mathbf{S}, \mathbf{E}, \mathbf{T})$
```

n, as well as q. Our work is inspired by the LWE problem. The LWE distribution is on pairs (**A**, **AS** + **E** (mod q)), where **S** and **E** are matrices. $\mathbf{S} \in \mathbb{Z}_q^{n\times 1}$ and $\mathbf{E} \in \mathbb{Z}_q^{m\times 1}$ are chosen to have entries sampled independently from the distributions $\mathbf{D}_{\sigma_S}$ and $\mathbf{D}_{\sigma_E}$. Matrix $\mathbf{A} \in \mathbb{Z}_q^{m\times n}$ is a master public key, thus is shared across all instances.

As shown in Theorem 3.3.1, we recommend that the parameters n, q, and α are chosen in a manner such that $\sigma = \alpha q > 2\sqrt{n}$, where $0 < \alpha < 1$. Further, m should be chosen such that $q^m > q^n(2E+1)^n$. This condition will ensure that there will be a unique solution S that could be used to generate the required samples from the LWE distribution.

The main feature of the proposed scheme is to have a public key of the form

$$\mathbf{T}_1 = Power2Round_q(\mathbf{T}, d),$$

where $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ and **A** is an $m \times n$ matrix whose entries are from $\mathbb{Z}_q^{m\times n}$. The private keys **S** and **E** are selected to have entries sampled independently from the distributions $\mathbf{D}_{\sigma_S}$ and $\mathbf{D}_{\sigma_E}$. Thus, in Step 4, the value of the function $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ is then computed.

The approach here is to hash and sign. As the signing algorithm commences, the entire function $\mathbf{T} = \mathbf{AS} + \mathbf{E}$ is split into $\mathbf{T}_1$ and $\mathbf{T}_0$ such that $\mathbf{T}_1 \cdot 2^d + \mathbf{T}_0 = \mathbf{T}$ as shown in Lemma 4.1.1.

The signer samples **Y** with coefficients in $\mathbf{D}_{\sigma_Y}^{n\times 1}$ after which, in Step 5, $\mathbf{W} = \mathbf{AY}$ is computed. Next, the signer must decompose **W** to obtain $\mathbf{W}_1$. Note that $\mathbf{W}_0$ here is concealed in the interval $-\delta$ and δ inclusive, and is not revealed. Subsequently, the signer breaks **W** and writes $\mathbf{W} = \mathbf{W}_1 \cdot 2\delta + \mathbf{W}_0$, and computes the hash value $c = H(\mathbf{T}_1, \mathbf{W}_1, \mu)$. The message is denoted by $\mu \in M$. Once the c value has been obtained, the signer

```
Algorithm 2 Sign
Input : $sk = (\mathbf{S}, \mathbf{E}, \mathbf{T}), \mu \in M, \mathbf{D}_{\sigma_Y}^{n\times 1}$
1.    $\mathbf{A} \longleftarrow \mathbb{Z}_q^{m\times n}$
2.    $\mathbf{T}_1 := Power2Round_q(\mathbf{T}, d)$
3.    $\mathbf{T}_0 := \mathbf{T} - \mathbf{T}_1 \cdot 2^d$
4.    $\mathbf{Y} \longleftarrow \mathbf{D}_{\sigma_Y}^{n\times 1}$
5.    $\mathbf{W} := \mathbf{AY}$
6.    $\mathbf{W}_1 := HighBits_q(\mathbf{W}, 2\delta)$
7.    $c := H(\mathbf{T}_1, \mathbf{W}_1, \mu)$
8.    $\mathbf{Z} := \mathbf{Y} + c\mathbf{S}$
9.    $(\lambda_1, \lambda_0) := Decompose_q(\mathbf{W} - c\mathbf{E}, 2\delta)$
10.   if $\|\mathbf{Z}\|_\infty \geq \beta - \tau$ or $\|\lambda_0\|_\infty \geq \delta - \tau$ or $\lambda_1 \neq \mathbf{W}_1$ then go to 4
11.   $\mathbf{Z}_2 := Makehint_q(-c\mathbf{T}_0, \mathbf{W} - c\mathbf{E} + c\mathbf{T}_0, 2\delta)$
12.   if $\|c\mathbf{T}_0\|_\infty \geq \delta$ or the number of 1's in $\mathbf{Z}_2$ is greater than $\omega$, then go to 4
13.   return $\xi := (\mathbf{Z}, \mathbf{Z}_2, c)$
```

then computes $\mathbf{Z} = \mathbf{Y} + c\mathbf{S}$. The rejection sampling technique is used to return $\mathbf{Z}$ with probability $\min\left(\frac{\mathbf{D}_{\mathbf{Z}}^n(\mathbf{Z})}{\mathbf{D}_{\mathbf{Y},\mathbf{S}c}^n(\mathbf{Z})}\right)$. This ensures that $\mathbf{Z}$ is generated in order to protect $\mathbf{S}$.

At this stage, the signing procedure can restart subject to two conditions. Firstly, if some coefficient of $\mathbf{Z}$ is at least $\beta - \tau$, and secondly, if the magnitude of some coefficient of $\lambda_0 = LowBits_q(\mathbf{W} - c\mathbf{E}, 2\delta)$ is at least $\alpha - \tau$. This step is quite an essential part of the scheme. This part ensures that no information about the secret keys $\mathbf{S}$ and $\mathbf{E}$ is leaked. In order to verify correctness, it is important to check and ensure that $\lambda_1 = \mathbf{W}_1$. It must be noted that if $\|c\mathbf{E}\|_\infty \leq \tau$, then $\|\lambda_0\|_\infty$ automatically becomes less than $\delta - \tau$, which implies that $\lambda_1 = \mathbf{W}_1$.

In the event that every one of the checks are successful and a restart is not required, then it can be shown that $HighBits_q(\mathbf{AZ} - c\mathbf{T}, 2\delta) = \mathbf{W}_1$. The proof is shown in Lemma 4.1.2.

Lemma 4.1.1. *To enhance the security of the system, **T**, we break down **T** to $\boldsymbol{T}_1$:*
$\boldsymbol{T}_1 := Power2Round_q(\boldsymbol{T}, d)$.

Proof. As mentioned earlier, q is a prime number, and d is some random positive integer such that it satisfies the condition $q > 2^d$. To break up $\mathbf{T}$:

We have $\frac{T}{2^d} = \mathbf{T}_1$ remainder $\mathbf{T}_0$.

This implies that $\mathbf{T} = \mathbf{T}_0 + \mathbf{T}_1 \cdot 2^d$.

Thus, $\mathbf{T}_1 = \frac{\mathbf{T}-\mathbf{T}_0}{2^d}$. $\square$

Definition 4.1.1. We now define the functions $Decompose_q$, $HighBits_q$ and $LowBits_q$: The procedure $Decompose_q(\boldsymbol{W} - c\boldsymbol{E}, 2\delta)$ breaks down the function $\boldsymbol{W} - c\boldsymbol{E}$ into its components λ_1 and λ_0. Then, we have

$\frac{\boldsymbol{W}-c\boldsymbol{E}}{2\delta}$ = λ_1 *remainder* λ_0,

$\boldsymbol{W} - c\boldsymbol{E}$ = $\lambda_1.2\delta + \lambda_0$,

$\lambda_1 = HighBits_q(\boldsymbol{W} - c\boldsymbol{E}, 2\delta) = \frac{\boldsymbol{W}-c\boldsymbol{E}-\lambda_0}{2\delta}$,

$\lambda_0 = LowBits_q(\boldsymbol{W} - c\boldsymbol{E}, 2\delta) = \boldsymbol{W} - c\boldsymbol{E} - \lambda_1.2\delta$.

The components λ_1 and λ_0 are simply extracted from the function $Decompose_q$ using the functions $HighBits_q$ and $LowBits_q$, respectively.

We employ the cryptographic hash function defined in [12] for our signature scheme.

Definition 4.1.2. A hash function is a function which maps from arbitrary string $\{0,1\}^*$ to $B_{k,w}$, where k is the length of the string and w is the weight of the string. Symbolically, we notate the function as $H : \{0,1\}^* \longrightarrow B_{k,w}$. Concretely, we use the following *:*

```
1. Initialize $c$ = $c_0 c_1 ... c_{255} = 00...0$
2. for i := 196 to 255 do
3. $j \longleftarrow \{0, 1, ..., i\}$
4. $s \longleftarrow \{0, 1\}$
5. $c_i := c_j$
6. $c_j := (-1)^s$
7. end for
```

```
8. return $c$
```

Thus, a random 256-element array with 60$\pm$ 1's and 196 0's will be generated.

Definition 4.1.3. We refer to [12] for the procedure $Makehint_q$ $(-c\boldsymbol{T}_0, \boldsymbol{W} - c\boldsymbol{E} + c\boldsymbol{T}_0, 2\delta)$, which is defined as follows:

```
1. $\lambda_1 := HighBits_q(\boldsymbol{W} - c\boldsymbol{E} + c\boldsymbol{T}_0, 2\delta)$
2. $v_1 := HighBits_q(\boldsymbol{W} - c\boldsymbol{E}, 2\delta)$
3. if $\lambda_1 = v_1$ then
4. return $0$
5. else
6. return $1$
7. end if
```

Definition 4.1.4. We again refer to [12] for the procedure $UseHint_q(\boldsymbol{Z}_2, \boldsymbol{AZ} - c\boldsymbol{T}_1 \cdot 2^d, 2\delta)$, which is defined as follows:

```
1. let $m := (q-1)/2\delta$
2. $(\lambda_1, \lambda_0)$:= $Decompose_q(\boldsymbol{AZ} - c\boldsymbol{T}_1 \cdot 2^d, 2\delta)$
3. if $\boldsymbol{Z}_2 = 1$ and $\lambda_0 > 0$ then
4. return $(\lambda_1 + 1)$ mod$^+ m$
5. else if $\boldsymbol{Z}_2 = 1$ and $\lambda_0 \leq 0$ then
6. return $(\lambda_1 - 1)$ mod$^+ m$
7. else
8. return $\lambda_1$
9. end if
```

```
Algorithm 3 Verify
Input : $pk = (\mathbf{T}_1), \mu \in M, \xi := (\mathbf{Z}, \mathbf{Z}_2, c)$
1.    $\mathbf{A} \longleftarrow \mathbb{Z}_q^{m \times n}$
2.    $\mathbf{W}_1 := UseHint_q(\mathbf{Z}_2, \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d, 2\delta)$
3.    if $c = \mathbf{H}(\mathbf{T}_1, \mathbf{W}_1, \mu)$ and $\|\mathbf{Z}\|_\infty \geq \beta - \tau$ and the number of 1's in $\mathbf{Z}_2$ is $\leq \omega$ then
4.    return 1
5.    else
6.    return 0
7.    end if
```

In the verification algorithm, the signature and the public key are used to regenerate $\mathbf{W}_1$. In line 3, it also analyzes that c = H$(\mathbf{T}_1, \mathbf{W}_1, \mu)$. It examines all the coefficients of **Z** and also scrutinizes all the number of 1's in $\mathbf{Z}_2$, which is decided by the values of $c\mathbf{T}_o$ that cause a carry to occur. Hence, the verification algorithm will accept if the $\mathbf{W}_1$ calculated by the verifier is the same as that of the signer.

Lemma 4.1.2. *To prove the correctness of our signature scheme, we claim that* $HighBits_q(\boldsymbol{AZ} - c\boldsymbol{T}, 2\delta) = \boldsymbol{W}_1$.

Proof. We refer to [12] for the proof. Since $\mathbf{W} = \mathbf{AY}$ and $\mathbf{T} = \mathbf{AS} + \mathbf{E}$, we have $\mathbf{W} - c\mathbf{E} = \mathbf{AY} - c\mathbf{E} = \mathbf{A}(\mathbf{Z} - c\mathbf{S}) - c\mathbf{E} = \mathbf{AZ} - c\mathbf{T}$. Furthermore, $\mathbf{W} - c\mathbf{E} + c\mathbf{T}_0 = \mathbf{T} - \mathbf{T}_1 \cdot 2^d$. Thus, while verifying, the verifier will compute $UseHint_q(\mathbf{Z}_2, \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d, 2\delta) = HighBits_q(\mathbf{AZ} - c\mathbf{T}, 2\delta)$.

In addition, the signer must also check that $\lambda_1 = \mathbf{W}_1$, which is obviously equivalent to $Highbits_q(\mathbf{W} - c\mathbf{E}, 2\delta) = HighBits_q(\mathbf{W}, 2\delta)$. As a result, the verification algorithm will always accept because the $\mathbf{W}_1$ computed by the verifier is exactly same as the one obtained by the signer. □

4.2 Security Analysis

In this section, we provide security analysis of the scheme. The scheme can be computationally secure.

4.2.1 Security of the Secret Key

In order for our scheme to be secure, the minimum requirement is that the security of the secret key is given the highest priority. An attacker has access to the public key of the signer. The determination of the secret key from the public key must be impossible at all times.

Our proposed signature scheme involves public keys $(\mathbf{T}_1, \mathbf{A})$ and secret keys $(\mathbf{T}, \mathbf{S}, \mathbf{E})$, and consists of mathematical algorithms that use these keys. A very interesting characteristic of our verification algorithm is that it does not have any idea about the secret keys while performing the verification process. The verification algorithm has access only to the public key. We have taken extreme precaution while constructing our scheme, assuring that the owner of the secret key should be able to produce valid signature, however knowledge of the public key must not reveal the secret key.

Two general conditions that guarantee the security of the secret key and our proposed scheme are :

1. Given a public key $(\mathbf{T}_1, \mathbf{A})$, an attacker cannot feasibly determine the secret key $(\mathbf{T}, \mathbf{S}, \mathbf{E})$. Moreover, the attacker must not be able to negotiate any other secret key that can produce the same signatures.

2. Given a public key $(\mathbf{T}_1, \mathbf{A})$ and a list of signed messages $\mu_1, \mu_2, ..., \mu_t$ with their signatures $\xi_1, \xi_2, ..., \xi_t$, an attacker is not able to conceivably obtain a valid signature on any message μ that is not in the list $\mu_1, \mu_2, ..., \mu_t$.

According to condition 2, an attacker can generate as many message and signature pairs as desired, since these can be created using the common public key. However, a digital signature scheme reveals a new message/signature pair each time a new message is signed. This provides new information to the attacker. Condition 2 further says that the attacker gains nothing else apart from the knowledge of that new pair. An attack on a digital signature scheme that makes use of a large number of known signatures is called a transcript attack.

4.2.2 Brute Force Attack

Brute Force Attack is a trial and error method used by attackers to exploit and decode encrypted data, through a vigorous and exhaustive approach rather than applying some intellectual strategies.

In cryptography, this method can be used against any encrypted data. A Brute Force Attack simply uses the search algorithm. This approach usually becomes necessary when it is not possible to take advantage of other weaknesses in an encryption system that would make the task easier. It involves systematically checking all possible keys until the correct key is found. In the worst case, this would involve browsing through the entire search space. The size of the key used in the scheme determines the feasibility and performance of a brute-force attack. Larger key lengths would be more harder to crack than the shorter ones.

For our scheme, we suggest a 256-bit secret key which requires 2^{256} operations to break the scheme via brute force attack. It can take a total of 10^{57} years to do so.

4.2.3 Security Proof against Forgery

Assume that the attacker obtains the key **T = AS + E**. This assumption is of a disadvantage to the attacker because in our proposed scheme, he only has access to $\mathbf{T}_1$, thus in actual practice, our scheme will be more difficult to exploit.

The underlying goal of any attacker is forgery. We consider the case where the attacker attempts to produce an alternate message which did not originate with the sender. The attacker is allowed a chosen message to attack in the process of trying to produce forgeries. Therefore, in this situation, our objective is to establish a measure of insecurity against forgery under chosen message attack for our scheme.

The attacker's actions can be divided into two phases. The first is a learning phase where it is provided access to the oracle from which both our public keys and secret keys were randomly sampled according to Key Generation. It can interrogate this oracle up to q times, in any way it pleases, as long as all the queries are messages in the underlying message space. Once the attacker has cleared this phase, it enters a forgery phase, in which it attempts to output a message.

Lemma 4.2.1. *In order to forge a signature, one must find $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3$ such that $\| \boldsymbol{v}_1 \|_\infty \leq 2\beta$, $\| \boldsymbol{v}_2 \|_\infty \leq 4\delta + 2$, and $\| \boldsymbol{v}_3 \|_\infty \leq 2$, and obtain $\boldsymbol{A}\boldsymbol{v}_1 + \boldsymbol{v}_2 = \boldsymbol{v}_3 \boldsymbol{T}_1 . 2^d$ with $\boldsymbol{v}_1, \boldsymbol{v}_2, \boldsymbol{v}_3 \neq 0$. Moreover, $\boldsymbol{v}_2$ has at most 2ω coefficients of absolute value greater than 2δ.*

Proof. We refer to the analysis in [12]. For a message μ, if the attacker produces a valid signature $(\mathbf{Z}, \mathbf{Z}_2, c)$, then he must have already queried

$$H(\mathbf{T}_1, \mathbf{W}_1, \mu) = c,$$

where $\mathbf{W}_1 = UseHint_q(\mathbf{Z}_2, \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d, 2\delta)$. Since c was computed here, there exists $(\mathbf{Z}', \mathbf{Z}'_2, c)$ and $\mathbf{W}'_1$ for a message μ' such that $H(\mathbf{T}_1, \mathbf{W}'_1, \mu') = c = H(\mathbf{T}_1, \mathbf{W}_1, \mu)$. We now have $\mu = \mu'$ and $\mathbf{W}_1 = \mathbf{W}'_1$, therefore it must be that

$$\mathbf{W}_1 = UseHint_q(\mathbf{Z}_2, \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d, 2\delta),$$

$$\mathbf{W}_1 = UseHint_q(\mathbf{Z}'_2, \mathbf{AZ}' - c\mathbf{T}_1 \cdot 2^d, 2\delta).$$

Due to rejection sampling technique, it can be assured that $\mathbf{Z} \neq \mathbf{Z}'$. Now, we have

$$\| \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d - \mathbf{W}_1, 2\delta \|_\infty \leq 2\delta + 1,$$

$$\| \mathbf{AZ}' - c\mathbf{T}_1 \cdot 2^d - \mathbf{W}_1, 2\delta \|_\infty \leq 2\delta + 1.$$

By the triangular inequality, this implies that

$$\mathbf{A}(\mathbf{Z} - \mathbf{Z})' + \mathbf{v} = \mathbf{0}$$

for some $\mathbf{v}$ such that $\| \mathbf{v} \|_\infty \leq 4\delta + 2$, where $\mathbf{Z} - \mathbf{Z}' \neq 0$. Thus, from our signing algorithm, we know that except ω elements, the rest of the elements in $\mathbf{Z}_1$ and $\mathbf{Z}'_1$ are equal to 0. According to [12], all but ω coefficients of $\mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d - \mathbf{W}_1, 2\delta$ and $\mathbf{AZ}' - c\mathbf{T}_1 \cdot 2^d - \mathbf{W}_1, 2\delta$ are less than δ. As a result, all but 2ω coefficients of **v** are less than 2δ. $\square$

Our scheme is secure as even after such an attack, the attacker will encounter a very low probability of producing forgeries. This is because firstly, our hash function, $c = \mathbf{H}(\mathbf{T}_1, \mathbf{W}_1, \mu)$, is a one-way function. This means that computing an inverse image for a given image is intractable. Such a hash function prevents forgery since even if it is possible to generate a valid signature for this hash value, it is impossible to find a document with the same hash value. And secondly, to ensure security of our scheme against forgery, the q-value selected should be sufficiently large, as this makes sure that forgery becomes near to impossible.

4.2.4 Other Possible Attacks

The main objective of an attacker in our scheme is to determine **S** and **E** given $\mathbf{T}_1$ and **A**. We show two such possible attacks and their consequences below.

1. The attacker performs a statistical analysis of the values $\mathbf{T}_1$ and **A** to figure out the values of **S** and **E**. He can even do an analysis and obtain the pair $(\mathbf{Z}, c)$, and attempt to "average" the **Z** and acquire **S**, to achieve this. Another option is to use the fact that $Decompose_q(\mathbf{W}, 2\delta) = Decompose_q(\mathbf{W} - c\mathbf{E}, 2\delta)$ in order to attain some knowledge about **E**. However, for our scheme, since the distribution of the values $(\mathbf{Z}, c)$ is independent of the secret keys, all attacks of this nature are unsuccessful.

2. Another possibility is that the attacker is successful in negotiating a signature pair $(\mathbf{Z}, c)$ on message μ. Being in possession of his own function c' and hoping that $c' = c$, he then proceeds to determine a second message μ' and attempt to verify such that

$$Highbits_q(\mathbf{AZ} - c\mathbf{T}_1, 2\delta) = \mathbf{W}_1 = \mathbf{W}_1' = Highbits_q(\mathbf{AZ}' - c'\mathbf{T}_1, 2\delta).$$

This means that the attacker is negotiating with our scheme and computing a second pre-image of our hash function. However, due to our collision resistent hash function, this forgery attempt will also result as unsuccessful.

Nowadays, hackers use computer programs and software that runs automatically to get the encryption key or password. We suggest that the best approach to counter this attack is to apply the CAPTCHA Code Use (Completely Automated Public Turing test to tell Computers and Humans Apart). According to [51], CAPTCHA is a type of challenge - response test used in computing to determine whether or not the user is human. A CAPTCHA code is a technique by which we recognize a computer or a human, by making some questions or images or numbers, the answer of which is not submitted by the computer automatically.

4.3 Performance Analysis - Time Complexity

The time complexity is one of the most important measures for cryptographic algorithm development. The most commonly utilized measure is the big $\mathcal{O}$ notation. The algorithms in our scheme heavily involve multiplication, thus will have a running time of $\mathcal{O}(n^2)$ as generally described in [36]. The maximum running time over the input parameters are usually considered in the worst case. In the literature [13] and [37], lattice-based cryptosystems are considered to be efficient and feasible. Furthermore, the performance analysis for the scheme to be carried for concrete parameters.

4.4 Summary

Our proposed signature scheme begins with Algorithm 1. The public keys are $(\mathbf{T}_1, \mathbf{A})$, while the secret keys are **T, S** and **E**. These keys are matrices sampled from a uniform distribution and Gaussian distribution. The public key of our scheme is the function $\mathbf{T} = \mathbf{AS} + \mathbf{E}$. In Algorithm 2, in order to enhance the security level, $\mathbf{T}$ is reduced to $\mathbf{T}_1$ and made public. After selecting a vector $\mathbf{Y}$, $\mathbf{W} = \mathbf{AY}$ is computed, and then hashed with message μ to determine c. This c value is used to determine the value of the function $\mathbf{Z} = \mathbf{Y} + c\mathbf{S}$, whose distribution is made independent of the secret using rejection sampling.

In Algorithm 3, the signature is verified by computing $\mathbf{W}_1 = HighBits_q(\mathbf{AZ} - c\mathbf{T}, 2\delta)$ $= UseHint_q(\mathbf{H}, \mathbf{AZ} - c\mathbf{T}_1 \cdot 2^d, 2\delta)$. Furthermore, analysis of the scheme has been discussed.

Chapter 5

Conclusion and Further Work

A Lattice-based digital signature scheme for the Post-quantum world has been designed and presented in this thesis. The algorithms installed in our scheme are secure under chosen message attacks based on the hardness of lattice problems over lattices. High priority has been given to ensure a tight security of the secret key. In addition, the public key has been further reduced into its components, thus also contributing towards the security. Our scheme has been analyzed and proven to be secure against known attacks.

In this thesis, developments in Lattice-based cryptography are discussed. Due to it's elusiveness, lattice-based digital signature schemes are now being considered for real world applications, and are leading contenders in contributing towards enhanced information security.

However, the threat of further attacks must be considered, as lattice based digital signature schemes become more practical and publicly available. Appropriate counter measures must be taken to prevent attacks from breaking signature schemes.

Further research will be needed in Lattice-based cryptography to evaluate the cryptographic algorithms employed by signature schemes, and its effect on enhancing security. Future studies are essential in this area, especially regarding the contribution of parameters used in algorithms, their security analysis and efficiency of the scheme which needs to be verified.

Bibliography

[1] Shweta Agrawal, Dan Boneh, and Xavier Boyen. Efficient lattice (h) ibe in the standard model. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 553–572. Springer, 2010.

[2] Miklós Ajtai. Generating hard instances of lattice problems. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 99–108. ACM, 1996.

[3] Shi Bai and Steven D Galbraith. An improved compression technique for signatures based on learning with errors. In *Cryptographers Track at the RSA Conference*, pages 28–47. Springer, 2014.

[4] Jean-Claude Bajard, Julien Eynard, Nabil Merkiche, and Thomas Plantard. Babai round-off CVP method in RNS: Application to lattice based cryptographic protocols. In *Integrated Circuits (ISIC), 2014 14th International Symposium on*, pages 440–443. IEEE, 2014.

[5] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography Conference*, pages 253–273. Springer, 2011.

[6] Xavier Boyen. Attribute-based functional encryption on lattices. In *Theory of Cryptography*, pages 122–142. Springer, 2013.

[7] Jean-Sébastien Coron. Resistance against differential power analysis for elliptic curve cryptosystems. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 292–302. Springer, 1999.

[8] Ivan Bjerre Damgård. Collision free hash functions and public key signature schemes. In *Workshop on the Theory and Application of of Cryptographic Techniques*, pages 203–216. Springer, 1987.

[9] Seda Davtyan, Aggelos Kiayias, Laurent Michel, Alexander Russell, and Alexander A Shvartsman. Integrity of electronic voting systems: Fallacious use of cryptography. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, pages 1486–1493. ACM, 2012.

[10] Luc Devroye. Sample-based non-uniform random variate generation. In *Proceedings of the 18th conference on Winter simulation*, pages 260–265. ACM, 1986.

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

[12] Léo Ducas, Tancrede Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals–dilithium: Digital signatures from module lattices. *IACR TCHES*, 2018(1):238–268, 2018.

[13] Léo Ducas, Vadim Lyubashevsky, and Thomas Prest. Efficient identity-based encryption over NTRU lattices. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 22–41. Springer, 2014.

[14] Nagarjun C. Dwarakanath and Steven D. Galbraith. Sampling from discrete Gaussians for lattice-based cryptography on a constrained device. *Applicable Algebra in Engineering, Communication and Computing*, **25**(3):159–180, 2014.

[15] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, **31**(4):469–472, 1985.

[16] Pierrick Gaudry. Index calculus for abelian varieties of small dimension and the elliptic curve discrete logarithm problem. *Journal of Symbolic Computation*, **44**(12):1690–1702, 2009.

[17] Pierrick Gaudry. Integer factorization and discrete logarithm problems. In *International Conference on the Theory and Applications of Cryptographic Techniques*, pages 19–34. Springer, 2014.

[18] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford University, 2009.

[19] Oded Goldreich, Shafi Goldwasser, and Shai Halevi. Public-key cryptosystems from lattice reduction problems. In *Annual International Cryptology Conference*, pages 112–131. Springer, 1997.

[20] S. Dov Gordon, Jonathan Katz, and Vinod Vaikuntanathan. A group signature scheme from lattice assumptions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 395–412. Springer, 2010.

[21] Tim Güneysu, Vadim Lyubashevsky, and Thomas Pöppelmann. Practical lattice-based cryptography: A signature scheme for embedded systems. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 530–547. Springer, 2012.

[22] Darrel Hankerson, Alfred J Menezes, and Scott Vanstone. *Guide to elliptic curve cryptography*. Springer Science & Business Media, 2006.

[23] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. Ntru: A ring-based public key cryptosystem. In *International Algorithmic Number Theory Symposium*, pages 267–288. Springer, 1998.

[24] Andrea Kirkby. Cryptography and e-commerce: A Wiley tech brief, 2001.

[25] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of computation*, **48**(177):203–209, 1987.

[26] Jeffrey C. Lagarias and Andrew M. Odlyzko. Solving low-density subset sum problems. *Journal of the ACM (JACM)*, **32**(1):229–246, 1985.

[27] Christoph Ludwig. A faster lattice reduction method using quantum search. In *International Symposium on Algorithms and Computation*, pages 199–208. Springer, 2003.

[28] Vadim Lyubashevsky. Lattice signatures without trapdoors. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 738–755. Springer, 2012.

[29] Vadim Lyubashevsky. Digital signatures based on the hardness of ideal lattice problems in all rings. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 196–214. Springer, 2016.

[30] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 1–23. Springer, 2010.

[31] Daniele Micciancio. The shortest vector in a lattice is hard to approximate to within some constant. *SIAM Journal on Computing*, **30**(6):2008–2035, 2001.

[32] Daniele Micciancio. Generalized compact knapsacks, cyclic lattices, and efficient one-way functions. *Computational Complexity*, **16**(4):365–411, 2007.

[33] Daniele Micciancio and Shafi Goldwasser. *Complexity of lattice problems: a cryptographic perspective*, volume 671. Springer Science & Business Media, 2012.

[34] Daniele Micciancio and Chris Peikert. Hardness of SIS and LWE with small parameters. In *Advances in Cryptology–CRYPTO 2013*, pages 21–39. Springer, 2013.

[35] Victor S Miller. Use of elliptic curves in cryptography. In *Conference on the theory and application of cryptographic techniques*, pages 417–426. Springer, 1985.

[36] Christos H. Papadimitriou. *Computational complexity*. John Wiley and Sons Ltd., 2003.

[37] Chris Peikert. A decade of lattice cryptography. *Foundations and Trends® in Theoretical Computer Science*, **10**(4):283–424, 2016.

[38] Chris Peikert and Alon Rosen. Efficient collision-resistant hashing from worst-case assumptions on cyclic lattices. In *Theory of Cryptography Conference*, pages 145–166. Springer, 2006.

[39] Christophe Petit, Michiel Kosters, and Ange Messeng. Algebraic approaches for the elliptic curve discrete logarithm problem over prime fields. In *IACR International Workshop on Public Key Cryptography*, pages 3–18. Springer, 2016.

[40] Thomas Pöppelmann, Léo Ducas, and Tim Güneysu. Enhanced lattice-based signatures on reconfigurable hardware. In *International Workshop on Cryptographic Hardware and Embedded Systems*, pages 353–370. Springer, 2014.

[41] John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves. *arXiv preprint quant-ph/0301141*, 2003.

[42] Oded Regev. New lattice-based cryptographic constructions. *Journal of the ACM (JACM)*, **51**(6):899–942, 2004.

[43] Oded Regev. Quantum computation and lattice problems. *SIAM Journal on Computing*, **33**(3):738–760, 2004.

[44] Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *Journal of the ACM (JACM)*, **56**(6):34, 2009.

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

[46] Martin Roetteler, Michael Naehrig, Krysta M Svore, and Kristin Lauter. Quantum resource estimates for computing elliptic curve discrete logarithms. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 241–270. Springer, 2017.

[47] Jörg Rothe. Some facets of complexity theory and cryptography: A five-lecture tutorial. *ACM Computing Surveys (CSUR)*, **34**(4):504–549, 2002.

[48] Vagner Schoaba, Felipe Eduardo Gomes Sikansi, and Luiz Castelo Branco. Digital signature for mobile devices: A new implementation and evaluation. *International Journal of Future Generation Communication and Networking*, **4**(2):23–36, 2011.

[49] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Review*, **41**(2):303–332, 1999.

[50] Peter van Emde Boas. Another NP-complete problem and the complexity of computing short vectors in a lattice. *Tecnical Report, Department of Mathmatics, University of Amsterdam*, 1981.

[51] Luis von Ahn, Benjamin Maurer, Colin McMillen, David Abraham, and Manuel Blum. RECAPTCHA: Human-based character recognition via web security measures. *Science*, **321**(5895):1465–1468, 2008.