A usability study of post-quantum algorithms

MARCUS KINDBERG

MASTER´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

A usability study of post-quantum algorithms

Marcus Kindberg
marcus.kindberg@outlook.com

Department of Electrical and Information Technology
Lund University

Principal supervisor: Paul Stankovski

Assistant supervisors (Advenica AB): Niklas Lindskog, Sebastian Mauritsson and Alexander Nilsson

Examiner: Thomas Johansson

June 13, 2017

© 2017
Printed in Sweden
Tryckeriet i E-huset, Lund

Abstract

There is a non-negligible risk that a quantum computer capable of breaking most modern public key encryption will be invented within the next couple of decades. All data that have to stay secret for more than 10-20 years should therefore be encrypted using quantum-resistant algorithms. There are different ways of approaching the problem of quantum security and the currently existing quantum-resistant algorithms for encryption and key exchange can be divided into four categories; Lattice-based, Supersingular elliptic curves, Code-based and Multivariate. The performance of the algorithms in the different categories varies and to evaluate the strengths and weaknesses of each, further study is needed. This thesis provides an overview of algorithms in each category, a comparison of existing implementations of algorithms from the first three categories, and an evaluation of the results. The comparison includes metrics concerning the performance, implementation and security of each algorithm.

All of the considered categories have both advantages and disadvantages and, to be able to choose the right one, the requirements of the application must be considered. Overall, however, the lattice-based algorithms seem to provide the best trade-off between speed, key size and memory consumption, and are relatively easy to implement.

Table of Contents

Chapter 1

Introduction

The security of most public key cryptography used today depends on the hardness of two mathematical problems; integer factorization and discrete logarithms.

These two problems can be solved in a reasonable amount of time on a quantum computer using an algorithm developed by Peter Shor in 1994 [79]. This causes quantum computers to pose a serious threat to many of the currently used cryptographic solutions.

A considerable amount of research still needs to be done before we have fully functioning large scale quantum computers that can break modern cryptography. In [27], seven stages of development were defined, of which three have been completed, and in [59] it was predicted that a quantum computer able to break RSA-2048 would be invented by 2026 with a probability of 1/7 and by 2031 with a probability of 1/2.

This, however, does not mean that the application of quantum-resistant algorithms can be postponed until quantum computers are invented. Changing standards and official recommendations takes time, performing enough cryptanalysis for an algorithm to be trusted takes even more time and most importantly, data that is saved today can in the future be decrypted by an adversary with a quantum computer. All these factors make post quantum cryptography a highly relevant area of research and all data that has to stay secret for the next decade or two ought to be encrypted with quantum-resistant algorithms. There is even interest on a government level with ETSI (European Telecommunications Standards Institute) releasing a whitepaper about the algorithms and their benefits and challenges (see [18]), NIST (National Institute of Standards and Technology) announcing a competition to find new effective algorithms in [62] and NSA (National Security Agency) announcing plans to transition to quantum-resistant algorithms "in the not too distant future" in [64].

1.1 Goals and problem formulation

The cybersecurity company Advenica is developing a file encryption product that must be secure many years into the future. They are therefore interested in quantum-resistant cryptography. The software will run on an embedded system with high performance demands and will therefore have to perform well with limited resources. The public and private keys will be distributed both through file sharing and using smart cards. Since most smart cards have a limited storage capacity it is important that the size of the keys is reasonable in that context. They are also interested in quantum-resistant cryptography for incorporation into other products in the future.

The primary goal of this thesis is be to study the existing quantum-resistant algorithms, to compare them, and to decide which of them, if any, are ready for industry adoption in general and usage in Advenica's file encryption product in particular. This is done by answering the following questions:

- Which quantum-resistant algorithms are most suited for industry adoption?
  - Which configurations are most suitable and what consequences do these have for security?
  - How well do they perform in constrained environments?
- How well implemented are quantum-resistant algorithms today?
  - What metrics are suitable to use in a comparison?
  - How do the algorithms compare on these metrics?
- What improvements are needed for industry adoption?
  - Is it enough to optimize implementations?
  - Are theoretical improvements needed?
  - Do entirely new algorithms need to be developed?

1.2 Method

To gain a deeper understanding of the threats that quantum computers cause and why they are so important, the work began by a prestudy of quantum computers and the quantum algorithms that are pose a threat to cryptography. This was mostly conducted by reading books on the topic.

The prestudy was followed by a survey of which quantum-resistant algorithms exist today and how they work. This was done to make the step of selecting implementations easier and to gain a better understanding of how the implementations actually work. The relevant algorithms and information about these were mostly found in research papers.

Once this theoretic part was done, meetings were held with Advenica to discuss what metrics would be suitable and how these were to be measured. The chosen metrics can be split up into three different categories; performance, implementation and assurance. What metrics that belong to each category, a short motivation for why they were considered important and how they were measured can be found in Sections 4.3.1-4.4.

In these meetings, it was also discussed how the implementations, that were to be compared, should be selected and what parameters would be suitable to use. It was concluded that it would be more fair if all implementations were written in the same programming language and C was chosen as a suitable candidate. It was also decided that the aim should be to select two parameter sets for each algorithm; one suitable for short term security (5-10 years) and one for long term security (10+ years). The implementations were found by searching on GitHub and by mentions in research papers. The chosen implementations and parameter sets are discussed further in Section 4.1.

1.3 Scope

While it is important that algorithms for encryption and key exchange are secure for some time before the invention of quantum computers, the same urgency does not apply to algorithms for digital signatures. Although a gradual transition and proper cryptanalysis is preferable, it would theoretically be possible to consider all non-quantum-resistant signatures invalid from one day to another, thereby removing the danger that quantum computers pose. For this reason, and due to time constraints, only encryption and key exchange algorithms are considered in this thesis.

Implementing cryptographic algorithms efficiently and securely is a hard and time consuming process. To be able to focus as much as possible on the comparisons, only existing implementations are considered.

1.4 Related work

The Open Quantum Safe project is a project with the goal of supporting development of quantum-resistant cryptography. To aid in this another goal is to provide prototypes of quantum-resistant cryptographic algorithms and integration of these into existing protocols and applications, such as OpenSSL. A GitHub repository containing a collection of prototypes, as well as test- and benchmarking suites to test the correctness and performance of these, called liboqs is maintained as well [65]. In addition to this, a whitepaper on the topic has been written (see [84]), focusing on the LWE-based algorithms in their repository and their implementation of these.

Microsoft research has improved upon two quantum-resistant cryptography schemes and released the code in the LatticeCrypto library [56] and the SIDH library [57]. The code from LatticeCrypto and the first version of SIDH is also included in liboqs.

In [35], Gautam et al. presented security proofs for lattice-based schemes for both encryption and signatures. They then compared the performance of these methods for different key sizes and input data in both constrained and non-constrained environments.

In [45], Johansson and Strahl tested and evaluated a ring learning with errors scheme on a raspberry pi.

1.5 Organization of this thesis

The rest of the thesis is organized as follows. Chapter 2 contains a short theoretical background on cryptography, the algorithms that are at risk, and the results from the prestudy on quantum computers and algorithms. Chapter 3 continues with theory, but contains the results from the survey of the quantum-resistant algorithms that might replace the currently used ones. Chapter 4 describes the selected implementations, the metrics used to compare the these, and how they were tested. The results are presented in Chapter 5. These are discussed in Chapter 6, whereafter conclusions and suggestions for future work are presented in Chapter 7.

Chapter 2

Theoretic background

2.1 Encryption

2.1.1 Symmetric encryption

In symmetric encryption, the keys used to encrypt and decrypt are the same. The main advantage of symmetric encryption is that it is fast, and due to this it is often used when large amounts of data need to be transferred. A problem with symmetric encryption is that both the sender and the receiver have to secretly agree on the same key.

2.1.2 Asymmetric encryption

In asymmetric encryption, the keys to encrypt and decrypt are different. This removes the problem of key distribution since the encryption key, also known as the public key, can be published publicly without compromising the security. A disadvantage is that the encryption and decryption are slow compared to symmetric encryption. For this reason, asymmetric encryption is often used to establish a shared key for a symmetric algorithm that is then used to encrypt the data.

There are principally two ways to use asymmetric encryption to establish a shared key. Either you use an encryption algorithm like RSA (described below) or a key exchange algorithm like Diffie-Hellman (also described below). When an encryption algorithm is used, one party encrypts a randomly selected key with the public key of the other party and sends the ciphertext to them. The other party can then decrypt the ciphertext and the communication can be initiated. If instead a key exchange algorithm is used, the parties agree on a shared secret that both selects a part of. The key can then either be this shared secret or be extracted from it by using, for example, a hash algorithm.

2.2 Classic algorithms

2.2.1 RSA

RSA was first presented in [74] and uses modular exponentiation to encrypt messages. It encrypts data by calculating

$$y = x^e \mod n,$$

where x is the message to be encrypted, n is the product of two large prime numbers (p and q) and e is a number between 1 and $(p-1)(q-1)$. The message can then be recovered by calculating

$$x = y^d \mod n,$$

where d is chosen such that $ed = 1 \mod (p-1)(q-1)$.

The, by far, most time consuming part of RSA is the generation of large primes for the key. To reduce this cost it is possible to instead use several smaller primes, at the cost of reduced security. This approach was taken by Bernstein et al. who, in [10], presented a quantum-resistant version of RSA that uses a one terabyte key consisting of 4096-bit prime numbers.

The security of RSA depends on the fact that, given n and e, it is difficult to compute the factors p and q, and thereby the private key d.

2.2.2 Diffie-Hellman (D-H)

The Diffie-Hellman key exchange was first presented in [28] and uses modular exponentiation to let two parties secretly agree on a common key. It begins with the two parties (Alice and Bob) agreeing on a public modulus p and a base g. Alice then chooses a secret number a and calculates

$$A = g^a \mod p,$$

which she sends to Bob. Bob similarly sends

$$B = g^b \mod p$$

to Alice. They can then both compute the key as

$$B^a = g^{ab} = A^b \mod p.$$

The security of the D-H key exchange is based on the assumption that it is difficult to calculate the discrete logarithm and thereby extracting x from $g^x \mod p$.

2.2.3 Elliptic curve Diffie-Hellman (ECDH)

The way Elliptic curve Diffie-Hellman key exchange works is similar to the classic version, but when using elliptic curves the key-sizes can be decreased. The scheme begins with both parties agreeing on the public parameters which are

- the field $\mathbb{F}_p$ to work in,

- an elliptic curve (e.g. $y^2 = x^3 + ax + b$),
- a cyclic set of points on the curve (cyclic subgroup) represented by one element in the set (G).

Alice then chooses a secret number d_A and calculates $d_A G$ which she sends to Bob, who similarly sends $d_B G$ to Alice. They can then both extract the key from coordinates of $d_A d_B G$.

Similarly to Diffie-Hellman, the discrete logarithm on elliptic curves has to be calculated in order to break the scheme.

2.3 Quantum computing

Below is a brief introduction to quantum computing, a more in depth description can be found in [61].

The idea of taking advantage of properties of quantum particles for computation was first proposed in the 1980's, but the interest was mostly theoretical. It stayed that way until 1994, when Shor published [79], where he devised a quantum algorithm that could factor integers in polynomial time instead of exponential time. Two years later, Grover devised another important quantum algorithm in [37] that leads to a quadratic speed-up when searching through an unordered set. These two algorithms are the primary reason that quantum computers pose a threat to classical cryptography.

2.3.1 Qubits

Quantum computers use qubits instead of regular bits to do their calculations.

A qubit can be represented by any two-state quantum mechanical system, like the polarization of a photon or the spin of an electron. It can, like a classical bit, be in two different base states. These states are often denoted $|0\rangle$ and $|1\rangle$ and represented mathematically by two orthogonal unit vectors. A key difference is, however, that the qubits can be in a linear combination of both states called a superposition. The superposition state is often denoted as

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$$

where α and β are called the amplitudes of the state and are complex numbers where

$$|\alpha|^2 + |\beta|^2 = 1.$$

While this might make it seem like it is possible to store an infinite amount of data in a single qubit, this is not the case. When measured, the superposition collapses into one of the two base states with probability $|\alpha|^2$ for $|0\rangle$ and $|\beta|^2$ for $|1\rangle$, leaving the qubit in that state. Although this makes the qubit more similar to classical bits, the power of the superposition state can be used to perform parallel calculations in a way that is not possible in a classical computer. This can be achieved by using clever mathematics and a phenomenon called quantum entanglement.

Quantum entanglement is when two quantum particles are linked together so that they affect each other, no matter how far apart they are. For example, a two qubit system can be in a superposition denoted $\frac{|00\rangle+|11\rangle}{\sqrt{2}}$. In this state the probability of either qubit being in either state is $1/2$, but when one of them is measured the probability that the other one is in the same state increases to 1. Einstein, who was one of the authors of the paper [30] that first showed this effect, considered the phenomenon impossible and argued that the theory must be incomplete. Quantum entanglement has, however, been demonstrated in several experiments on different kinds of quantum particles. One example is [40], where it was shown for electrons separated by 1.3 kilometers at the time of measurement.

2.3.2 Quantum gates

Just like digital circuits are built using logic gates, quantum circuits are built using quantum gates. Three important gates are the NOT gate, the controlled NOT gate and the Hadamard gate.

The NOT gate is applied to single qubits and works just like a regular NOT gate in that it turns the state $|0\rangle$ into $|1\rangle$ and vice versa. When applied to a quantum bit in superposition it works by interchanging the two probabilities.

The controlled NOT gate works in a similar way, but acts on two qubits and only changes the value of the second qubit when the first qubit is in the state $|1\rangle$.

While the previous two gates could be implemented on a classical computer as well, the Hadamard gate is unique for quantum computers. It acts on a single qubit and places it in a superposition between the two states. Qubits in the state $|0\rangle$ are mapped to $\frac{|0\rangle+|1\rangle}{\sqrt{2}}$, qubits in the state $|1\rangle$ are mapped to $\frac{|0\rangle-|1\rangle}{\sqrt{2}}$ and qubits in a superposition are mapped to a linear combination of these two new bases. When applied to a system of n entangled qubits in succession it can place them in the superposition

$$\frac{1}{\sqrt{N}}\sum_{x=0}^{N-1}|x\rangle$$

where $N = 2^n$. This means that the amount of information the qubits are able to work with increases exponentially with the number of qubits.

2.3.3 Grover's algorithm

Grover's algorithm is a quantum algorithm for searching that was first proposed by Grover in [37]. It can find an element in an unsorted set of data with a high probability in only $O(\sqrt{n})$ steps instead of the $O(n)$ steps required by a classical computer. It works by first placing the qubits in a superposition of all possible states using Hadamard gates, and then enhancing the probability of the sought element. This is done by applying two operators multiple times.

The first operator is called a *phase inversion* and it works by changing the sign of the amplitude of the sought element. For instance, the state $\frac{|00\rangle+|01\rangle+|10\rangle+|11\rangle}{2}$ gets mapped to $\frac{|00\rangle+|01\rangle+|10\rangle-|11\rangle}{2}$ if we are looking for the last element in the set.

The second operator is called *inversion about the mean* and works similarly, but instead of flipping the sought element around 0 it flips all elements around the mean of the amplitudes. This will lower the probability for all states except for the sought one, which is amplified.

By applying these two operators multiple times, the probability of finding the correct element can be made arbitrarily high before a measurement is made.

Grover's algorithm increases the speed at which it is possible to do a brute-force search for cryptographic keys. This affects all cryptographic algorithms, but a sufficient counter-measure is to double the key-size.

2.3.4 Shor's algorithm

Shor's algorithm is a quantum algorithm for factoring integers and calculating discrete logarithms in polynomial time, proposed by Shor in [79]. It has since been modified and generalized by, among others, Ekerå and Håstad in [31]. The algorithm works by using classical computations to translate the problem to the problem of finding the period of a function. The period is then found using quantum computation before it is classically translated back to the original problem.

The period-finding part of Shor's algorithm begins, just as Grover's algorithm, by placing the qubits in a superposition of all possible states. The function is then applied to all states, whereafter the result is transformed using a quantum version of the Fourier transformation. Finally the qubits are measured with a high probability of getting the correct answer. In the generalized version of the algorithm the quantum part can be repeated a couple of times in succession to reduce the number of qubits needed at the cost of more runs.

Shor's algorithm poses a devastating threat to cryptographic algorithms such as RSA and Diffie-Hellman key exchange that cannot be solved by a reasonable increase in key size. This means that, to protect against quantum computers, entirely new algorithms are needed.

2.4 Problems

Quantum computers seem to excel at number-theoretic problems, such as integer factorization and discrete logarithms, as a result of algorithms like the quantum Fourier transform. Other problems, like the ones stated below, are harder for both quantum and classical computers and there are currently no algorithms that can solve large instances of these problems in a reasonable amount of time. This quality means that they are good bases to build cryptographic schemes upon.

2.4.1 Terminology

Complexity classes Computational complexity theory divides computational problems into different classes based on how hard they are to solve. The most basic class is called P and contains all decision problems (problems with yes or no answer) that can be solved in polynomial time. The class P is contained within the class NP, which contains all problems for which the solution is verifiable in polynomial time. If P=NP is one of the most famous unsolved problems in computer science and it is widely believed that this is not the case. Another important class is called NP-hard, and a problem belongs to this category if every problem in NP can be reduced to it in polynomial time and they are therefore at least as hard as the problems in NP.

Lattice A lattice is a discrete subgroup of $\mathbb{R}^n$ that can be represented as a linear combination of n basis vectors with integer coefficients. A visual representation of a two-dimensional lattice can be found in Figure 2.1.

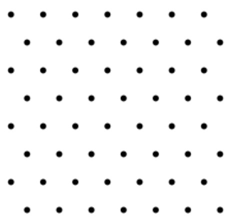

Figure 2.1: A visual representation of a two-dimensional lattice

2.4.2 Lattice problems

Lattice problems are well suited for cryptographic uses since many of them are proved to be average case hard.

Shortest Vector Problem (SVP) Given a lattice L and a norm N, the shortest vector problem is the problem of finding the shortest nonzero vector in L as measured by N. The problem was conjectured to be NP-hard in 1981 and was later proved to be so under certain conditions using randomized reduction in [1] and [54]. The equivalent decision version of this problem is to, when given a number d, decide if the shortest vector of the lattice is longer or shorter than d. A related problem is the GapSVP where an additional parameter γ is given and the goal is to decide if the shortest vector of the lattice is shorter than d or longer than γd.

Closest vector problem (CVP) A generalization of SVP is called the closest vector problem. Here the goal is to, when given a vector space V, a metric M, a lattice L and a vector v in V, find the vector in L that is closest to v as measured by M.

2.4.3 Learning With Errors (LWE)

The goal of the LWE problem is to solve a system of equations containing errors sampled from some statistical distribution. The errors make this problem harder since some classic methods for solving systems of equations, like Gaussian elimination, will make the errors propagate and thereby make the system even harder to solve. The decision version of this problem is to, when given pairs of integers (a_i, b_i), decide if there exist variables s and e_i such that $a_i = b_i s + e_i$. It is usually written as the matrix equation $\mathbf{Ax} = \mathbf{b}$ where $\mathbf{A}$ is a matrix and $\mathbf{b}$ is a vector. This problem has been shown by Regev in [73], to be at least as hard as the hardest lattice-problems.

To reduce the size of the parameters, it is common to add structure to the problem using a polynomial ring. This turns all variables into polynomials instead of vectors and matrices. Most importantly $\mathbf{A}$ becomes a polynomial of order n instead of an $n \times n$ matrix, thereby reducing the number of calculations that has to be made and the data that has to be stored. The Ring-LWE problem has been proven in [49] be at least as hard as the SVP on ideal lattices, which are lattices with a certain structure. Although this structure might make it easier to solve the problem, there is no known method for exploiting it today.

2.4.4 Supersingular elliptic curve isogenies

An isogeny is a mapping between elliptic curves that preserves the algebraic structure of the curves. More formally it is a non-trivial, rational mapping $\phi : E_1 \to E_2$ such that $\phi(P + Q) = \phi(P) + \phi(Q)$ where P and Q are any points in E_1.

General version The problem of finding isogenies between supersingular elliptic curves has proven to be hard even for a quantum computer. The best known quantum algorithm for solving the general version of this problem is described in [26] and does this in exponential time.

Claw problem The version of the problem used in the SIDH key exchange described in Section 3.3 uses a sightly easier version of the general problem called the claw problem. The claw problem is to, when given two functions $f : A \to C$ and $g : B \to C$, find a pair (a, b) such that $f(a) = g(b)$. While this problem is easier, the best known quantum algorithm still requires exponential time.

2.4.5 Decoding a linear code

A linear code is a way to encode data to be able to correct errors that show up during e.g. transmission of this data. In a linear code any linear combination of codewords is also a codeword. The code-based cryptosystems presented in Section 3.4 are all based on the problem of decoding a linear code. One of the best algorithms for solving this problem is presented in [12], it is not, however, good enough to pose a significant threat to the schemes presented in Section 3.4.

2.4.6 Multivariate quadratic equations

The problem of solving a system of m quadratic equations in n variables is proven to be NP-complete in [34]. There are, however, some special cases where there exist polynomial time algorithms. Two of these are when the number of variables are a lot more or a lot fewer than number of equations. Examples of these two cases can be found in [89] and [23] respectively.

Chapter 3

Algorithms

Describing each quantum-resistant algorithm in detail would be enough to fill a book (like [7]). This chapter aims to provide an overview of most of the currently existing quantum-resistant algorithms for encryption and key exchange with slightly more detail regarding the ones considered in the tests later.

3.1 Notation

Let $\mathbb{Z}_q$ be the rational integers modulo q and let $\mathscr{R}_q = \mathbb{Z}_q[X]/(X^n + 1)$ be the ring of polynomials modulo $(X^n + 1)$ with coefficients in $\mathbb{Z}_q$.

If χ is a probability distribution over $\mathbb{Z}_q$, then $x \overset{\$}{\leftarrow} \chi$ denotes sampling a value from $\mathbb{Z}_q$ according to χ and if χ is a set it denotes sampling uniformly from this set.

As is common in in the field of cryptography, the two parties involved in the encryption or key exchange will be called Alice and Bob.

3.2 Lattice-based algorithms

Lattice-based cryptography is a promising area for post quantum cryptography. Lattice problems often have good security proofs and provide an overall good performance. This section consists of three parts, all containing a scheme based on the LWE problem described in Section 2.4.3 or the lattice problems in Section 2.4.2. The first part describes different versions of an algorithm for key exchange and the second and the third describe algorithms for encrypting data.

3.2.1 LWE key exchange

The key exchange algorithms presented in this section are all based on the LWE problem and work in about the same way. The main idea is to agree on an approximately equal secret and then use special functions to extract bits from that secret.

Four special functions that are important for this type of scheme are; the modular rounding function, the cross rounding function, the randomized doubling function and the reconciliation function.

The modular rounding function extracts the B most significant bits of a number and is defined as

$$\lfloor\cdot\rceil_{2^B} : \mathbb{Z}_q \to \mathbb{Z}_{2^B}, x \mapsto \lfloor x\rceil_{2^B} = \lfloor 2^{-\bar{B}}x\rceil \mod 2^B$$

where $\lfloor\cdot\rceil$ rounds a real number to the closest integer and $\bar{B} = \log_2 q - B$.

The cross rounding function extracts the (B+1)th most significant bit and is defined as

$$\langle\cdot\rangle_{2^B} : \mathbb{Z}_q \to \mathbb{Z}_2, x \mapsto \langle x\rangle_{2^B} = \lfloor 2^{-\bar{B}+1}x\rfloor \mod 2.$$

The randomized doubling function doubles the input value and introduces a small random noise. It is defined as

$$dbl(\cdot) : \mathbb{Z}_q \to \mathbb{Z}_{2q}, x \mapsto dbl(x) = 2x + e$$

where e is sampled uniformly from $\{-1, 0, 0, 1\}$.

The reconciliation function, denoted $rec(w, b)$, takes $w \in \mathbb{Z}_q$ and $b \in \{0, 1\}$ as input and outputs $\lfloor v\rceil_{2^B}$ where v is the closest integer to w where $\langle v\rangle_{2^B} = b$. It can be implemented in different ways, but the original suggestion by Peikert in [68] works by partitioning the integers modulo q into two intervals depending on b and then returning 1 or 0 depending on which interval w is in.

When applied to polynomials, the aforementioned functions are applied to each coefficient. For matrices and vectors they are applied element-wise.

BCNS15 In [14] Bos et al. presented an implementation of the ring-LWE cryptosystem from [49]. It has four public parameters

- n, the size of the polynomials used.
- q, the modulus used for the coefficients.
- $\mathbf{a}$, a polynomial corresponding to the $\mathbf{A}$ matrix in the regular LWE problem.
- χ, the distribution the random variables are sampled from, in this case it is a discrete Gaussian distribution.

The scheme begins by having Alice sample her secret polynomial $\mathbf{s}$ and an error polynomial $\mathbf{e}$ from χ. She then computes

$$\mathbf{b} = \mathbf{as} + \mathbf{e},$$

which she sends to Bob. He then samples a secret polynomial $\mathbf{s}'$ and two error polynomials $\mathbf{e}'$ and $\mathbf{e}''$, whereafter he computes

$$\mathbf{b}' = \mathbf{as}' + \mathbf{e}'$$

and

$$\mathbf{v} = \mathbf{bs}' + \mathbf{e}''.$$

Then he uses the randomized doubling function to get

$$\overline{\mathbf{v}} = dbl(\mathbf{v})$$

since the rounding and the reconciliation functions require the input modulus to be a multiple of two. After this he can calculate the key as

$$k_b = \lfloor \overline{\mathbf{v}} \rceil_2.$$

Finally, he sends $\mathbf{b}'$ to Alice together with the cross rounding of $\overline{\mathbf{v}}$, whereafter she calculates the key as

$$k_A = rec(\mathbf{b}'\mathbf{s}, c).$$

The keys will be equal with high probability since

$$\mathbf{v} = \mathbf{b}\mathbf{s}' + \mathbf{e}'' = \mathbf{a}\mathbf{s}\mathbf{s}' + \underline{\mathbf{e}\mathbf{s}' + \mathbf{e}''} \approx \mathbf{a}\mathbf{s}'\mathbf{s} + \underline{\mathbf{e}'\mathbf{s}} = \mathbf{b}'\mathbf{s}.$$

The approximation equality is close enough for the reconciliation function to correctly recreate the bit.

VS In [81], and later in [82] together with Chopra, Singh presented a version of BCNS15 where they, somewhat counterintuitively, managed to decrease the key size and increase the security by decreasing the modulus q. This increases the security since it will increase the noise-to-modulus ratio, making it harder to break, but also increases the risk of errors during key exchange. They also proposed that the noise should be sampled from a uniform distribution instead of a Gaussian to increase the speed. Together with the paper they presented a repository (see [80]) with implementations of their improved algorithm using both uniform and Gaussian distributions for different security levels. This repository has later been expanded to include noise sampled from a binomial distribution as well.

New Hope This is an improved version of BCNS15 presented by Alkim et al. in [4]. There are five major improvements:

- A binomial distribution ψ is used instead of a discrete Gaussian. This makes the implementation faster since the binomial distribution is easier to sample from.
- The public polynomial $\mathbf{a}$ is calculated from a seed generated by one of the parties. This increases the amount of data sent slightly and each party has to do some extra calculations, but it defends against backdoors and all-for-the-price-of-one attacks since the parameter changes between sessions.
- The modulus q is lowered significantly to increase both speed and security.
- A more effective implementation of the number theoretic transform is used. This helps speed up the multiplication of large numbers.
- A new reconciliation function and a function called HelpRec are introduced to replace the modular rounding and the cross rounding functions.

The new reconciliation function uses the coefficients of the polynomial as coordinates in a four dimensional space. The space is then divided into cells and it gives the same output for coordinates in the same cell. The HelpRec function works by calculating in which part of the cell the coordinate is so that the distance from the center can be subtracted in the reconciliation function, thereby increasing its accuracy. The scheme is described in Figure 3.1.

Public parameters: n, q, ψ		
Alice		Bob
$seed \leftarrow \{0,1\}^{256}$		
$\mathbf{a} \leftarrow \text{parse}(seed)$		
$\mathbf{s}, \mathbf{e} \overset{\$}{\leftarrow} \psi$		
$\mathbf{b} \leftarrow \mathbf{as} + \mathbf{e}$	$\xrightarrow{\mathbf{b}, seed}$	$\mathbf{s}', \mathbf{e}', \mathbf{e}'' \overset{\$}{\leftarrow} \psi$
		$\mathbf{a} \leftarrow \text{parse}(seed)$
		$\mathbf{b}' \leftarrow \mathbf{as}' + \mathbf{e}'$
		$\mathbf{v} \leftarrow \mathbf{bs}' + \mathbf{e}''$
		$\mathbf{r} \leftarrow HelpRec(\mathbf{v})$
$k_A \leftarrow rec(\mathbf{b}'\mathbf{s}, \mathbf{r})$	$\xleftarrow{\mathbf{b}', \mathbf{r}}$	$k_B \leftarrow rec(\mathbf{v}, \mathbf{r})$

Figure 3.1: New hope key exchange

MSRLN16 In [48] Longa and Naehrig presented improvements to the number theoretic transform used in New Hope. They also implemented their own version of New Hope with this improvement. These changes were later implemented in New Hope as well, making the two implementations very similar.

Frodo Although there currently is no way of taking advantage of the structure of ring-LWE in any significant way when trying to break the previously presented schemes, further cryptanalysis might find one. This motivated Bos et al. in [13] to present a scheme where they removed the ring structure.

Removing the ring structure leads to larger keys and slower performance, mostly since $\mathbf{a}$ now has to be an $n \times n$ matrix instead of a polynomial of size n.

3.2.2 LWE encryption

LP In [47], Lindner and Peikert presented a public encryption algorithm based on the LWE problem that is an instance of an abstract cryptosystem by Micciancio in [55]. The system has four public parameters q, χ_k, χ_e, and $\mathbf{A}$. The parameter q is the modulus used for all calculations, χ_k and χ_e are discrete Gaussian distributions where the keys and the noise are sampled from, and $\mathbf{A}$ is a uniformly sampled random matrix of size $n \times n$. The message, m, is of length l. The key generation begins by having Alice sample two random matrices, $\mathbf{E}$ and $\mathbf{S}$, of size $n \times l$ from χ_k. She can then compute the public key as

$$\mathbf{P} = \mathbf{E} - \mathbf{A} \cdot \mathbf{S}$$

and use $\mathbf{S}$ as the private key.

To encrypt a message Bob samples three error vectors, $\mathbf{e_1}$, $\mathbf{e_2}$ and $\mathbf{e_3}$ of length n, n and l respectively. He can then compute the ciphertext as

$$\mathbf{c_1} = \mathbf{e_1} \cdot \mathbf{A} + \mathbf{e_2}$$

and

$$\mathbf{c_2} = \mathbf{e_1} \cdot \mathbf{P} + \mathbf{e_3} + \mathbf{m}.$$

To decrypt this message, Alice computes

$$\overline{\mathbf{m}} = \mathbf{c_1} \cdot \mathbf{S} + \mathbf{c_2} = \mathbf{e_1} \cdot \mathbf{A} \cdot \mathbf{S} + \mathbf{e_2} \cdot \mathbf{S} + \mathbf{e_1} \cdot \mathbf{P} + \mathbf{e_3} + \mathbf{m},$$

which she then runs through a decoding algorithm to remove the errors and extract the message. The algorithm works if

$$|\mathbf{e_1} \cdot \mathbf{A} \cdot \mathbf{S} + \mathbf{e_2} \cdot \mathbf{S} + \mathbf{e_1}\mathbf{P} + \mathbf{e_3}|$$

$$= |\mathbf{e_1} \cdot \mathbf{A} \cdot \mathbf{S} + \mathbf{e_2} \cdot \mathbf{S} + \mathbf{e_1} \cdot (\mathbf{E} - \mathbf{A} \cdot \mathbf{S}) + \mathbf{e_3}|$$

$$= |\mathbf{e_2} \cdot \mathbf{S} + \mathbf{e_1}\mathbf{E} + \mathbf{e_3}|$$

is smaller than the error correcting threshold for the decoding algorithm.

U-LP In [16], Cabarcas et al. proposed that the secret keys and the noise are sampled uniformly instead of according to a discrete Gaussian distribution, since it is faster and easier. This, however, leads to larger matrices having to be used for equal security. This scheme is described in Figure 3.2.

Public parameters: q, $\mathbf{A} \xleftarrow{\$} \mathbb{Z}_q^{n\times n}$		
Alice		Bob
$\mathbf{E} \xleftarrow{\$} \mathbb{Z}_{s_k}^{n\times l}$		
$\mathbf{S} \xleftarrow{\$} \mathbb{Z}_{s_k}^{n\times l}$		
$\mathbf{P} \leftarrow \mathbf{E} - \mathbf{A} \cdot \mathbf{S}$		
Alice's public key: $\mathbf{P}$		
Alice's private key: $\mathbf{S}$		
		$\mathbf{e_1}, \mathbf{e_2} \xleftarrow{\$} \mathbb{Z}_{s_e}^{1\times n}$
		$\mathbf{e_3} \xleftarrow{\$} \mathbb{Z}_{s_e}^{1\times l}$
		$\mathbf{c_1} \leftarrow \mathbf{e_1} \cdot \mathbf{A} + \mathbf{e_2}$
$\overline{\mathbf{m}} \leftarrow \mathbf{c_1} \cdot \mathbf{S} + \mathbf{c_2}$	$\xleftarrow{\mathbf{c_1},\mathbf{c_2}}$	$\mathbf{c_2} \leftarrow \mathbf{e_1} \cdot \mathbf{P} + \mathbf{e_3} + \mathbf{m}$
$\mathbf{m} \leftarrow \mathit{decode}(\overline{\mathbf{m}})$		

Figure 3.2: U-LP encryption scheme

3.2.3 NTRU

NTRU was first presented by Hoffstein, Pipher and Silverman in [43]. The system is characterized by the public parameters N, that specifies the size of the polynomials used, a large modulus q and a smaller modulus p. The process begins by having Alice create a public and a private key by generating two polynomials $\mathbf{f}$ and $\mathbf{g}$, where $\mathbf{f}$ is invertible modulo both p and q. She can then calculate the public key as

$$\mathbf{h} = p\mathbf{f}_q\mathbf{g} \mod q$$

where

$$\mathbf{f}_q = \mathbf{f}^{-1} \mod q.$$

The private key consists of the polynomials $\mathbf{f}$ and

$$\mathbf{f}_p = \mathbf{f}^{-1} \mod p.$$

When Bob wants to send a message m to Alice he encodes it as a polynomial with coefficients modulo p and generates a random polynomial $\mathbf{r}$ with small coefficients to cloak the message. He can then calculate the encrypted message as

$$\mathbf{e} = \mathbf{rh} + \mathbf{m} \mod q.$$

To decrypt this message Alice begins by computing

$$\mathbf{a} = \mathbf{fe} = \mathbf{frh} + \mathbf{fm} = p\mathbf{rg} + \mathbf{fm} \mod q.$$

After this she removes the cloaking by computing

$$\mathbf{b} = \mathbf{a} \mod p,$$

after which she can extract the message as

$$\mathbf{m} = \mathbf{f}_p\mathbf{b} \mod p.$$

The scheme is also described in Figure 3.3.

Public parameters: N, p, q		
Alice		Bob
$\mathbf{f} \xleftarrow{\$} \mathscr{R}_p$		
$\mathbf{g} \xleftarrow{\$} \mathscr{R}_p$		
$\mathbf{f_p} \leftarrow \mathbf{f}^{-1} \mod p$		
$\mathbf{f_q} \leftarrow \mathbf{f}^{-1} \mod q$		
$\mathbf{h} \leftarrow \mathbf{pf_qg}$		
Alice's public key: $\mathbf{h}$		
Alice's private key: $\mathbf{f}$, $\mathbf{f_p}$		
		$\mathbf{m} \in \mathscr{R}_p$
		$\mathbf{r} \xleftarrow{\$} \mathscr{R}_p$
$\mathbf{a} \leftarrow \mathbf{fe} \mod q$	$\xleftarrow{\mathbf{e}}$	$\mathbf{e} \leftarrow \mathbf{rh} + \mathbf{m}$
$\mathbf{b} \leftarrow \mathbf{a} \mod p$		
$\mathbf{m} \leftarrow \mathbf{f_p b} \mod p$		

Figure 3.3: NTRU public key encryption scheme.

3.3 Supersingular isogenic Diffie-Hellman (SIDH)

The idea to use supersingular elliptic curves for key exchange is a relatively new one. It has therefore not received the same amount of cryptanalysis as most of the other algorithms presented in this thesis, but like Diffie-Hellman over regular elliptic curves it uses small keys which makes it an interesting alternative to consider. The main idea behind SIDH is for both parties to each construct a private supersingular elliptic curve and then exchange information so they can construct a second pair of supersingular elliptic curves. The curves in this second pair are so similar to each other that an equal key can be extracted from them. The algorithm in its current form was first presented by De Feo, Jao and Plût in [26] and later improved upon by Costello, Longa and Naehrig in [22]. A technique for compressing the public keys to further reduce their size has been proposed by Costello et al. in [21]. The scheme is described in more detail below and in Figure 3.4.

The public parameters of the scheme are

- a prime $p = w_A^{e_A} w_B^{e_B} f \pm 1$, where w_A and w_B are primes, and e_A and e_B are integers,
- a supersingular elliptic curve E over $\mathbb{F}_{p^2}$,
- four points on E called P_A, Q_A, P_B and Q_B, where the order of P_A and Q_A is $w_A^{e_A}$ and the order of P_B and Q_B is $w_B^{e_B}$.

Alice begins by generating two random integers, m_A and n_A, and calculates a point

$$R_A = m_A P_A + n_A Q_A,$$

which she uses to create a private isogeny mapping

$$\phi_A : E \to E_A.$$

She then sends the new elliptic curve E_A to Bob together with the two points $\phi_A(P_B)$ and $\phi_A(Q_B)$. Bob performs similar calculations and sends E_B, $\phi_B(P_A)$ and $\phi_B(Q_A)$ to Alice. Alice can now generate the point

$$S_{BA} = m_A \phi_B(P_A) + n_A \phi_B(Q_A),$$

which she uses to create another isogeny mapping

$$\psi_{BA} : E \to E_{BA}.$$

Similarly, Bob creates

$$\psi_{AB} : E \to E_{AB}.$$

They can then calculate the a property called the j-invariant of E_{AB} and E_{BA}, respectively. The j-invariant of isomorphic curves are equal, meaning the two keys will be equal.

Public parameters: p, $E/\mathbb{F}_{p^2}$, P_A, Q_A, P_B, Q_B		
Alice		Bob
$m_A, n_A \xleftarrow{\$} \mathbb{F}_{p^2}$		
$R_A \leftarrow m_A P_A + n_A Q_A$		
Create $\phi_A : E \to E_A$	$\xrightarrow{E_A, \phi_A(P_B), \phi_A(Q_B)}$	$m_B, n_B \xleftarrow{\$} \mathbb{F}_{p^2}$
		$R_B \leftarrow m_B P_B + n_B Q_B$
		Create $\phi_B : E \to E_B$
		$S_{AB} \leftarrow m_A \phi_A(P_B) + n_A \phi_A(Q_B)$
		Create $\psi_{AB} : E \to E_{AB}$
		$K \leftarrow$ j-invariant(E_{AB})
	$\xleftarrow{E_B, \phi_B(P_A), \phi_B(Q_A)}$	
$S_{BA} \leftarrow m_B \phi_B(P_A) + n_B \phi_B(Q_A)$		
Create $\psi_{BA} : E \to E_{BA}$		
$K \leftarrow$ j-invariant(E_{BA})		

Figure 3.4: SIDH-based key exchange.

3.4 Code-based cryptography

The code-based methods are the oldest of the quantum-resistant algorithms. They have an overall good performance, but the keys are usually very large. Attempts to decrease the key-size by using different kinds of codes have been made, but these often result in insecure schemes. The original scheme by McEliece uses a binary linear code by Goppa called Goppa code, this was first presented in [36] (in Russian) and a summary in English can be found in [6].

3.4.1 McEliece

McEliece was the first public key cryptosystem to use randomness while encrypting. It was presented by McEliece in [52] and has since been improved upon for increased performance. The basic idea behind the scheme has remained the same though.

The process begins by having Alice construct the public and private keys. To do this she first chooses a binary linear (n, k)-code C, this code must have an efficient decoding algorithm and be able to correct t errors. She then generates a $k \times n$ generator matrix $\mathbf{G}$ for C, a $k \times k$ binary non-singular scrambling matrix $\mathbf{S}$ and a random $n \times n$ permutation matrix $\mathbf{P}$. She can then compute the $k \times n$ matrix

$$\hat{\mathbf{G}} = \mathbf{S} \cdot \mathbf{G} \cdot \mathbf{P},$$

which together with t makes up the public key. The private key consists of the matrices $\mathbf{S}$, $\mathbf{G}$ and $\mathbf{P}$ and the decoding algorithm $\mathcal{D}_G$.

When Bob wants to send a message to Alice he has to encode it as binary vectors $\mathbf{m}$ of length k. He can then encrypt the message by calculating

$$\mathbf{c} = \mathbf{m} \cdot \hat{\mathbf{G}} \oplus \mathbf{e}$$

for every $\mathbf{m}$, where $\mathbf{e}$ is a binary vector containing t 1's placed randomly.

When Alice wants to decrypt the ciphertext she begins by multiplying it with the inverse of the permutation matrix to get

$$\mathbf{c} \cdot \mathbf{P}^{-1} = \mathbf{m} \cdot \mathbf{S} \cdot \mathbf{G} \oplus \mathbf{e} \cdot \mathbf{P}^{-1}.$$

She then uses the decoding algorithm to remove the errors whereafter she chooses k columns from $\hat{\mathbf{G}}$ that form an invertible matrix $\mathbf{G}_J$ and selects the corresponding columns from the decoded vector to form $\hat{\mathbf{c}}_J$. Finally, she multiplies $\hat{\mathbf{c}}_J$ with the inverses of $\mathbf{G}_J$ and $\mathbf{S}$ to extract the message. There is a small chance that the decryption fails, in which case Alice has to notify Bob so that he can resend the message with a new $\mathbf{e}$ vector.

Quasi cyclic McEliece In [33] Gaborit proposed that the key size could be reduced by using quasi cyclic (QC) low density parity check (LDPC) codes. Quasi cyclic means that each row in the matrix is a cyclic shift of the first leading to the possibility of representing a full matrix using only a single row, and LDPC codes have sparse parity check matrices leading to further reduced key sizes. Later, in [58], Misoczki et al. proposed to use quasi cyclic medium density parity check (MDPC) codes instead. Using MDPC provides a couple of benefits such as reduced private key size since the MDPC codes have a random component meaning that there is no need for the permutation and substitution matrices. The quasi cyclic schemes are, however, harder to keep secure due to the reduced amount of possible matrices and that part of the structure is revealed. In [39] Guo et al. presented a way for Bob to calculate Alice's private key based on decryption failures when using QC MDPC codes, and in [50] Löndahl et al. presented an attack on QC MDPC of even dimension. An attack similar to the one by Guo et al. is presented by Fabsic et. al in [32], this attack does, however, pose a threat to QC LDPC codes instead.

These attacks are not devastating enough to render the quasi cyclic schemes useless, more research and slight modifications will be needed before they can be used securely.

3.4.2 Niederreiter scheme

One of the more significant changes to the McEliece cryptosystem is by Niederreiter who, in [60], presented changes to decrease the key size of the scheme. He did this by using the parity check matrix instead of the generator matrix and by encoding the message into the $\mathbf{e}$ vector. These changes reduce the size of the public key to a $(n-k) \times n$ matrix. The Niederreiter scheme is shown in Figure 3.5.

Public parameters: t, n, k		
Alice		Bob
$G \leftarrow$ linear (n,k)-code		
$\mathcal{D}_G \leftarrow$ Decoding algorithm for G		
$\mathbf{H} \leftarrow$ parity check matrix for G		
$\mathbf{S} \xleftarrow{\$} \{0,1\}^{(n-k)\times(n-k)}$		
$\mathbf{P} \leftarrow n \times n$ permutation matrix		
$\hat{\mathbf{H}} \leftarrow \mathbf{S} \cdot \mathbf{H} \cdot \mathbf{P}$		
Alice's public key: $\hat{\mathbf{H}}$		
Alice's private key: $\mathbf{H}$, $\mathbf{S}$, $\mathbf{P}$, $\mathcal{D}_G$		
		$\mathbf{m} \leftarrow \mathbb{Z}_2$ (only t ones)
	$\xleftarrow{\mathbf{c}}$	$c \leftarrow \hat{\mathbf{H}} \cdot \mathbf{m}^T$
$\hat{\mathbf{c}} \leftarrow \mathbf{S}^{-1} \cdot \mathbf{c}$		
$\hat{\mathbf{m}} \leftarrow \mathcal{D}_G(\hat{\mathbf{c}})$		
$\mathbf{m} \leftarrow (\mathbf{P}^{-1} \cdot \hat{\mathbf{m}})^T$		

Figure 3.5: Niederreiter public key encryption scheme.

McBits In [9] Bernstein, Chou and Schwabe present a version of the Niederreiter scheme where their main goal was to improve the decryption time. While their scheme is based on the Niederreiter scheme, they have introduced some quite large changes. The private key is created from a random sequence $(\alpha_1 ... \alpha_n)$ of distinct elements from $\mathbb{F}_{2^m}$ and an irreducible polynomial g of degree t with coefficients from $\mathbb{F}_{2^m}$ by creating the matrix

$$\begin{pmatrix} 1/g(\alpha_1) & 1/g(\alpha_2) & \cdots & 1/g(\alpha_n) \\ \alpha_1/g(\alpha_1) & \alpha_2/g(\alpha_2) & \cdots & \alpha_n/g(\alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{t-1}/g(\alpha_1) & \alpha_2^{t-1}/g(\alpha_2) & \cdots & \alpha_n^{t-1}/g(\alpha_n) \end{pmatrix}$$

and replacing each entry by its m-bit representation. The public key can then be calculated by Gaussian elimination on the private key.

When Bob wants to send a message to Alice he generates a random binary error vector of length n with exactly t ones which he multiplies with the public key to obtain w, which will be part of the ciphertext. The error vector is then hashed and the hash is split up into two parts. One part is used as a key to encrypt the message using a symmetric encryption algorithm and the other is used as a key for a MAC to verify the encrypted message. The ciphertext consists of the encrypted error vector, the encrypted message and the MAC.

When Alice receives the ciphertext she uses her knowledge of the secret key to decode the error vector. She can then hash it herself to obtain the keys, whereafter she verifies the MAC and lastly decrypts the message.

3.5 Multivariate cryptography

The main idea behind the cryptosystems based on multivariate equations is to create a system of equations having hidden properties that makes it easy to solve. The public key is then created by hiding these properties using affine transformations.

To encrypt a message Bob encodes it as a vector x and inserts it into the public system to get the ciphertext y. To decrypt the message Alice uses her secret information about the system to solve for x. If the system is of higher degree it will have multiple solutions, in that case Bob has to add redundancy to his message to help Alice find the right one.

This idea was first proposed by Matsumoto and Imai in [51]. Their scheme was broken by Patarin who later, in [67], presented a new scheme called Hidden Field Equations (HFE) that is a generalized version of Matsumoto and Imai's scheme and solves the problems he found. This generalized version has since been broken for encryption, but remains secure for signatures.

The main problem with the two previously mentioned schemes is that some quadratic forms associated with the central map has low rank. A newer scheme that intends to solve this problem was presented by Tao et al. in [88]. The scheme is called Simple Matrix Scheme or ABC for short and uses a central map of relatively high rank and solves all problems related to it. The ABC scheme has since been broken as well. It was broken in [38] by Gu who proved that the secret key can be recovered by taking advantage of a certain algebraic structure of the scheme.

Another scheme that aimed to solve the low rank problem was presented by Porras et al. in [71, 72]. Their approach has been more successful than the ABC scheme and there is currently no known attack that breaks the scheme. Unfortunately, no implementation of this algorithm has been found during the research for this thesis.

Chapter 4

Implementations and metrics

This chapter described the implementations that were selected and the metrics on which they were compared.

4.1 Implementations

4.1.1 Code

To make the comparisons easier and more fair all implementations are written in C and all implementations except for the ones from VS crypto with Gaussian distribution run in constant time. During the tests the following repositories have been used.

liboqs [65] As mentioned in Chapter 1, the Open Quantum Safe Project is a project aiming to support development of quantum-resistant cryptography and to provide prototypes of quantum-resistant algorithms. The provided repository, liboqs, is a collection of algorithms made both by the creators of the project and others. Apart from the tested algorithms mentioned below it contains an older version of the SIDH algorithm by Microsoft research as well as a test suite and a benchmarking routine. The algorithms from liboqs tested in this thesis are;

- the ring-LWE algorithms BCNS15, New Hope, and MSRLN16,
- the LWE algorithm Frodo,
- the Code-based algorithm McBits,

NTRUosp [76] The company behind NTRU released a sample implementation to motivate a more widespread adoption of their algorithm. From this repository two parameter sets have been tested. The sets are called NTRU_EES439EP1 and NTRU_EES743EP1, and are referred to as NTRU 439 and NTRU 743 in this thesis.

Microsoft SIDH v2.0 [57] Microsoft research has released a library with an implementation of SIDH proposed by Costello et al. in [22]. This library also contains the compression technique proposed in [21]. The two versions, with and without compression, will be referred to as CLN16 and CLN16 (comp).

VS crypto [80] In addition to proposing changes of the distribution from which the noise is sampled, Singh also provided a repository of sample implementations using a lot of different parameter sets in a repository that will be referred to as VScrypto in this thesis. The implementations from VScrypto that are tested in this thesis are:

- VS-512-uniform and VS-1024-uniform, that use a uniform distribution and the same reconciliation method as BCNS15,
- VS-512-gauss and VS-1024-gauss that, use a Gaussian distribution and the same reconciliation method as BCNS15,
- VS-512-gauss-ct and VS-1024-gauss-ct, that is a constant time implementation using a Gaussian distribution and the same reconciliation method as BCNS15.,
- VS-512-bin and VS-1024-bin, that use a binomial distribution and the same reconciliation method as New Hope.

OpenSSL [66] To test RSA and ECDH, libcrypto from OpenSSL 1.1.0 has been used. Since the focus of this thesis was to compare the quantum-resistant algorithms, the classical algorithms were only measured when they would be helpful as a reference.

4.1.2 Security and parameters

In [5], NIST recommends that all data that has to stay secure for more than a decade should use at least 128-bit security. The aim of the parameter selection has been to provide one recommended parameter set with about 128 bits of security and one high security parameter set with about 256 bits. The high security parameter sets are included for two reasons.

- Since the quantum-resistant algorithms lack the same cryptanalysis as RSA and ECDH, further cryptanalysis might weaken the security. 256-bit security provides a good margin for the security.
- If Grover's algorithm can be applied directly to the algorithm (which is not always the case), this would halve the security. The high security parameter sets will thus provide at least 128 bits of quantum security.

In some cases it has not been practical to provide both parameter sets, either due to them being highly correlated with the implementation, or because no suitable parameter sets has been found. The claimed security levels are all taken from the papers mentioned below and there might therefore be attacks that have been presented recently that would decrease the claimed security levels. All current security claims are summarized in Table 4.1.

Name	**From**	**Claimed classical security**
New Hope	Liboqs	281
MSRLN16	Liboqs	281
VS-1024 bin	vscrypto	281
NTRU 743	NTRUosp	256
VS 1024 gauss	vscrypto	247
VS 1024 gauss ct	vscrypto	247
VS 1024 uni	vscrypto	247
CLN16	Microsoft SIDH	192
CLN16 comp	Microsoft SIDH	192
VS 512 bin	vscrypto	131
NTRU 439	NTRUosp	131
McBits	Liboqs	129
RSA	OpenSSL	128
ECDH	OpenSSL	128
Frodo	Liboqs	114
VS 512 gauss	vscrypto	114
VS 512 gauss ct	vscrypto	114
VS 512 uni	vscrypto	114
BCNS15	Liboqs	86

Table 4.1: Security claims for the compared algorithms

Classical algorithms

In [5], NIST recommends to use 3072-bit prime numbers with RSA, and 256-bit curves with ECDH to achieve 128-bit security. In the tested code the ANSI X9.62 Prime 256v1 curve has been used.

LWE-based

BCNS15 In the paper [14], the authors suggest that polynomials of length $n = 1024$, the modulus $q = 2^{32} - 1$ and a Gaussian distribution with standard deviation $\sigma = 8/\sqrt{2\pi}$ should be used to obtain at least 128 bit classical security. Considering the distinguishing attack described in [46], and using a script from [2], they claim that these parameters provide a security level of 163.8 bits. Later, in [4], Alkim et al. performed a more pessimistic analysis of the scheme and considered the attacks in [19] and [75], and concluded that these parameters only provide 86-bit security.

New Hope In their paper, Alkim et al. also proposed two sets of parameters for their own version of the algorithm. One set use $n = 512$, $q = 12,289$ and a binomial distribution with standard deviation $\sigma = \sqrt{12}$ and provide 131-bit security according to their analysis. The other set is more secure and uses the parameters $n = 1024$, $q = 12,289$ and a binomial distribution with standard deviation $\sigma = \sqrt{8}$ to provide about 281 bits of security. They recommend using the latter set to be guaranteed to achieve more than 128-bit security even after more cryptanalysis has been performed, this is also the version used in the tests. They also provided a script in [3] to estimate the security of any similar scheme.

VScrypto In [81] and [82] Singh and Chopra present parameter sets for different levels of security for the implementations using uniform and Gaussian distributions. From these sets the ones which they claim provide 128 and 256 bits of classical security have been chosen. For 128-bit security they suggest the parameters $n = 512$ and $q = 25601$, and to achieve 256 bits of security they suggest $n = 1024$ and $q = 40961$, all using a distribution with standard deviation $\sigma = \sqrt{10}$. Their security claims refers to analyses from [19] and [46]. According to the script by Alkim et al., however, their parameter choices only achieve 114 and 247 bits of security respectively. Their implementations using a binomial distribution use the same parameters as the ones proposed by Alkim et al. and therefore achieve the same levels of security.

NTRU In [42], Hoffstein et al. suggest parameter sets for NTRU for different levels of security. They suggest to use $n = 439$ for 133-bit security and $n = 743$ for 256-bit security. Almost all their suggestions (including the two used here) use $q = 2048$ and $p = 3$. In their analysis they considered the attacks from [41] and [19].

Frodo When presenting the scheme Frodo in [13] the authors also suggested four of parameter sets and presented security estimates for these. The one considered in this thesis is called "recommended" and use a matrix of size $n = 752$, a modulus $q = 2^{12}$ and a rounded Gaussian distribution with standard deviation $\sigma = \sqrt{7/4}$. This set of parameters is claimed to provide 144 bits of security and in their analysis they consider the same attacks as Alkim et al. did in their analysis of New Hope.

SIDH

CLN16 Both De Feo et al. in [26] and Costello et al. in [22] proposed that a prime with a length close to 768 bits should be used to construct the field which the curves are defined over to achieve a classical security of 192 bits and a quantum security of 128 bits. Their analysis looked at the attacks from [87] and [92]. The implementation used in this thesis use a 751 bit prime, giving approximately the same security.

Code-based

McBits The implementation of McBits tested in this thesis use the parameters $n = 4096$, $k = 3352$ and $t = 62$. According to a script in [69] by Peters that is based on the paper [70], these give approximately 157 bits of classical security.

4.2 System description

All speed tests have been performed on a computer with a 3.2 GHz Intel Core i5-4460 processor and 8 GB RAM running Ubuntu 16.04.

4.3 Metrics

4.3.1 Efficiency

Speed

Cryptography needs to be fast to become widely adopted. A survey by Google in [78] shows that more than 50% of visits to mobile web-pages are abandoned if the page loads in more than three seconds. If the cryptography is one of the main reasons for slow loading times, there is a risk that it will be disabled in favor of performance.

In the case of Advenica's file encryption product it is meant to have high performance, meaning speed will be an important metric.

To benchmark the speed of the algorithms, a benchmarking suite from [83] by Douglas Stebila was used. The calculations of each party was run repeatedly for 10 seconds each and then the average runtime was calculated.

Data sent

The data sent between the parties to negotiate the key consists of public keys for key exchange algorithms, and public keys and ciphertexts for encryption algorithms. Having large public keys can cause problems for devices with constrained resources and limited bandwidth, and in the case of Advenica's file encryption product, where the keys are to be transferred via smart cards, the size is critical for the product to work.

The amount of data sent between the parties to negotiate a key have been calculated theoretically.

Memory usage

As many devices, e.g. IoT devices, have limited amounts of RAM. In order to be widely applicable, cryptographic algorithms cannot use too much memory since this would make them unusable in these constrained environments. This is also relevant for Advenica since their file encryption product is supposed to run in a constrained environment.

To measure the memory usage, the *Valgrind* [90] tool *Massif* was used. Each scheme was run once with the option –stacks=yes, to include memory allocated on the stack, and the peak was recorded for both heap, stack and total memory usage.

4.3.2 Implementation and deployment

Lines of code

While the numbers of lines of code needed to implement an algorithm can vary a lot between implementations, it might still give some information about the amount of code needed for a working implementation and the complexity of the algorithm. Fewer lines of code give less room for errors and is easier to check for correctness. This will thus make the code cheaper to write.

To measure the number of lines of code, the tool *cloc* [25] has been used.

Optimization potential

One way to optimize an algorithm is to create dedicated processor instructions. This has, for example, been done by Intel for the symmetric encryption algorithm AES (see [91]). If a specific part of an algorithm is found to consume a lot of time it might be possible to speed it up by using or building dedicated hardware.

To evaluate optimization potential, the profiling tool *Oprofile* [20] has been run to see what parts of the schemes are run the most.

Size of secret

Keeping data secret can be hard and might require special protected memory depending on how important the data is. Keeping this memory protected is often expensive and it is therefore preferable to minimize the amount of data that needs to stay secret.

The size of the secret data has been calculated theoretically.

Randomness

Randomness is an important part of secure cryptographic algorithms. Good (pseudo) random number generators can be slow and sometimes block the execution of an algorithm, and true randomness can be hard to find. If an algorithm can be secure with fewer random bits, this might make it easier to implement efficiently and to keep secure. Using too much randomness might cause problems, since some generators, like /dev/random in Unix-like operating systems, will block the execution if it cannot produce randomness with high enough quality.

The number of random bits used by the schemes was calculated by placing a counter in the function that was called to get randomness.

Mathematical theory

The complexity of the mathematical theory behind the algorithm might limit the rate of adoption. It might also make errors easier to introduce and harder to find.

This will be discussed in Chapters 5 and 6.

Hybrid solutions

To secure data before quantum-resistant cryptography can be completely trusted, a hybrid solution, using both classical and quantum-resistant algorithms, might be of interest. This would, however, require that the cost of using both algorithms together is acceptable.

The existence of effective hybrid solutions will be detailed in Chapter 5 and discussed further in Chapter 6.

Licenses and Patents

As NSA writes in [63], one thing that is limiting wider adoption of elliptic curve cryptography is that some implementations and techniques to increase performance are patented. Patents might make companies prefer other algorithms to avoid having to pay licence fees or to avoid legal problems.

4.4 Assurance

Special constants

In elliptic curve cryptography, randomly chosen curves cannot be used. Some curves have special properties that might make them easier to attack, and some have properties that makes encryption and decryption more efficient without compromising security. Choosing a good curve can be challenging, and for this reason NIST, among others, have released a set of standardized curves that they claim are both secure and efficient. This has the consequence that they have to be trusted, something that might not be preferable considering, in the case of NIST, the Snowden documents (see e.g. [53]). If some of these curves contain backdoors that a third party (e.g. NSA) can access, a lot of cryptography would be insecure. For this reason it is preferable if all parameters could be chosen at random without security or performance issues.

One possible solution to this problem is to use nothing-up-my-sleeve numbers. These are numbers for which it can be arugmented that they are chosen without ulterior motive. One example of this would be to choose digits from the decimal expansion of a known constant such as π or $\sqrt{2}$. While this concept might help to mitigate the risk of backdoors, it is argued in [8] that there are enough nothing-up-my-sleeve numbers to still enable an adversary to add a backdoor.

This will be discussed in Chapter 5.

Performed cryptanalysis

The more cryptanalysis that has been conducted, the smaller the risk is that new devastating attacks will be discovered. A large amount of cryptanalysis is essential for widespread adoption of a cryptographic algorithm, since it can have catastrophic consequences if it were to be broken.

This will be discussed in Chapter 6.

Chapter 5

Results

This chapter contains the results from the measurements and comparisons explained in the previous chapter. To make the tables slightly smaller and easier to get a good overview of, the results in some of the tables have been merged into one entry if they were the same. It should, however, still be easy to extract the result of each algorithm. Each table also contains a column that acts as a reminder about which type the algorithm is. To ease the reading the types are abbreviated, the abbreviations and their full names are shown in Table 5.1.

Abbreviation	**Full name**
M	Modular exponentiation
E	Elliptic curve
L	Lattice-based
L/L	Lattice/Learning with errors
L/R	Lattice/Ring learning with errors
S	Supersingular isogenic Diffie Hellman
C	Code-based

Table 5.1: abbreviations used in later tables.

5.1 Efficiency

5.1.1 Speed

The measurements of speed are shown in Table 5.2. The table is split into four columns, the first three are corresponding to the work done by the parties in each step and the fourth is the total time, which the table is sorted by. The encryption algorithms appear twice, once with key generation and once without, to emphasize that the key does not necessarily have to be regenerated when a new symmetric key should be established.

Scheme		**Time consumed (ms)**			
Name	**Cat.**	**Alice 0**	**Bob**	**Alice 1**	**Total**
VS 512 uni	L/R	0.03	0.05	0.02	0.1
MSRLN16	L/R	0.06	0.10	0.02	0.18
McBits (no keygen)	C	0.00	0.05	0.14	0.19
ECDH	E	0.03	0.10	0.07	0.2
New Hope	L/R	0.07	0.11	0.02	0.2
NTRU 439 (no keygen)	L	0.00	0.15	0.05	0.2
VS 1024 uni	L/R	0.06	0.11	0.04	0.21
VS 512 bin	L/R	0.07	0.12	0.03	0.22
NTRU 743 (no keygen)	L	0.00	0.19	0.11	0.3
VS 1024 bin	L/R	0.11	0.20	0.05	0.36
VS 512 gauss	L/R	0.16	0.26	0.02	0.44
NTRU 439	L	0.64	0.15	0.05	0.84
VS 1024 gauss	L/R	0.33	0.52	0.04	0.89
VS 512 gauss ct	L/R	0.35	0.54	0.02	0.91
NTRU 743	L	1.53	0.19	0.11	1.83
VS 1024 gauss ct	L/R	0.71	1.09	0.04	1.84
BCNS15	L/R	1.05	1.65	0.17	2.87
RSA (no keygen)	M	0.00	0.07	2.93	3.00
Frodo	L/L	3.61	4.14	0.11	7.86
McBits	C	126.24	0.05	0.14	126.43
RSA	M	329.97	0.07	2.93	332.97
CLN16	S	82.97	185.38	78.12	346.47
CLN16 (comp)	S	279.42	429.59	134.06	843.07

Table 5.2: The speed of the key exchanges.

5.1.2 Data sent

Table 5.3 shows how much data, in bytes, each party has to send to the other to establish a correct key. It is sorted according to the total amount of data sent. As for the table showing speed, the encryption algorithms appear twice to emphasize that the public key might not have to be send each time a new key should be negotiated.

Scheme		**Data sent (bytes)**		
Name	**Cat.**	**Alice $\rightarrow$ Bob**	**Bob $\rightarrow$ Alice**	**Total**
ECDH	E	32	32	64
McBits (no keygen)	C	0	141	141
RSA (no keygen)	M	0	384	384
NTRU 439 (no keygen)	L	0	604	604
CLN16 (comp)	S	328	330	658
RSA	M	387	384	771
NTRU 743 (no keygen)	L	0	1022	1022
CLN16	S	564	564	1128
NTRU 439	L	609	604	1213
VS 512 bin	L/R	896	960	1856
VS 512	L/R	960	1024	1984
NTRU 743	L	1027	1022	2049
VS 1024 bin	L/R	1792	1920	3712
MSRLN16	L/R	1824	2048	3872
New Hope	L/R	1824	2048	3872
VS 1024	L/R	2048	2176	4224
BCNS15	L/R	4096	4224	8320
Frodo	L/L	11280	11288	22568
McBits	C	311736	141	311877

Table 5.3: The amount of data sent between the parties.

5.1.3 Memory usage

Table 5.4 shows how much memory that was allocated by each implementation, both on the heap and on the stack. It is sorted according to the maximum total amount of space allocated at any time during the key exchange.

Scheme		**Peak memory consumption (byte)**		
Name	**Cat.**	**Heap**	**Stack**	**Total**
ECDH	E	6113	2296	8409
NTRU 439	L	7635	4560	12195
CLN16	S	3952	11592	15544
NTRU 734	L	11614	4448	16062
CLN16 (comp)	S	5200	15088	20288
RSA	M	25080	3744	27696
New Hope	L/R	7499	23832	31299
VS 512 gauss	L/R	1024	34648	35672
VS 512 gauss ct	L/R	1024	34648	35672
VS 512 bin	L/R	1024	34680	35704
MSRLN16	L/R	9517	30832	40349
VS 512 uni	L/R	1024	42024	43048
VS 1024 bin	L/R	1024	67576	68600
VS 1024 gauss	L/R	1024	67608	68632
VS 1024 gauss ct	L/R	1024	67608	68632
VS 1024 uni	L/R	1024	67608	68632
BCNS15	L/R	116969	11544	128513
Frodo	L/L	36845	99288	136341
McBits	C	319233	409544	728777

Table 5.4: The amount of memory allocated by each implementation.

5.2 Implementation and deployment

5.2.1 Lines of code

The lines of code of each implementation is shown in Table 5.5. The data is split into two columns, one named Code, containing the logic of the implementation, and one named Header, containing definitions of constants like parameter sets. The table is sorted in ascending order according to the total number of lines.

Scheme		**Lines of code**		
Name	**Cat.**	**Code**	**Header**	**Total**
New Hope	L/R	379	20	399
VS 512 uni	L/R	343	83	426
VS 512 gauss	L/R	386	83	469
VS 512 bin	L/R	392	83	475
VS 512 gauss ct	L/R	392	83	475
VS 1024 uni	L/R	343	147	490
Frodo	L/L	478	25	503
VS 1024 gauss	L/R	386	147	533
VS 1024 bin	L/R	392	147	539
VS 1024 gauss ct	L/R	392	147	539
MSRLN16	L/R	574	50	624
BCNS15	L/R	398	336	734
McBits	C	1297	13	1310
NTRU	L	1317	599	1916
CLN16	S	4001	286	4287

Table 5.5: The lines of code of each implementation.

5.2.2 Optimization potential

The two or three most time consuming operations from the profiling tests are shown in Table5.6.

Name	**Cat.**	**Most**	**Second**	**Third**
BCNS15	L/R	Sampling (64%)	FFT multiplication (23%)	Random number generation (12%)
New Hope	L/R	NTT (29%)	Random number generation (23%)	Sampling (15%)
MSRLN16	L/R	Forward NTT (27%)	Random number generation (19%)	Inverse NTT (12%)
Frodo	L/L	Matrix multiplication (82%)	AES (used during key generation) (10%)	
NTRU 439	L	Polynomial multiplication (70%)	Polynomial inversion (18%)	
NTRU 734	L	Polynomial multiplication (74%)	Polynomial inversion (18%)	
CLN16	S	Multiplication (60%)	Montgomery reduction (33%)	
CLN16 (comp)	S	Multiplication (59%)	Montgomery reduction (34%)	
McBits	Cat	Gaussian elimination (84%)	Matrix multiplication (11%)	
VS 521 bin	L/R	Sampling (38%)	Forward FFT (11%)	Backward FFT (11%)
VS 521 uni	L/R	Forward FFT (24%)	Backward FFT (20%)	Sampling (19%)
VS 521 gauss	L/R	Sampling (66%)	Forward FFT (9%)	Backward FFT (8%)
VS 521 gauss ct	L/R	Sampling (89%)	Forward FFT (4%)	Backward FFT (2%)
VS 1024 bin	L/R	Sampling (34%)	Forward FFT (23%)	Backward FFT (11%)
VS 1024 uni	L/R	Forward FFT (31%)	Backward FFT (18%)	Sampling (14%)
VS 1024 gauss	L/R	Sampling (63%)	Forward FFT (14%)	Backward FFT (8%)
VS 1024 gauss ct	L/R	Sampling (89%)	Forward FFT (5%)	Backward FFT (2%)

Table 5.6: The most time consuming operations for each implementation.

5.2.3 Size of secret

The sizes of the secrets are shown in Table 5.7. It is split up into how much each party has to keep secret and the total amount of secret data. It is sorted in ascending order according to the total amount of secret data.

Scheme		**Size of secret (bytes)**		
Name	**Cat.**	**Alice**	**Bob**	**Total**
ECDH	E	32	32	64
RSA	M	384	0	384
CLN16	S	235	236	471
NTRU 439	L	714	604	1,318
VS 512 bin	L/R	768	1,152	1,920
NTRU 743	L	1,202	1,022	2,224
VS 512 gauss	L/R	896	1,344	2,240
VS 512 gauss ct	L/R	896	1,344	2,240
VS 512 uni	L/R	896	1,344	2,240
New Hope	L/R	1,280	1,920	3,200
MSRLN16	L/R	1,280	1,920	3,200
VS 1024 bin	L/R	1,280	1,920	3,200
VS 1024 gauss	L/R	1,792	2,688	4,480
VS 1024 gauss ct	L/R	1,792	2,688	4,480
VS 1024 uni	L/R	1,792	2,688	4,480
BCNS15	L/R	1,792	2,688	4,480
McBits	C	5,984	0	5,984
Frodo	L/L	6,016	6,272	12,288

Table 5.7: The size of the secrets for each implementation.

5.2.4 Randomness

The amount of randomness needed by each algorithm (in bytes) is shown in Table 5.8. It is sorted in ascending order based on the amount of randomness needed.

Name	Cat.	Randomness used (byte)
CLN16	S	95
NTRU 439	L	292
NTRU 743	L	576
VS 512 uni	L/R	1896
VS 1024 uni	L/R	3760
VS 512 bin	L/R	15400
MSLNR15	L/R	15424
VS 1024 bin	L/R	20520
New Hope	L/R	20544
Frodo	L/L	36192
McBits	C	36352
VS 512 gauss	L/R	61768
VS 512 gauss-ct	L/R	61768
VS 1024 gauss	L/R	123528
VS 1024 gauss-ct	L/R	123528
BCNS15	L/R	247552

Table 5.8: The amount of randomness used by each implementation.

5.2.5 Mathematical theory

Lattice-based The lattice problems that these algorithms are based on are fairly easy to understand, and LWE is just a system of equations with small errors introduced. The operations used are only basic ones like addition, multiplication and inversion of polynomials and matrices. While the schemes are slightly more complicated than RSA and Diffie-Hellman, they are not as complex as elliptic curves, and basic version should be fairly easy to implement.

Code-based The coding theory behind the code-based cryptosystems is more complicated than the theory behind the lattice-based methods. It is still quite easy to grasp and most of the operations used are basic ones on matrices.

SIDH The abstract algebra and algebraic geometry behind SIDH is a lot more complicated than that of the other algorithms. The calculation of points on the curve and of the j-invariant are fairly straightforward steps, but generation of new supersingular elliptic curves and the theory behind the j-invariant are a lot more complicated.

5.2.6 Hybrid solutions

In [22], in addition to their SIDH scheme, the authors presented a hybrid solution combining SIDH with ECDH. They also claim that this hybrid solution would give a total of 384 bits of classical security with only about 17% increased key size and 13% increased runtime. The other algorithms does not seem to have any particular advantages that would make a hybrid solution easier. In an experiment with quantum-resistant cryptography in the alpha version of Google Chrome mentioned in [15], Google uses New Hope in combination with elliptic curves.

5.2.7 Licenses and Patents

None of the compared algorithms are patented. NTRU was patented, but these patents were released into the public domain under the creative commons licence in march 2017 (see [77]). There exist other patents on LWE schemes, like [29], but these are different enough for Bos et al. to claim that "There are no known patents covering R-LWE" in [14].

5.3 Assurance

5.3.1 Special constants

The only algorithms containing predetermined constants that might be special in the same way as the curves in ECDH are the ring-LWE scheme BCNS15, the SIDH based schemes and the code based schemes.

BCNS15 The parameter that has a risk of being a special constant in this implementation is the polynomial $\mathbf{a}$. That it is replaced by a randomly chosen polynomial in New Hope provides proof that this is not the case, and the only risk is that it is not regenerated often enough or chosen to make all-for-one attacks easier.

SIDH The supersingular elliptic curve used might be such a special constant, but in [26] De Feo et al. presents a relatively simple algorithm for finding supersingular elliptic curves with sufficient cardinality. While future cryptanalysis might reveal attacks exploiting a certain structure, this is currently not the case.

Code-based The McEliece and Niederreiter cryptosystems work with any type of code, and different types have been tried. Most of these, like the quasi cyclic codes mentioned earlier, have properties that can be exploited by an attacker though. However, as long as a binary Goppa code are used the system seems to be secure.

Chapter 6

Discussion

In this chapter the results will be discussed. The algorithms will be referred to by both name and category. As a quick reminder, the tested algorithms and their categories are:

- **Lattice-based**; BCNS15, New Hope, MSRLN16, Frodo, NTRU, and all algorithms from VS crypto.
- **LWE-based**; All lattice-based except NTRU.
- **Ring LWE-based**; All LWE-based expect Frodo.
- **SIDH**; CLN16 and CLN16 comp.
- **Code-based**; McBits.

6.1 Efficiency

6.1.1 Speed

Most of the tested implementations performed well on the speed test, especially the lattice-based methods. There are, however, some noticeable results.

- The encryption algorithms perform overall worse than the key exchange algorithms when the key generation was included in the calculations. This might not be a problem since the keys can be reused for multiple key exchanges, and considering that RSA has slower key generation than all other algorithms this problem is obviously surmountable. A slight problem with slow key generation is that the keys would probably be used for longer, thereby increasing the amount of communication affected if the key were to be stolen.
- CLN16 is by far the slowest key exchange algorithm, especially with compression, and is more than a thousand times slower than the fastest one. This could partly be compensated for by the small key size, but most systems would probably have easier to compensate for a slightly larger key than for this kind of decline in performance and it is therefore only applicable to certain systems where the size of the key is very important and the speed is not. Microsoft also provides an architecture specific implementation that

ran about five times faster when tested. This version was not considered in the tests since it is architecture specific and not as portable as the other implementations.

- Both schemes based on ring-LWE and NTRU perform really well and even though they provide more security, they are just as fast as ECDH.
- The effect of the ring structure on the lattice-based schemes is significant and if it is removed the scheme gets about 40 times slower. While this is better than CLN16, it would still be hard to cope with for many applications.

6.1.2 Data sent

The key size and amount of data that need to be communicated to negotiate a key is the most problematic area for almost all quantum-resistant algorithms, and no algorithm can really compare to ECDH and only a few to RSA. McBits comes close when the key can be reused, but the size of the key is large enough to cause problem when it has to be distributed, even if it does not happen that often. Other noticeable results are

- The encryption algorithms perform overall better than the key exchange algorithms when the public key has already been distributed.
- CLN16, especially with compressed keys, performs well here, and is the only key exchange algorithm that is comparable to the ciphertexts of the encryption algorithms.
- NTRU is in the top here as well, even when the key distribution is included, and is almost as good as RSA.
- This is the area where the LWE-based methods perform the worst. The ring structure helps to bring down the key size significantly, but it is still not enough to be comparable to RSA or ECDH.

6.1.3 Memory usage

The increase in memory usage is at an overall more acceptable level and most require at most twice of what RSA requires, and NTRU is almost at the same level as ECDH. Most noticeable here is that McBits needed about 700 MB of memory, and while certainly not preferable, it might be acceptable in a computer with 8 or 16 GB RAM. In a more constrained environment like a Raspberry pi, which only has 1 GB of RAM, this might cause serious problem though.

6.1.4 Summary

Overall the lattice-based algorithms performed best on these three metrics, with a slight advantage for NTRU due to the lower size of key/ciphertext and lower memory usage. The increased key size of the ring LWE-based schemes is a problem, but is partly compensated for by the fast speeds. Frodo performed in the bottom part on all metrics, and its performance is probably not good enough for many applications. If the ring structure of ring LWE turn out to be insecure it does however provide the best trade-off between speed and key size among the compared quantum-resistant algorithms. McBits and CLN16 are strong contenders in some areas, but have weaknesses that, at least right now, makes them useless in others. The fast encryption and decryption, and small ciphertexts makes McBits great for fast communications, but the large keys and memory usage limits the applicability in constrained environments. The small keys of CLN16 and the availability of good compression makes it a great option for constrained environments, but the slow speeds will cause problem if performance is a requirement.

Since Advenica needs an algorithm that can perform well in constrained environments NTRU is probably the best candidate from a performance point of view.

6.2 Implementation and deployment

6.2.1 Lines of code and mathematical theory

The measurement of the number of lines of code might be a bit unfair since the quality of the implementations and the included features vary. Table 5.5 contains an attempt at filtering out equally relevant code. While this still might not give a completely fair picture of the amount of code needed for a working implementation, it might provide some clue as to how complex the algorithms are. Most notable is probably that the implementation of CLN16 contain a lot more code than the others and that the implementations of the LWE-based key exchange methods contain the least. This also matches the fact that SIDH is built on a more complex mathematical theory and uses more advanced operations than the basic operations on polynomials used in the LWE-based algorithms. The increase in both complexity of theory and lines of code would likely make an implementation of SIDH more expensive to write since it would probably be easier to introduce bugs and harder to find them among the large amount of code. It could also affect the adoption since it might feel more secure to use an algorithm that you understand how it works and that you are able to implement for yourself, something that might not be possible with SIDH if you do not have a degree in mathematics.

6.2.2 Randomness

The amount of randomness needed depends largely on the distribution that is used, and all implementations in the bottom of the table use a Gaussian distribution. The more notable results can be found in the other end of the table where CLN16, that only needs a small amount of randomness for to produce the secret points used to construct the curves, and NTRU, that only randomizes polynomials with small coefficients, are found.

6.2.3 Optimization potenial

The sampling method is an important part, at least for the LWE-based implementations, which can be seen from the profiling tests where sampling and random number generation takes up a significant part of the total runtime, especially when a Gaussian distribution is used. Other than that, most time is spent on multiplication, or trying to speed up multiplication by transformations. Techniques for speeding up the multiplications are used in all of the tested implementations, but further improvements in this area would probably have the largest impact on most of the algorithms. An important detail related to optimization that cannot be seen in the profiling test is that Microsoft provided architecture specific versions of CLN16 that run faster than the portable C versions that were tested. This signifies that there are ways to improve the algorithm and reach more reasonable speeds. An optimized version of New Hope has also been released by the authors of [86]. This version is optimized to take advantage of the parallelism of the ARMv8-A architecture and according to their measurements it runs about 8 times faster than the implementation that was tested here.

6.2.4 Size of secret

The size of the secret data is often linked to the size of the public key since the two must share certain properties for everything to work as expected, and are often derived from one another. A notable difference, though, is that McBits, while still larger than most, does not have the largest secrets and is not as far behind the other as when considering the public key. Instead, Frodo comes in at the last place due to having to keep multiple matrices secret. This could make security critical implementations more expensive since the protected memory needed to store the secrets securely is more expensive. Deleting secrets securely is not as easy as just deleting either, instead you have to explicitly reset that part of the memory, eg. by using a function like *memset* in C.

6.2.5 Summary

As when considering efficiency, the lattice-based algorithms, and especially NTRU perform well when considering factors important for implementation. The biggest problems for them are probably that Frodo requires a lot of data to stay secret. SIDH also stand out with its low resource requirements and the possibility of efficient hybrid implementations, but has the drawback of being more complicated and requiring more code than the others.

For Advenica NTRU is probably the best choice from an implementation point of view as well due to its relatively low resource requirements and simplicity.

6.3 Assurance

Being secure and that this fact can be trusted is probably the most important quality a cryptographic algorithm can have. This could cause some problem for quantum-resistant cryptography since it is a relatively new field. An exception to this is the original code-based systems that have been around for 30-40 years and should therefore be able to be considered secure. The newer improvements of the systems such as quasi cyclic MDPC codes does not have this property and according to Bernstein and Lange in [11] the original systems by McEliece and

Niederreiter are the only ones that have received enough study to be recommended for adoption.

The reduction by Regev from LWE to lattice problems mentioned in Section 2.4.3 provides a strong security proof for LWE. The ring structure and the reduction to ideal lattices weakens this proof slightly. Similar proofs does not exist for NTRU, but the fact that is has survived for 20 years is promising. A version of NTRU that does have security proofs was presented in [85]. This version was however shown in [17] to be about 100 times slower and provide 100 times larges keys than the regular NTRU, a large drawback that is hard to motivate. Quantum attacks like the ones mentioned in [24] that target principal ideal lattices and some problems on ideal lattices might raise some concern about the ring structure, but they are not applicable to any of the tested schemes.

No attacks or special constants exist for SIDH-based systems. This could be due the fact that they were first proposed in their current form only 6 years ago and that more research is devoted to code- and lattice-based methods due to their superior performance.

From an assurance point of view the code-based algorithms look like the best option for Advenica's file encryption product due to the considerable amount of scrutiny it has received. The algorithms based on LWE are good candidates as well due to their superior security proofs.

6.4 Summary

6.4.1 NTRU

NTRU provides the best trade-offs between speed and key size and is in the top on almost every metric. It is a bit slower than the fastest ring-LWE algorithms due to the key generation, but the smaller keys and lower resource requirements compensates for this. The mathematics behind it is at the same level as for the algorithms based on ring LWE, so the increased code size could depend on the fact that the implementation is closer to production quality than the others. The weaker security proofs could be considered a flaw, but the fact that it has survived about 20 years without any significant attack is promising.

6.4.2 Ring LWE-based

The algorithms based on ring LWE are the best when only considering speed. They do however have larger keys and overall higher resource requirements, which makes them slightly less suitable for adoption, especially into constrained environments. The relatively short and simple implementations are, however, an advantage, as well as the security proofs.

6.4.3 LWE-based

The performance of Frodo is significantly lower than that of the others and optimizations like the ring structure are definitely needed for widespread adoption. That the security proofs are even stronger than for ring LWE is a clear advantage, and if a severe attack on the ideal lattices that ring-LWE and NTRU is based upon is found, Frodo gives the best trade-off between speed and key size. It would however still not be good enough for widespread adoption and other algorithms would most certainly be needed.

6.4.4 SIDH

When considering key size and resource requirements SIDH is the clear winner. It is however very slow and improvements in this area will be needed for implementations in applications with any kind of speed requirements. That it is far more complex and requires a lot more code to implement is also a disadvantage, but something that might be easier to overcome. The largest issue for adoption at the moment is probably the lacking amount of security. This is mostly because it is a relatively new scheme and a lot of research has been devoted to the lattice based schemes due to their superior performance.

6.4.5 Code-based

The code-based algorithms are fast and provide small ciphertexts, which is promising for fast communication. They do, however suffer from incredibly large keys which affects both key generation and memory usage. This makes them hard to implement in constrained environments, but when the key size can be handled they are a viable option. The most significant advantage is that the original schemes by McEliece and Niederreiter have survived for 30-40 years and could therefore be considered secure.

6.5 Industry adoption

Replacing all of the currently used cryptography with quantum-resistant algorithms today will be a tough challenge. Code-based systems might be a good replacement in some areas thanks to the small ciphertexts, fast encryption and decryption, and well studied security. The large key size could in part be acceptable for web-pages since they have an average size of 2600 kB according to [44]. The public keys will thus only increase the total traffic by 10-15% when they have to be sent. The problems would however still exist in constrained environments, where they would probably be insurmountable, especially the high memory usage. The use of for example quasi cyclic codes could be part of a solution, but from a security perspective they do not seem to be a viable option at the moment.

The key and resource problems could be solved by SIDH, but the slow speeds would probably become a problem if the system also has high performance demands. The lack of rigorous study is also a significant disadvantage of the algorithm. The risk of attacks from classical computers can be solved by a hybrid solution with ECDH, but the risk of quantum attacks being developed will still exist.

The best solution would probably be one of the lattice-based systems since they performed good overall on all metrics. When selecting which lattice-based system one would have to consider the advantages and disadvantages of each one, and compare with the requirements of the application. The ones based on ring-LWE are faster than the others and have strong security proofs, but require more resources and have larger keys. NTRU is the most resource efficient and the oldest, but does not have the same level of security proofs. If the application allows it, an option could be to remove the ring structure to remove that attack surface, but the cost in reduced performance is probably too great for most applications.

When considering the parameters needed to achieve long term quantum security all parameter sets that provide at least 128 bit quantum security should be sufficient. This is true for the high security options tested for the lattice-based algorithms and for the SIDH implementation. The quantum security of the parameters used by McBits was, however, not discussed in the paper and further increases in security would probably be necessary. This would increase the size of the public key even more, thereby making it even harder to handle.

In the case of Advenica's file encryption system the currently available best solution would probably be a lattice-based method. Due to the lower key size NTRU would probably be preferable, but a ring LWE based key exchange could be a viable option to increase the speed slightly at the cost of a higher resource consumption.

6.6 Improvements

All of the compared algorithms have some advantages, but theoretical improvements would make industry adoption easier. The code-based systems would benefit greatly from smaller keys, and quasi cyclic codes could be a good way to go if the security issues could be solved. This would probably still not make it applicable enough for constrained environments and other algorithms will be needed in that area.

The low resource requirement of SIDH makes it a good contender for constrained environments, but it is a lot slower than the other algorithms and probably too slow for application with high performance requirements. The architecture specific implementations are a promising way of speeding it up, but in its current form it is still not enough to be able to match the other algorithms.

The lattice-based systems with the ring structure are adoptable without any significant improvements and would probably be able to perform well in most applications. The increase in key size compared to ECDH is significant and might require some changes in existing applications, it is probably manageable in most cases though. Frodo proves that needs more improvements to be able to compete with the other algorithms, both in speed and key size. The ring structure is one such improvement (that was purposely removed from Frodo), but other improvements that does not rely on ideal lattices could be useful, at least as a backup if algorithms like the ones mentioned in [24] are found for more problems.

Due to NTRU and ring LWE no new algorithms are needed and improvements to existing ones will most certainly be enough. If they prove to be insecure, however, no algorithms are close to them in performance and completely new ones with reasonable trade-off between speed and key size will probably be needed.

Chapter 7

Conclusions

The biggest obstacle for a transition to quantum-resistant algorithms is probably the increase in key size, since none of the algorithms can match ECDH on this metric. To cope with this change, some applications would presumably need infrastructural changes, but in most cases, including Advenica's, the increase would be within a reasonable level.

When considering specific algorithms, they all have advantages and disadvantages.

- The code-based systems have small ciphertexts, fast performance and have survived rigorous study, but they also have large keys and high resource requirements.
- SIDH uses small keys and have low resource requirements, but is a considerably slower than the other algorithms and is a very new invention.
- The lattice-based schemes have fast performance but larger keys than the currently used algorithms, and the ring structure of the faster versions might raise some security concerns.

The algorithm most suitable for widespread adoption would probably be NTRU due to it performing in the top on all metrics, and comparable to the currently used algorithms in most. These qualities would also make it a great option for Advenica's file encryption software. The ring LWE-based key exchange algorithm could also be a possible contender for the same reasons, but the larger keys makes NTRU preferable. The attacks on similar schemes is a bit concerning, but as of now they do not seem to cause any problem. If they nevertheless would, the transition to quantum algorithms would become more troublesome. The other algorithms have some advantages, but their disadvantages are too large for them to be applied widely and new algorithms would probably be needed.

7.1 Future work

The field of post quantum cryptography is currently in a phase where a lot is happening. This thesis provides a review of the advantages and disadvantages of the currently relevant algorithms, but as new improvements and algorithms are developed they need to be evaluated as well. A study of quantum secure algorithms for signing could also be relevant as the development of quantum computers progresses.

References

[1] M. Ajtai, “The shortest vector problem in l2 is np-hard for randomized reductions (extended abstract),” in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, ser. STOC ’98. New York, NY, USA: ACM, 1998, pp. 10–19. [Online]. Available: http://doi.acm.org/10.1145/276698.276705 [Cited on page 14.]

[2] M. R. Albrecht, R. Player, and S. Scott, “On the concrete hardness of learning with errors,” Cryptology ePrint Archive, Report 2015/046, 2015, http://eprint.iacr.org/2015/046. [Cited on page 31.]

[3] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, “Pqsecurity,” accessed 2017-04-24. [Online]. Available: https://github.com/tpoeppelmann/newhope/blob/master/scripts/PQsecurity.py [Cited on page 32.]

[4] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, “Post-quantum key exchange - a new hope,” Cryptology ePrint Archive, Report 2015/1092, 2015, http://eprint.iacr.org/2015/1092. [Cited on pages 19 and 31.]

[5] E. Barker, “Recommendation for key management part 1: General,” NIST, 2016. [Online]. Available: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-57pt1r4.pdf [Cited on pages 30 and 31.]

[6] E. Berlekamp, “Goppa codes,” *IEEE Transactions on Information Theory*, vol. 19, no. 5, pp. 590–592, September 1973, DOI: 10.1109/TIT.1973.1055088. [Cited on page 24.]

[7] D. J. Bernstein, J. Buchmann, and E. Dahmen, *Post-quantum cryptography.* Springer, 2009, ISBN: 978-3-540-88702-7. [Cited on page 17.]

[8] D. J. Bernstein, T. Chou *et al.*, “How to manipulate curve standards: a white paper for the black hat,” Cryptology ePrint Archive, Report 2014/571, 2014, http://eprint.iacr.org/2014/571. [Cited on page 35.]

[9] D. J. Bernstein, T. Chou, and P. Schwabe, “Mcbits: fast constant-time code-based cryptography,” Cryptology ePrint Archive, Report 2015/610, 2015, http://eprint.iacr.org/2015/610. [Cited on page 26.]

[10] D. J. Bernstein, N. Heninger, P. Lou, and L. Valenta, “Post-quantum rsa,” Cryptology ePrint Archive, Report 2017/351, 2017, http://eprint.iacr.org/2017/351. [Cited on page 10.]

[11] D. J. Bernstein and T. Lange, “Post-quantum cryptography—dealing with the fallout of physics success,” Cryptology ePrint Archive, Report 2017/314, 2017, http://eprint.iacr.org/2017/314. [Cited on page 50.]

[12] D. J. Bernstein, T. Lange, and C. Peters, “Attacking and defending the mceliece cryptosystem,” Cryptology ePrint Archive, Report 2008/318, 2008, http://eprint.iacr.org/2008/318. [Cited on page 15.]

[13] J. Bos, C. Costello *et al.*, “Frodo: Take off the ring! practical, quantum-secure key exchange from lwe,” Cryptology ePrint Archive, Report 2016/659, 2016, http://eprint.iacr.org/2016/659. [Cited on pages 20 and 32.]

[14] J. W. Bos, C. Costello, M. Naehrig, and D. Stebila, “Post-quantum key exchange for the tls protocol from the ring learning with errors problem,” Cryptology ePrint Archive, Report 2014/599, 2014, http://eprint.iacr.org/2014/599. [Cited on pages 18, 31, and 46.]

[15] M. Braithwaite, “Google security blog: Experimenting with post-quantum cryptography,” July 2016, accessed 2017-05-16. [Online]. Available: https://security.googleblog.com/2016/07/experimenting-with-post-quantum.html [Cited on page 46.]

[16] D. Cabarcas, F. Göpfert, and P. Weiden, “Provably secure lwe encryption with smallish uniform noise and secret,” in *Proceedings of the 2Nd ACM Workshop on ASIA Public-key Cryptography*, ser. ASIAPKC ’14. New York, NY, USA: ACM, 2014, pp. 33–42. [Online]. Available: http://doi.acm.org/10.1145/2600694.2600695 [Cited on page 21.]

[17] D. Cabarcas, P. Weiden, and J. Buchmann, *On the Efficiency of Provably Secure NTRU*. Cham: Springer International Publishing, 2014, pp. 22–39. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11659-4_2 [Cited on page 51.]

[18] M. Campagna, L. Chen *et al.*, “Quantum safe cryptography and security,” 2015, accessed 2017-03-31. [Online]. Available: http://www.etsi.org/images/files/ETSIWhitePapers/QuantumSafeWhitepaper.pdf [Cited on page 5.]

[19] Y. Chen and P. Q. Nguyen, “Bkz 2.0: Better lattice security estimates,” in *ASIACRYPT*, ser. Lecture Notes in Computer Science, vol. 7073. Springer, 2011, pp. 1–20, DOI: 10.1007/978-3-642-25385-0_1. [Cited on pages 31 and 32.]

[20] W. Cohen, S. Suthikulpanit *et al.*, “Oprofile,” accessed 2017-05-15. [Online]. Available: http://oprofile.sourceforge.net/news/ [Cited on page 34.]

[21] C. Costello, D. Jao *et al.*, “Efficient compression of sidh public keys,” Cryptology ePrint Archive, Report 2016/963, 2016, http://eprint.iacr.org/2016/963. [Cited on pages 23 and 29.]

[22] C. Costello, P. Longa, and M. Naehrig, “Efficient algorithms for supersingular isogeny diffie-hellman,” Cryptology ePrint Archive, Report 2016/413, 2016, http://eprint.iacr.org/2016/413. [Cited on pages 23, 29, 32, and 46.]

[23] N. Courtois, A. Klimov, J. Patarin, and A. Shamir, "Efficient algorithms for solving overdefined systems of multivariate polynomial equations," in *Proceedings of the 19th International Conference on Theory and Application of Cryptographic Techniques*, ser. EUROCRYPT'00. Berlin, Heidelberg: Springer-Verlag, 2000, pp. 392–407. [Online]. Available: http://dl.acm.org/citation.cfm?id=1756169.1756206 [Cited on page 15.]

[24] R. Cramer, L. Ducas, and B. Wesolowski, "Short stickelberger class relations and application to ideal-svp," Cryptology ePrint Archive, Report 2016/885, 2016, http://eprint.iacr.org/2016/885. [Cited on pages 51 and 53.]

[25] A. Danial, "Cloc," accessed 2017-05-12. [Online]. Available: https://github.com/AlDanial/cloc [Cited on page 34.]

[26] L. De Feo, D. Jao, and J. Plût, "Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies," Cryptology ePrint Archive, Report 2011/506, 2011, http://eprint.iacr.org/2011/506. [Cited on pages 15, 23, 32, and 46.]

[27] M. H. Devoret and R. J. Schoelkopf, "Superconducting circuits for quantum information: An outlook," *Science*, vol. 339, no. 6124, pp. 1169–1174, 2013. [Online]. Available: http://science.sciencemag.org/content/339/6124/1169 [Cited on page 5.]

[28] W. Diffie and M. Hellman, "New directions in cryptography," *IEEE Transactions on Information Theory*, vol. 22, no. 6, pp. 644–654, Nov 1976, DOI:10.1109/TIT.1976.1055638. [Cited on page 10.]

[29] J. Ding, "Cryptographic systems using pairing with errors," Jan. 26 2016, uS Patent 9,246,675. [Online]. Available: https://www.google.com/patents/US9246675 [Cited on page 46.]

[30] A. Einstein, B. Podolsky, and N. Rosen, "Can quantum-mechanical description of physical reality be considered complete?" *Phys. Rev.*, vol. 47, pp. 777–780, May 1935, DOI: 10.1103/PhysRev.47.777. [Cited on page 12.]

[31] M. Ekerå and J. Håstad, "Quantum algorithms for computing short discrete logarithms and factoring rsa integers," Cryptology ePrint Archive, Report 2017/077, 2017, http://eprint.iacr.org/2017/077. [Cited on page 13.]

[32] T. Fabsic, V. Hromada *et al.*, "A reaction attack on the qc-ldpc mceliece cryptosystem," Cryptology ePrint Archive, Report 2017/494, 2017, http://eprint.iacr.org/2017/494. [Cited on page 25.]

[33] P. Gaborit, "Shorter keys for code based cryptography," 2005, pp. 81–90. [Cited on page 25.]

[34] M. R. Garey and D. S. Johnson, *Computers and Intractability; A Guide to the Theory of NP-Completeness.* New York, NY, USA: W. H. Freeman & Co., 1990, ISBN: 0716710455. [Cited on page 15.]

[35] A. Gautam, A. Goyaly, R. Munshiz, and V. Chhabra, "Post-quantum primitives for constrained environments," 2016. [Online]. Available: https://github.com/gautamgitspace/Post-Quantum-Security-Algorithms/blob/master/PostQuantumPrimitives.pdf [Cited on page 7.]

[36] V. D. Goppa, "A new class of linear error-correcting codes," *Probl. Peredachi Inf.*, vol. 6, pp. 24–30, 1970. [Online]. Available: http://mi.mathnet.ru/eng/ppi1748 [Cited on page 24.]

[37] L. K. Grover, "A fast quantum mechanical algorithm for database search," in *Proceedings of the Twenty-eighth Annual ACM Symposium on Theory of Computing*, ser. STOC '96. New York, NY, USA: ACM, 1996, pp. 212–219. [Online]. Available: http://doi.acm.org/10.1145/237814.237866 [Cited on pages 11 and 12.]

[38] C. Gu, "Cryptanalysis of simple matrix scheme for encryption," Cryptology ePrint Archive, Report 2016/1075, 2016, http://eprint.iacr.org/2016/1075. [Cited on page 27.]

[39] Q. Guo, T. Johansson, and P. Stankovski, "A key recovery attack on mdpc with cca security using decoding errors," Cryptology ePrint Archive, Report 2016/858, 2016, http://eprint.iacr.org/2016/858. [Cited on page 25.]

[40] B. Hensen, H. Bernien *et al.*, "Loophole-free bell inequality violation using electron spins separated by 1.3 kilometres," *Nature*, vol. 526, p. 682–686, 2015, DOI: 10.1038/nature15759. [Cited on page 12.]

[41] J. Hoffstein, J. H. Silverman, and W. Whyte, "Meet-in-the-middle attack on an ntru private key," Tech. Rep., 2006. [Cited on page 32.]

[42] J. Hoffstein, J. Pipher *et al.*, "Choosing parameters for ntruencrypt," Cryptology ePrint Archive, Report 2015/708, 2015, http://eprint.iacr.org/2015/708. [Cited on page 32.]

[43] J. Hoffstein, J. Pipher, and J. H. Silverman, *NTRU: A ring-based public key cryptosystem.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 267–288. [Online]. Available: http://dx.doi.org/10.1007/BFb0054868 [Cited on page 21.]

[44] http archive, "Interesting stats," accessed 2017-05-07. [Online]. Available: http://www.httparchive.org/interesting.php?a=All&l=Apr%2015%202017 [Cited on page 52.]

[45] R. Johansson and T. Strahl, "Post-quantum secure communication on a low performance iot platform," Master's thesis, Lund University, 2016, http://www.eit.lth.se/sprapport.php?uid=973. [Cited on page 7.]

[46] T. Lepoint and M. Naehrig, "A comparison of the homomorphic encryption schemes fv and yashe," Cryptology ePrint Archive, Report 2014/062, 2014, http://eprint.iacr.org/2014/062. [Cited on pages 31 and 32.]

[47] R. Lindner and C. Peikert, *Better Key Sizes (and Attacks) for LWE-Based Encryption.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 319–339. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-19074-2_21 [Cited on page 20.]

[48] P. Longa and M. Naehrig, "Speeding up the number theoretic transform for faster ideal lattice-based cryptography," Cryptology ePrint Archive, Report 2016/504, 2016, http://eprint.iacr.org/2016/504. [Cited on page 20.]

[49] V. Lyubashevsky, C. Peikert, and O. Regev, “On ideal lattices and learning with errors over rings,” *J. ACM*, vol. 60, no. 6, pp. 43:1–43:35, Nov. 2013. [Online]. Available: http://doi.acm.org/10.1145/2535925 [Cited on pages 15 and 18.]

[50] C. Löndahl, T. Johansson, M. Koochak Shooshtari, M. Ahmadian-Attari, and M. Aref, “Squaring attacks on mceliece public-key cryptosystems using quasi-cyclic codes of even dimension,” *Designs, Codes, and Cryptography*, vol. 80, no. 2, pp. 359–377, 2016, DOI: 10.1007/s10623-015-0099-x. [Cited on page 25.]

[51] T. Matsumoto and H. Imai, *Public Quadratic Polynomial-Tuples for Efficient Signature-Verification and Message-Encryption.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1988, pp. 419–453. [Online]. Available: http://dx.doi.org/10.1007/3-540-45961-8_39 [Cited on page 27.]

[52] R. J. McEliece, “A public-key cryptosystem based on algebraic coding theory,” *Jet Propulsion Laboratory DSN Progress Report*, pp. 114–116, 1978. [Online]. Available: http://ipnpr.jpl.nasa.gov/progressreport2/42-44/44N.PDF [Cited on page 24.]

[53] J. Menn, “Exclusive: Secret contract tied nsa and security industry pioneer,” 2013, accessed 2017-04-23. [Online]. Available: http://www.reuters.com/article/us-usa-security-rsa-idUSBRE9BJ1C220131220 [Cited on page 35.]

[54] D. Micciancio, “The shortest vector in a lattice is hard to approximate to within some constant,” in *Proceedings 39th Annual Symposium on Foundations of Computer Science (Cat. No.98CB36280)*, Nov 1998, pp. 92–98, DOI: 10.1109/SFCS.1998.743432. [Cited on page 14.]

[55] D. Micciancio, “Duality in lattice cryptography,” Public key Cryptography, 2010, invited talk. [Cited on page 20.]

[56] “Lattice cryptography library,” Microsoft research, accessed 2017-04-24. [Online]. Available: https://www.microsoft.com/en-us/research/project/lattice-cryptography-library/ [Cited on page 7.]

[57] “Sidh library,” Microsoft research, accessed 2017-04-24. [Online]. Available: https://www.microsoft.com/en-us/research/project/sidh-library/ [Cited on pages 7 and 29.]

[58] R. Misoczki, J.-P. Tillich, N. Sendrier, and P. S. L. M. Barreto, “Mdpc-mceliece: New mceliece variants from moderate density parity-check codes,” Cryptology ePrint Archive, Report 2012/409, 2012, http://eprint.iacr.org/2012/409. [Cited on page 25.]

[59] M. Mosca, “Cybersecurity in an era with quantum computers: will we be ready?” Cryptology ePrint Archive, Report 2015/1075, 2015, http://eprint.iacr.org/2015/1075. [Cited on page 5.]

[60] H. Niederreiter, “Knapsack-type cryptosystems and algebraic coding theory,” *Problems of Control and Information Theory*, no. 15, pp. 159–166, 1986. [Cited on page 25.]

[61] M. A. Nielsen and I. L. Chuang, *Quantum computation and quantum information*, 10th ed. Cambridge University Press, 2009, ISBN: 9780521635035. [Cited on page 11.]

[62] NIST, "Post-quantum crypto project," 2016, accessed 2017-03-31. [Online]. Available: http://csrc.nist.gov/groups/ST/post-quantum-crypto/ [Cited on page 5.]

[63] "The case for elliptic curve cryptography," NSA, 2009, accessed 2017-04-23. [Online]. Available: https://web.archive.org/web/20090117023500/http://www.nsa.gov/business/programs/elliptic_curve.shtml [Cited on page 35.]

[64] NSA, "Cryptography today," 2015, accessed 2017-03-31. [Online]. Available: http://web.archive.org/web/20160420221628/https://www.nsa.gov/ia/programs/suiteb_cryptography/index.shtml [Cited on page 5.]

[65] "liboqs," Open Quantum Safe, accessed 2017-04-24. [Online]. Available: https://github.com/open-quantum-safe/liboqs [Cited on pages 7 and 29.]

[66] "Openssl," OpenSSL, accessed 2017-04-24. [Online]. Available: https://github.com/openssl/openssl [Cited on page 30.]

[67] J. Patarin, *Hidden Fields Equations (HFE) and Isomorphisms of Polynomials (IP): Two New Families of Asymmetric Algorithms.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1996, pp. 33–48. [Online]. Available: http://dx.doi.org/10.1007/3-540-68339-9_4 [Cited on page 27.]

[68] C. Peikert, "Lattice cryptography for the internet," Cryptology ePrint Archive, Report 2014/070, 2014, http://eprint.iacr.org/2014/070. [Cited on page 18.]

[69] C. Peters, "isdfq," accessed 2017-04-24. [Online]. Available: https://bitbucket.org/cbcrypto/isdfq/src [Cited on page 32.]

[70] C. Peters, *Information-Set Decoding for Linear Codes over F q.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 81–94. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-12929-2_7 [Cited on page 32.]

[71] J. Porras, J. Baena, and J. Ding, *ZHFE, a New Multivariate Public Key Encryption Scheme.* Cham: Springer International Publishing, 2014, pp. 229–245. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-11659-4_14 [Cited on page 27.]

[72] J. Porras, J. B. Baena, and J. Ding, "New candidates for multivariate trapdoor functions," Cryptology ePrint Archive, Report 2014/387, 2014, http://eprint.iacr.org/2014/387. [Cited on page 27.]

[73] O. Regev, "On lattices, learning with errors, random linear codes, and cryptography," *J. ACM*, vol. 56, no. 6, pp. 34:1–34:40, Sep. 2009. [Online]. Available: http://doi.acm.org/10.1145/1568318.1568324 [Cited on page 15.]

[74] R. L. Rivest, A. Shamir, and L. Adleman, "A method for obtaining digital signatures and public-key cryptosystems," *Commun. ACM*, vol. 21, no. 2, pp. 120–126, Feb. 1978. [Online]. Available: http://doi.acm.org/10.1145/359340.359342 [Cited on page 10.]

[75] C. P. Schnorr and M. Euchner, *Lattice basis reduction: Improved practical algorithms and solving subset sum problems.* Berlin, Heidelberg: Springer Berlin Heidelberg, 1991, pp. 68–85. [Online]. Available: http://dx.doi.org/10.1007/3-540-54458-5_51 [Cited on page 31.]

[76] "Ntruencrypt," Security Innovation, accessed 2017-04-24. [Online]. Available: https://github.com/NTRUOpenSourceProject/NTRUEncrypt [Cited on page 29.]

[77] "Security innovation makes ntruencrypt patent-free," Security Innovation, 2017, accessed 2017-05-29. [Online]. Available: https://www.securityinnovation.com/company/news-and-events/press-releases/security-innovation-makes-ntruencrypt-patent-free [Cited on page 46.]

[78] A. Shellhammer, "The need for mobile speed: How mobile latency impacts publisher revenue," 2016, accessed 2017-04-03. [Online]. Available: https://www.doubleclickbygoogle.com/articles/mobile-speed-matters/ [Cited on page 33.]

[79] P. W. Shor, "Algorithms for quantum computation: discrete logarithms and factoring," in *Proceedings 35th Annual Symposium on Foundations of Computer Science*, Nov 1994, pp. 124–134, DOI: 10.1109/SFCS.1994.365700. [Cited on pages 5, 11, and 13.]

[80] V. Singh, "A practical ring-lwe key exchange implementation," accessed 2017-04-24. [Online]. Available: https://github.com/vscrypto/ringlwe [Cited on pages 19 and 30.]

[81] V. Singh, "A practical key exchange for the internet using lattice cryptography," Cryptology ePrint Archive, Report 2015/138, 2015, http://eprint.iacr.org/2015/138. [Cited on pages 19 and 32.]

[82] V. Singh and A. Chopra, "Even more practical key exchanges for the internet using lattice cryptography," Cryptology ePrint Archive, Report 2015/1120, 2015, http://eprint.iacr.org/2015/1120. [Cited on pages 19 and 32.]

[83] D. Stebila, "ds_benchmark," accessed 2017-04-24. [Online]. Available: https://gist.github.com/dstebila/6980008ec98209ef6075 [Cited on page 33.]

[84] D. Stebila and M. Mosca, "Post-quantum key exchange for the internet and the open quantum safe project," 2016, accessed 2017-03-31. [Online]. Available: https://eprint.iacr.org/2016/1017.pdf [Cited on page 7.]

[85] D. Stehlé and R. Steinfeld, *Making NTRU as Secure as Worst-Case Problems over Ideal Lattices.* Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 27–47. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-20465-4_4 [Cited on page 51.]

[86] S. Streit and F. D. Santis, "Post-quantum key exchange on armv8-a – a new hope for neon made simple," Cryptology ePrint Archive, Report 2017/388, 2017, http://eprint.iacr.org/2017/388. [Cited on page 50.]

[87] S. Tani, "Claw finding algorithms using quantum walk," *Theoretical Computer Science*, vol. 410, no. 50, pp. 5285 – 5297, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397509006136 [Cited on page 32.]

[88] C. Tao, A. Diene, S. Tang, and J. Ding, *Simple Matrix Scheme for Encryption*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 231–242. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-38616-9_16 [Cited on page 27.]

[89] E. Thomae and C. Wolf, "Solving underdetermined systems of multivariate quadratic equations revisited," in *Proceedings of the 15th International Conference on Practice and Theory in Public Key Cryptography*, ser. PKC'12. Berlin, Heidelberg: Springer-Verlag, 2012, pp. 156–171. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-30057-8_10 [Cited on page 15.]

[90] "Valgrind," Valgrind developers, accessed 2017-05-15. [Online]. Available: http://valgrind.org/ [Cited on page 33.]

[91] L. Xu, "Secure the enterprise with intel® aes-ni: White paper," 2010. [Online]. Available: https://www.intel.com/content/www/us/en/enterprise-security/enterprise-security-aes-ni-white-paper.html [Cited on page 34.]

[92] S. Zhang, "Promised and distributed quantum search," in *Proceedings of the 11th Annual International Conference on Computing and Combinatorics*, ser. COCOON'05. Berlin, Heidelberg: Springer-Verlag, 2005, pp. 430–439. [Online]. Available: http://dx.doi.org/10.1007/11533719_44 [Cited on page 32.]

Printed by Tryckeriet i E-huset, Lund 2017

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2017-583

http://www.eit.lth.se