Mathematics of Post-Quantum Cryptography

Senior Thesis in Mathematics
American University in Bulgaria

Author: Vitaliy Konyukhov
Advisor: Prof. Alexander Ganchev

May 11, 2022

Abstract

This senior thesis aims to provide an overview of the existing cryptography methods and their flaws in light of quantum computing and explore the post-quantum cryptography algorithms with the mathematics behind them. Post-quantum cryptography refers to a family of cryptographic algorithms that are secure against an attack by both a classical and a quantum computer. Such cryptography will be required when quantum computers are powerful enough to perform Grover's or Shor's algorithms and break the existing public-key cryptography methods. Additionally, this paper covers quantum cryptography, which is also quantum-safe. Quantum key distribution is an example of quantum cryptography that uses properties of quantum mechanics to securely exchange the encryption keys.

Contents

Chapter 1

Introduction

Modern cryptography is the foundation of computer security. It is based on various mathematical concepts, such as number theory, computational complexity theory, and probability theory. The main component of cryptography is encryption. Messages are encrypted and decrypted using complex algorithms that utilize a combination of computer science and mathematics.

In any encryption method, an algorithm and a key are used to convert input data into encrypted output data. This method ensures that only the sender and addressee can view the messages because the encrypted information can only be decrypted by someone with the secret key to convert the message into plain text.

Encryption is widely and frequently used to secure information, and its importance is growing with the increasing use of the Internet. Every time a new encryption algorithm is created, it gets cracked and leads to the creation of another encryption algorithm.

The rapid popularization of computers has increased the need for encryption for private applications, such as personal confidential information transfer, business transactions, as well as military applications. Cryptographic systems with a private key are primarily used to transfer secret messages. The sender encrypts the message using a key, and the recipient decrypts the received encrypted message with the same key. Unfortunately, such systems with a private key have difficulties in practical implementation. The main question is how to distribute the keys since a malicious third party can capture the key and read the messages.

There are two types of encryption: symmetric and asymmetric (otherwise known as public-key encryption) [16]. In symmetric encryption, a key is created, the message is encrypted using that key, and the result is sent to the recipient while passing the key or a password separately. The addressee can read the message after running the reverse of the same encryption operation with the received key. Symmetric encryption is less secure than asymmetric encryption because the key can be intercepted.

Asymmetric encryption is more complicated and more secure. It requires a pair of corresponding keys: a public key and a private key. The public key is available to everyone. It only allows encrypting the data but cannot be used to decrypt it. The private key must be kept secret. When someone needs to send an encrypted message, they perform the encryption using a certain public key. Once the message is received, it can be decrypted using the corresponding private key.

A message encrypted with a certain public key can only be decrypted with the

corresponding private key, which means that attackers would try to find the private key. It cannot be intercepted because it is not transmitted anywhere, but theoretically, it can be obtained from the public key. However, cryptographic algorithms are specially designed so that the task of obtaining a private key from a public key cannot be solved in a reasonable amount of time.

The invention and development of quantum computers pose a threat to public-key cryptography. Due to the nature of quantum computers, they can solve the problem of obtaining a private key much faster than classical computers. Therefore, the unreasonable time required to break the cipher can turn into a relatively short time when using a quantum computer and render the existing cryptographic methods useless.

This paper will explore how the most widely used encryption algorithms work, their flaws and weaknesses that can be exploited by the quantum computer, quantum key distribution algorithms that utilize the principles of quantum mechanics to protect the encryption keys from interception, and study the post-quantum encryption methods that are secure against both a classical and a quantum computer attacks.

Chapter 2

Modern Cryptography

There are two main sciences regarding data encryption: cryptography and cryptanalysis. The goal of cryptography is to research the methods of transforming information. Cryptanalysis aims at evaluating the possibility of decrypting information without knowing the encryption keys.

Modern cryptography relies on mathematical theory and computer science algorithms to employ computational difficulty assumptions (when a problem cannot be solved in polynomial time) for designing cryptographic algorithms. Such algorithms are infeasible to break using the current computational devices, for example, involving the discrete logarithm or the integer factorization problems [24].

2.1 Time Complexity

A concept of polynomial time refers to the time complexity that describes the amount of computer time it takes to run an algorithm. In computer science, the time complexity is usually expressed using the big O notation. Linear time complexity is expressed as $O(n)$, where n is the size in bits that represents the input. The abovementioned polynomial time could be expressed as $O(n^c)$ or $poly(n)$, and its running time complexity is the polynomial expression with the degree of the complexity of the input. Computers perform mathematical operations of addition, subtraction, multiplication, division, square roots, powers, and logarithms in polynomial time.

The paradigm of time complexity is vastly important in the field of cryptography since it determines how safe an encryption scheme is. Any such scheme is designed to encrypt and decrypt a message in a short amount of time when the key is given and to make the decryption without a key infeasible. An encryption scheme would be useless if decryption without a key can be performed in a small amount of computational time. A strong encryption scheme requires years for a malicious third party to decrypt a single message.

The time complexity of an algorithm is determined from its design: sequential statements have constant time complexity $O(1)$, loop statements have linear time complexity $O(n)$, conditional statements have linearithmic time complexity $O(n \cdot log(n))$, recursive statements have exponential time complexity $2^{poly(n)}$, trial and error (brute-force) statements have factorial time complexity $O(n!)$. The time complexity can be improved if the algorithm uses multithreading and other optimization techniques.

2.2 One-way Functions

One-way functions are fundamental tools in cryptography, personal identification, authentication, and other areas of data protection. Although the existence of such functions remains unproven, there are one-way functions that have not been rejected yet. They are an integral part of most telecommunications systems, as well as e-commerce and online banking systems around the world since modern cryptography is heavily dependent on the use of such functions.

The main characteristic of one-way functions is that it is easy to compute the result of such functions but very difficult to compute the inverse of that result [3]. Even though we know the function, say $f(x)$, and we can easily calculate the result for any input, say $f(a) = b$ but finding the inverse $f^{-1}(b)$ of that function would take an unreasonable amount of computing time from the point of view of computational complexity theory. There are two types of one-way functions: those that produce a fixed-length output and those with a variable-length output.

Definition 2.2.1 (One-way Function)**.** A function $f : \{0,1\}^* \longrightarrow \{0,1\}^*$ is a one-way function if [3]:

1. f can be evaluated in polynomial time.
2. f^{-1} cannot be found in polynomial time or, in other words, there is only a negligible probability that any efficient algorithm will solve the problem of inverting f in polynomial time.

One type of one-way function is a trapdoor function. If f is a trapdoor function, then there exists a secret y, such that given $f(x)$ and y, x can be computed easily [3]. RSA and Rabin cryptosystems use asymmetric encryption implemented as trapdoor functions, and their security is related to the complexity of factorization. They are presented as the exponentiation modulo a composite number and assumed that it is difficult to factorize a large composite number. Functions involving a discrete logarithm problem, such as modulo a prime, are not trapdoor functions since there is no secret that would allow their efficient computation.

Another type of one-way function is a cryptographic hash function. It transforms the input of any size into a hash value of a fixed size. Its main property is that such a function should be collision-free, meaning two different input sequences should not produce the same hash value [3]. A slight change in the input normally causes a drastic change in the hash value. This way, data can be protected from any modifications since its integrity is verified by comparing the hash values. An inverse of a hash function can be obtained using a brute-force search or a rainbow table of matched hashes. Examples of one-way hash functions include MD5 (currently unsuitable for most use cases due to collisions), SHA-1 (considered vulnerable due to collisions), SHA-2, SHA-3, where SHA stands for Secure Hash Algorithm, and the security of each new generation is more robust than the predecessor.

2.3 The RSA Scheme

RSA (Rivest–Shamir–Adleman) is an asymmetric (public-key) encryption technique that is widely used for data transmission in computer systems [6]. Its cryptographic security is based on the complexity of factoring large numbers or, in other

words, on the exceptional difficulty of finding the secret key using the public key. It uses RSA numbers that are a set of large semiprimes (numbers with exactly two prime factors). The most secure systems use 2048-bit numbers that can store 2^{2048} or $3.23 \cdot 10^{616}$. That is a number with 617 decimal digits, and the corresponding RSA-2048 method is considered as not factorizable in the near future.

Algorithm 2.3.1 (RSA Key Generation)**.** To generate the public and the private keys, we need to do the following [6] [14]:

1. Choose two distinct prime integers p and q. These numbers are chosen at random and are kept secret.

2. Define n as a result of multiplication of these numbers: $n = p \cdot q$. It will be a part of the public key (the modulus), and its length represented in bits determines the key length for the RSA encryption.

3. Choose a random integer number e. This number must be coprime (have no common divisors except 1) with the result of multiplication $(p-1) \cdot (q-1)$. This number is also a part of the public key (the exponent).

4. Determine a number d for which the following relation is true:

   $$d \cdot e = 1 \bmod (p-1) \cdot (q-1)$$

   This number is the private key exponent.

5. The public key consists of the modulus n and the public exponent e. The private key is comprised of the modulus n and the private exponent d.

Algorithm 2.3.2 (RSA Encryption and Decryption)**.** In order to encrypt a message using the public key n,e, we need to perform the following operation [14] [24]:

1. Turn the message into a nonnegative integer m, such that $m < n$. If $m \geq n$, the message can be broken up into blocks, and each block is turned into a nonnegative integer.

2. Compute the ciphertext c as follows:

   $$c = m^e \bmod n$$

In order to decrypt the message using the private key n,d, we have to do the following [14] [24]:

1. Recover the original integer from the following formula:

   $$m = c^d \bmod n$$

2. Turn the recovered integer back into the text to obtain the original message. If the message consists of several blocks, reconstruct the message.

Proof. The correctness of the RSA ($m^{e \cdot d} = m \bmod n, n = p \cdot q$) can be proven using Euler's theorem. If e and d are positive integers such that $e \cdot d = 1 \bmod \varphi(n)$, then $e \cdot d = 1 + h \cdot \varphi(n)$ for some nonnegative integer h. Since m is coprime to n,

$$m^{ed} = m^{1+h \cdot \varphi(n)} = m(m^{\varphi(n)})^h = m(1)^h = m \bmod n,$$

due to Euler's theorem. □

Knowing the algorithms of RSA key generation, encryption, and decryption, we can analyze the potential flaws of this encryption technique. The security of an RSA implementation depends on the following parameters [24]:

1. Prime selection.
   The security of RSA is based on the difficulty of factorizing a large number. This number is a product of two primes. These primes have to be selected in a way that their product cannot be factorized. To ensure the security of the algorithm, the primes have to be truly random and independent. If they share enough of their upper bits, their product can be factorized using Fermat's factorization method. If there are multiple certificates generated, there is a possibility of duplicate primes, which allows factoring the modulus using the Euclidean algorithm.

2. Public exponent.
   In some cases, in order to speed up the encryption time, the public exponent is chosen to be a small number which leads to lower security of the RSA encryption. When e is small, the Franklin-Reiter attack can be used to decrypt two RSA-encrypted messages that differ by a known fixed difference.

3. Private exponent.
   To improve the decryption performance, the value of d is sometimes chosen to be small. In a case when $d < \frac{1}{3}n^{\frac{1}{4}}$, the private key can be recovered using Wiener's attack, and the RSA encryption will be compromised.

The abovementioned methods and attacks for decrypting a message without a private key or recovering the private key are efficient and can be performed in some cases. However, they are only applicable under certain circumstances when the RSA parameters are not chosen to be secure. For the RSA-2048, which is widely used today, the RSA numbers have been considered unfactorizable for many years.

2.4 Diffie–Hellman Key Exchange

Diffie–Hellman key exchange is a method that allows exchanging cryptographic keys over a public channel securely [15]. It was the first method that reduced the communication channel requirements for establishing a secure connection without prior exchange of keys. The protocol allows two parties to create a common session key using a communication channel that can be compromised by an intruder, but with the assumption that the latter cannot change the content of the messages.

Algorithm 2.4.1 (Diffie–Hellman Key Exchange)**.** Let p be a large prime number, and g a primitive root modulo p. p and g are publicly shared between Alice and Bob. A one-way function $y = g^x \bmod p$ is then used for the key exchange. The exchange protocol consists of the following [14] [15]:

1. Alice picks a random a such that $2 \leq a \leq p-1$ and computes $A = g^a \bmod p$. The result of the computation is then sent to Bob.
2. Bob picks a random b such that $2 \leq b \leq p-1$ and computes $B = g^b \bmod p$. The result of the computation is then sent to Alice.
3. Alice computes the session key $K = B^a \bmod p$
4. Bob computes the session key $K = A^b \bmod p$

The resulting session key K can be used as the symmetric encryption key, so the information is encrypted and decrypted using a symmetric-key cipher with the same key obtained by both parties.

Proof. To prove the equivalence of the shared secret, we need to show that

$$(g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$$

By the definition of mod, we can expand $g^a \bmod p$ as $g^a + pk$, $k \in \mathbb{Z}$. Then

$$(g^a + pk)^b = g^{ab} + \binom{b}{1} A^{a(b-1)} pk + \cdots + (pk)^b$$

Since all terms except the first one contain p,

$$(g^a \bmod p)^b = g^{ab} \bmod p$$

Analogously,

$$(g^b \bmod p)^a = g^{ab} \bmod p$$

Therefore,

$$(g^a \bmod p)^b = (g^b \bmod p)^a$$
$$(g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$$

□

The protocol provides only the generation of new session keys. In the absence of a trusted third party, it cannot provide authentication of the parties exchanging the keys [15]. Therefore, the Diffie-Hellman key exchange can be compromised by a man-in-the-middle attack if the communication channel is not secure. A third party standing between Alice and Bob can exchange the session keys with Alice and Bob separately and then forward the messages between them using different session keys. This way, the intruder would be able to read messages, and it would be impossible to detect. Normally, the encryption schemes involving Diffie-Hellman key exchange incorporate additional one-way or two-way authentication methods.

The security of Diffie-Hellman key exchange is based on the difficulty of solving the discrete logarithm problem [15], that is, calculating the inverse function

$$x = log_g y \bmod p.$$

Currently, computers cannot efficiently compute such problems. An efficient algorithm to solve them would make it possible to compute a or b and solve the Diffie–Hellman problem.

Chapter 3

Quantum Computers and Algorithms for Cryptanalysis

Quantum mechanics is a branch of physics that studies the mathematical description of the motion and interaction of subatomic particles, concepts of quantization of energy, wave-particle duality, the uncertainty principle, and the correspondence principle [22]. Quantum mechanics is different from classical physics because energy, momentum, and other quantities of the bound state of a system cannot take arbitrary values but are limited to discrete values. The characteristics of objects can be interpreted as both particles and waves (wave-particle duality), and we cannot accurately predict the value of a physical quantity before measuring it when given a full set of initial conditions (uncertainty principle).

Quantum mechanics describes physical phenomena whose actions are comparable in magnitude to the Planck's constant. Planck's constant is a fundamental physical constant denoted h. A photon's energy is calculated with the following formula:

$$E = hf$$

$$h = 6.62607015 \cdot 10^{-34} J \cdot Hz^{-1}$$

where f is the frequency, and h is the Planck's constant [22]. In quantum physics, a quantum state of a system is described by a wave function Ψ, which is a complex-valued probability amplitude function:

$$\Psi = e^{k\phi},\ k = \frac{i}{\hbar},\ \hbar = \frac{h}{2\pi}$$

where ϕ is the phase of the wave function.

A system with n states forms an orthonormal basis of a Hilbert space of dimension n, and the quantum state is the superposition of classical states [22]. We can perform either a unitary transformation of the quantum state or measure it since the process of measurement will lead to a change in the quantum state. While unitary transformations are reversible, the measurement is irreversible.

A qubit (quantum bit) is a unit of quantum information, which is a two-state system [22]. The system could be comprised of the spin of an electron (spin up and spin down) or the polarization of a photon (vertical polarization and horizontal polarization). A qubit can be in a superposition of both basis states, and when it is measured, it will collapse to one of its eigenstates which will be the outcome of this measurement [22].

A qubit's state of superposition can be represented using the Dirac notation [22]:

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

where α and β are the amplitudes (complex numbers) [13]. The two states of qubits form an orthonormal basis, and are given by

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix}0\\1\end{bmatrix}.$$

A quantum computer is a computing device that uses quantum mechanics (quantum superposition, entanglement, parallelism) to transmit and process data through the means of qubits. Quantum computation is controlled by a classical computer that inputs the sequence of unitary operations and then measures the state of the quantum processor, which produces the result of the calculation. Two basic operations are enough to perform any calculation. The quantum system produces a result that is correct only with some probability, but the probability of getting a correct result can be increased by slightly increasing the number of operations in the algorithm.

There are several algorithms that a quantum computer can perform [22]:

1. Shor's algorithm, which allows factoring a natural number in polynomial time.
2. Grover's algorithm, which allows finding the solution of an equation $f(x) = 1, 0 \leq x < N$ in time $O(\sqrt{N})$.
3. Deutsch–Jozsa algorithm, which determines whether a binary function is constant or balanced.
4. HHL algorithm, which solves linear systems.

The main feature of these algorithms is that they are faster than any possible classical algorithm for the respective task.

3.1 Basics of Quantum Mechanics

Quantum mechanics has the following fundamental principles [22]:

1. The uncertainty principle.
   According to Heisenberg's uncertainty principle, it is impossible to exactly determine the position and momentum of a particle at the same time [22]. In other words, the more accurate the predicted position of a particle is, the more uncertain our prediction about the momentum becomes, and vice versa.

   Let σ_x be the standard deviation of position, and σ_p be the standard deviation of momentum, then

   $$\sigma_x \cdot \sigma_p \geq \frac{\hbar}{2}$$

   where $\hbar$ is the reduced Planck's constant.

2. The complementarity principle.
   The complementarity principle states that there are complementary properties that cannot be observed or measured simultaneously [22]. Measuring the position of a particle makes it impossible to measure its momentum and vice versa. From the canonical commutation relation

$$[\hat{x}, \hat{p}] = i\hbar$$

   we can see that the position and the momentum are non-commuting operators and are called incompatible observables.

3. The superposition principle.
   The quantum superposition principle states that multiple quantum states can be added together and the result will be another quantum state, and every quantum state can be represented as a sum of several other quantum states [22]. If Ψ is a statefunction, φ_n are the normalized eigenstates, and b_n are the coefficients, then

$$|\,\Psi\rangle = \sum_{n=1}^{\infty} |\,b_n \varphi_n\rangle$$

4. Quantum entanglement.
   Quantum entanglement is a quantum mechanical property when the quantum states of multiple objects are correlated [22]. For example, if one of the two entangled photons is measured to have vertical polarization, the polarization of the other photon will be horizontal. Mathematically, quantum entanglement represents a tensor product of Hilbert spaces.

5. Quantum parallelism.
   Quantum parallelism allows performing a large number of operations in parallel since qubits can perform calculations using all possible input values simultaneously.

6. No-cloning theorem.
   The no-cloning theorem states that it is not possible to make an identical copy of an unknown quantum state without affecting that state [22].

3.2 Quantum Fourier Transform

The quantum Fourier transform is a quantum version of the discrete Fourier transform and is a linear transformation over the amplitudes of qubits. It is used in many quantum algorithms and allows for solving the hidden subgroup problem. Quantum computers can efficiently perform the quantum Fourier transform (in polynomial time).

A Fourier transform decomposes functions depending on time or space into functions depending on temporal or spatial frequency. A discrete Fourier transform is a Fourier transform for cases when the ordered pairs are equally spaced in their input variable.

The discrete Fourier transform maps a vector $(x_0, x_1, \ldots, x_{N-1}) \in \mathbb{C}^N$ to another vector $(y_0, y_1, \ldots, y_{N-1}) \in \mathbb{C}^N$ as follows:

$$y_k = \frac{1}{\sqrt{N}} \sum_{n=0}^{N-1} x_n \cdot e^{-\frac{2\pi i}{N} kn}.$$

The quantum Fourier transform on a basis $|\,0\rangle, |\,1\rangle, \ldots, |\,N-1\rangle$ is the following linear operation on the states [12]:

$$|\,j\rangle \rightarrow \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} e^{\frac{2\pi ijk}{N}} |\,k\rangle.$$

The quantum Fourier transform can be also expressed as a unitary matrix:

$$U_{QFT} = \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} \sum_{k=0}^{N-1} e^{\frac{2\pi ijk}{N}} |\,k\rangle\langle j\,|.$$

3.3 Shor's Algorithm

Peter Shor's algorithm is a quantum algorithm for integer factorization. The goal of the algorithm is to reduce the factorization problem to finding the period of a function. It allows factorizing the number N in polynomial time ($O(log^3 N)$) using $O(logN)$ qubits. This means that a quantum computer with about 3,000 qubits can break the RSA cryptographic system with a 2048-bit key in time comparable to encrypting the message. The efficiency of Shor's algorithm was first demonstrated in 2002 by factorizing the number 15 on a quantum computer that had seven qubits [25]. If a sufficiently powerful quantum computer is created, Shor's algorithm will break the existing asymmetric cryptographic schemes, such as RSA, and key exchange methods, such as Diffie-Hellman.

The goal of Shor's algorithm is to find an integer d (from 1 to N) that divides N (an odd composite number). The algorithm consists of two parts [23]:

1. Convert the factorization problem into the order-finding problem.

2. Find the quantum period using the quantum Fourier transform. This part uses quantum parallelism to speed up the computation of the factorization problem.

Algorithm 3.3.1 (Shor's Algorithm)**.** Shor's algorithm is performed as follows [12] [23]:

1. Pick a random integer r, such that $1 < r < N$, and r and N are coprime numbers.

2. Use a quantum computer to obtain p, the period of the function

$$f_{r,N}(x) = r^x \bmod N.$$

This step is a form of the hidden subgroup problem.

3. If p is an odd integer, return to step 1, otherwise we have

$$r^p - 1 = (r^{p/2} - 1)(r^{p/2} + 1) = 0 \bmod N$$

since p is an even integer.

4. If $(r^{p/2} + 1) = 0 \bmod N$, return to step 1, otherwise calculate

$$d = gcd(r^{p/2} - 1, N).$$

5. The resulting d is a nontrivial factor of N.

Suppose we have a number N, then a brute force algorithm would check all primes up to $\sqrt{N}$ to find factors of that number, which makes its time complexity exponential. The most efficient classical algorithm for factoring, the general number field sieve, still takes exponential time, despite being faster. On the other hand, Shor's algorithm factorizes a number in polynomial time using the quantum computer. By creating a superposition of states, implementing a function as a quantum transform, and performing a quantum Fourier transform, Shor's algorithm achieves such time complexity [23].

Quantum computers use the ability of qubits (information units) to take several values simultaneously and the state of entanglement to find the period of the function used for Shor's algorithm. This process is probabilistic since reading the resulting values from the qubits involves randomness.

The paper by Amico et al. has demonstrated how a compiled version of Shor's algorithm can be implemented on the IBM quantum computer to factor the numbers 15, 21, and 35 using 5, 6 and 7 qubits, respectively [2]. The paper shows that the numbers 15 and 21 can be successfully factored on the ibmqx5 quantum processor, but factoring the number 35 is successful only 14% of the time due to errors [2].

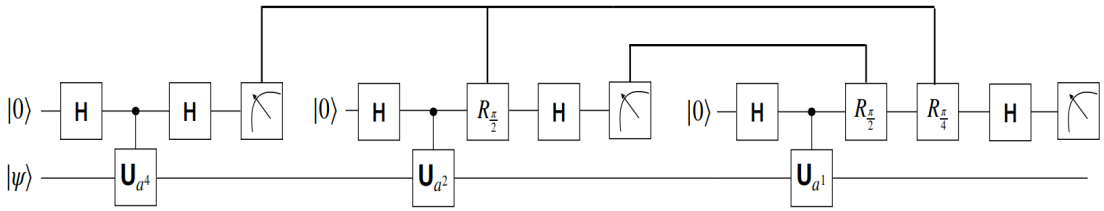

Figure 3.1: Circuit of Shor's algorithm for factoring numbers 15, 21, and 35 [2].

3.4 Hidden Subgroup Problem

Shor's algorithm involves solving the hidden subgroup problem (HSP) for finite abelian groups to extract the period in a superposition [4]. The HSP is defined as follows, "given a group G, an unknown subgroup H, and oracle access to a function f that is constant and distinct on cosets of H, find H"[10]. Quantum computers can solve such problems in polynomial time for finite abelian groups using a quantum Fourier transform. The hidden subgroup problem covers the problem of integer factorization and discrete logarithm problem.

In Shor's algorithm for factoring a number n, the abovementioned group G is the multiplicative group of integers mod n, $\mathbb{Z}_n^*$, which is abelian, the function is $f(x) = c^x \bmod n$, and H is the subgroup with the index of the multiplicative order of c. In this case, we extract the period in a superposition using Fourier transform.

A function $f : G \to S$, where S is any set, hides the subgroup H ($H \leq G$) if $\forall g_1, g_2 \in G$,

$$f(g_1) = f(g_2) \iff g_1 H = g_2 H.$$

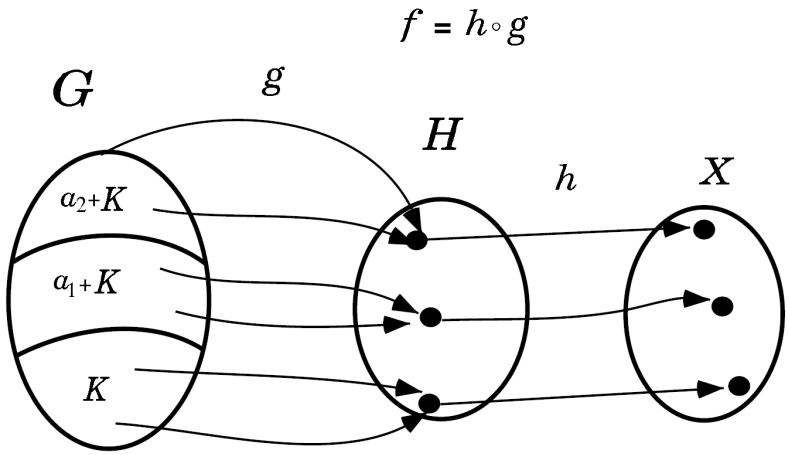

Figure 3.2: A function $f : G \to X$ could be represented as $f = h \circ g$, g is a homomorhism $G \to H$, h is one-to-one mapping $H \to X$, where X is a finite set, K is the kernel of a finitely generated group G, and $H \cong G/K$ [18]. In this case, the hidden subgroup problem is to find K.

Definition 3.4.1 (Character of an Abelian Group)**.** If G is an abelian group, a function $f : G \to \mathbb{C} \setminus \{0\}$ that is a group homomorhism from G to $\mathbb{C}^\times$ (the multiplicative group of non-zero complex numbers) is called a character of the group G.

Algorithm 3.4.1 (Abelian Hidden Subgroup Problem)**.** According to the paper by Lawrence Ip, if G is abelian, the hidden subgroup problem can be solved efficiently using a quantum computer as follows [10]:

1. Use the qubits in superposition

   $$\sum_{g \in G} |\, g, f(g)\rangle,$$

   to measure $f(g)$ to obtain a superposition over cH, which is a random coset of H

   $$\sum_{h \in H} |\, ch, f(c)\rangle,$$

   where c is a randomly selected element of G.

2. Perform the Fourier transform of ch and the resulting

   $$\sum_{\chi \in \hat{G}} \sum_{h \in H} \chi(ch) |\, \chi\rangle,$$

   has the dual group $\hat{G}$ of G. This is the group of characters of G.

3. Measure the first register and obtain the character χ and repeat the first three steps n times. This way, we obtain a random element of the dual space each time.

4. The resulting subgroup of G whose $H^{\perp}$ (the set of characters χ whose kernel contains H, or $\chi(h) = 1, \forall\, h \in H$) is equal to the subgroup of $\hat{G}$ generated by the characters $\chi_1, \ldots, \chi_n$.

Reducing a public key cryptosystem to the hidden subgroup problem will allow breaking the key of such cryptosystem using the quantum computer.

3.5 Grover's Algorithm

Lov Grover's algorithm or quantum search algorithm is a quantum algorithm for unstructured search that finds the input of an unknown function that produces a particular output value with complexity $O(\sqrt{N})$, where N is the domain of such function [24]. A black box function called an oracle is used as a database for the algorithm. If the oracle function evaluates to 1 (meaning the input is the correct answer), its eigenvalue will be -1, but if the input is incorrect, the eigenvalue will be 1. The algorithm allows finding the solution of a function of type $f(x) = y$ where f is a boolean function with n variables (the algorithm finds x for the given y). It uses a technique called amplitude amplification, which amplifies the amplitude of the target state by decreasing the amplitude of all other states [24]. The algorithm is probabilistic due to the nature of quantum algorithms and gives the correct answer with high probability.

Algorithm 3.5.1 (Grover's Algorithm)**.** Grover's algorithm is performed by the following steps [11] [8]:

1. Put the qubits into superposition

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

   Now the qubits have the uniform amplitude.

2. Use the oracle function to flip the amplitude of the desired input

$$U_f |\, x\rangle = (-1)^{f(x)} |\, x\rangle.$$

3. Apply a state change (Grover diffusion operator) to the qubits to reflect the desired input

$$U_s = 2|\, s\rangle\langle s\,| - 1$$

   and obtain the probability for the amplitude of the desired input at $t + 1$

$$|\, \psi_{t+1}\rangle = U_s U_f |\, \psi_t\rangle$$

   which amplifies the desired input.

4. Repeat steps 2 and 3 until the amplitude reaches some defined threshold.

5. The produced amplitude at time t will contain the desired input

$$|\psi_t\rangle = (U_s U_f)^t |\psi_0\rangle.$$

The figures below were created in IBM Quantum Composer. Grover's algorithm was constructed for two qubits with the expected outcome $|\omega\rangle = |00\rangle$ and run on a quantum computer.

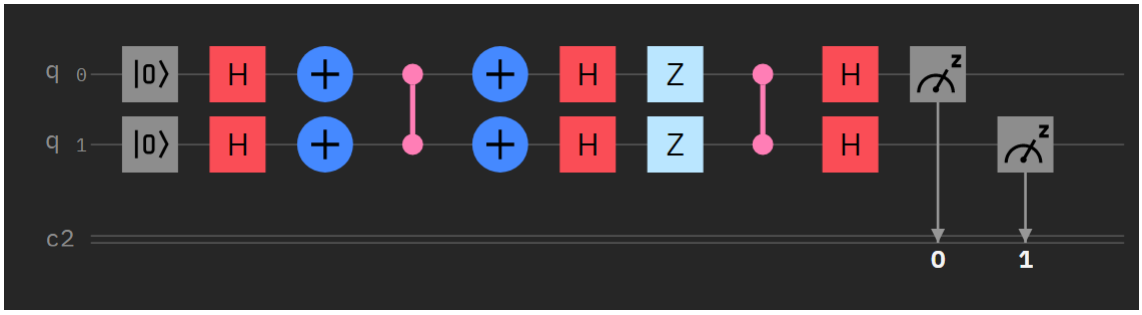

Figure 3.3: Quantum circuit for Grover's algorithm with oracle $|\omega\rangle = |00\rangle$.

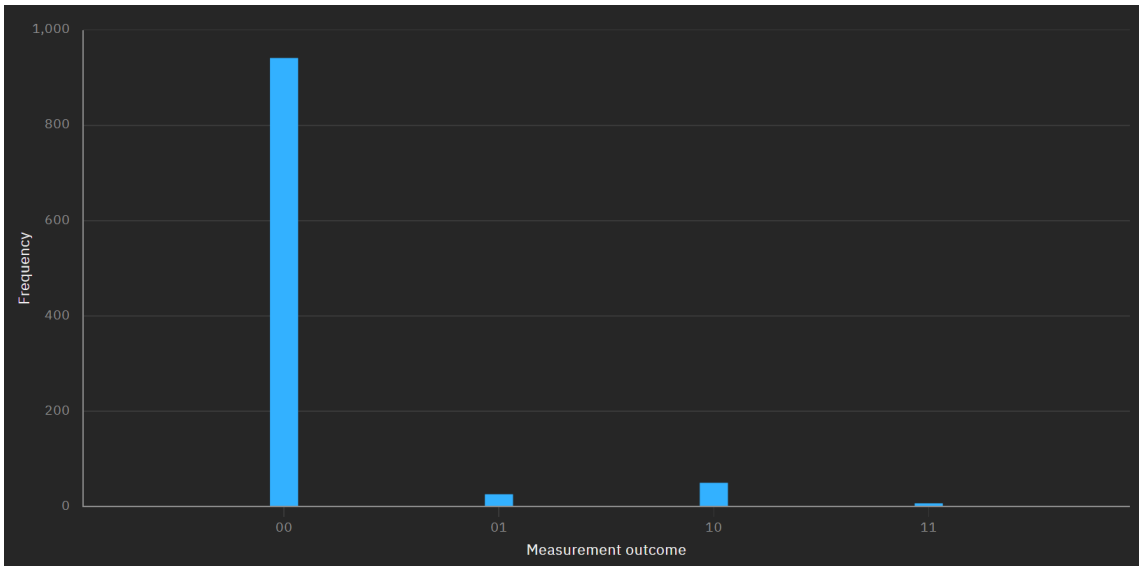

Figure 3.4: The outcome of Grover's algorithm with oracle $|\omega\rangle = |00\rangle$ after 1024 shots.

As we can see, Grover's algorithm allows finding the inverse value of a function in $O(\sqrt{N})$ time instead of $O(N)$ time required for a classical algorithm. This way, block ciphers, such as AES, become insecure. Consequently, Grover's algorithm speeds up attacks against symmetric ciphers.

Chapter 4

Post-Quantum Cryptography

The existing encryption methods will inevitably become insecure. When a sufficiently powerful quantum computer is created, algorithms such as Shor's and Grover's will crack the classical cryptosystems involving public-key encryption and key exchange algorithms. Therefore, there is a need to design cryptographic and key exchange algorithms that will be secure against cryptanalytic attacks by quantum computers. Such algorithms are referred to as post-quantum cryptography.

In many ways, the task of key distribution is as difficult as the task of private communication. A malicious third party can capture the key and easily obtain the original message. One way to avoid this is quantum encryption, in which the security of the key is guaranteed by the laws of quantum mechanics. The basic idea of a quantum key distribution algorithm is to use the feature of quantum mechanics that the act of observing a system changes the observed system. Thus, an interceptor who tries to eavesdrop will corrupt the message. In that case, the message can be restored after discarding the bad bits, and if there are too many of them, the message would be resent.

Post-quantum cryptography has three main objectives [5] [24]:

1. Replace digital signature schemes such as RSA, DSA, ECDSA.
2. Replace the RSA-based encryption protocol for key exchange.
3. Replace the key negotiation protocol based on the Diffie-Hellman algorithm.

Post-quantum cryptography research is currently exploring the following [4] [24]:

1. Lattice-based cryptography - currently the most promising candidate for post-quantum cryptography involving lattices.
2. Multivariate cryptography - uses multivariate polynomials over a finite field.
3. Hash-based cryptography - based on the security of hash functions.
4. Code-based cryptography - cryptosystems that use error-correcting codes.
5. Supersingular elliptic curve isogeny cryptography - utilizes the properties of supersingular elliptic curves and walks in a supersingular isogeny graph.

4.1 Quantum Key Distribution

Quantum key distribution (QKD) is a secure communication method that employs properties of quantum mechanics to obtain a secret shared random key used for encrypting and decrypting messages [21]. Its main property is that the communicating parties can detect eavesdropping during the process of obtaining the key since the process of measuring a quantum state changes the state. Unlike the traditional public-key cryptography, the security of QKD is proven with information theory and forward secrecy. However, QKD is only used to obtain and share keys, not for message transmission. The messages are transmitted over a standard communication channel. There are two main categories of QKD implementation: prepare-and-measure protocols (which use Heisenberg's uncertainty principle) and entanglement-based protocols (which use quantum entanglement) [16].

4.1.1 BB84 Scheme

BB84 quantum key distribution scheme was proposed by Charles Bennett and Gilles Brassard in 1984 [22]. BB84 is a prepare-and-measure implementation of QKD. It is provably secure due to the no-cloning theorem because it uses non-orthogonal quantum signal states. It allows secure transmission of a private key for use in one-time pad encryption (when each message is encrypted with a private key with a length at least as long as the message and the key is never reused) [22]. The protocol uses a quantum channel to transmit keys and a classical channel to send encrypted messages.

Algorithm 4.1.1 (BB84 Scheme)**.** If Alice wants to send a private key to Bob using the BB84 scheme, they perform the following [19] [22]:

1. Alice encodes two n bit long strings of bits, a and b, as a tensor product of n qubits:
$$|\,\psi\rangle = \bigotimes_{i=1}^{n} |\,\psi_{a_i b_i}\rangle$$
Therefore, $a_i b_i$ produces four qubit states, and b_i determines whether a_i is encoded in the computational basis or the Hadamard basis. The resulting qubits are not mutually orthogonal, making them impossible to distinguish without knowing b.

2. Alice sends the qubits $|\,\psi\rangle$ to Bob using a quantum channel ε.

3. Bob receives a state $\varepsilon(|\,\psi\rangle\langle\psi\,|)$, which may include noise in the channel and eavesdropping.

4. Bob generates a string of random bits b' with the same length as b and measures the received qubits using the basis b', obtaining a'.

5. Bob acknowledges the received transmission, then Alice and Bob communicate (Alice shares b) to determine which b_i and b'_i are not the same, then the corresponding bits in a and a' are discarded.

6. Now they have remaining k bits, and Alice randomly chooses $\frac{k}{2}$ bits and shares them with Bob. If a certain number of them match, both parties obtain a shared secret key; otherwise, the procedure starts over.

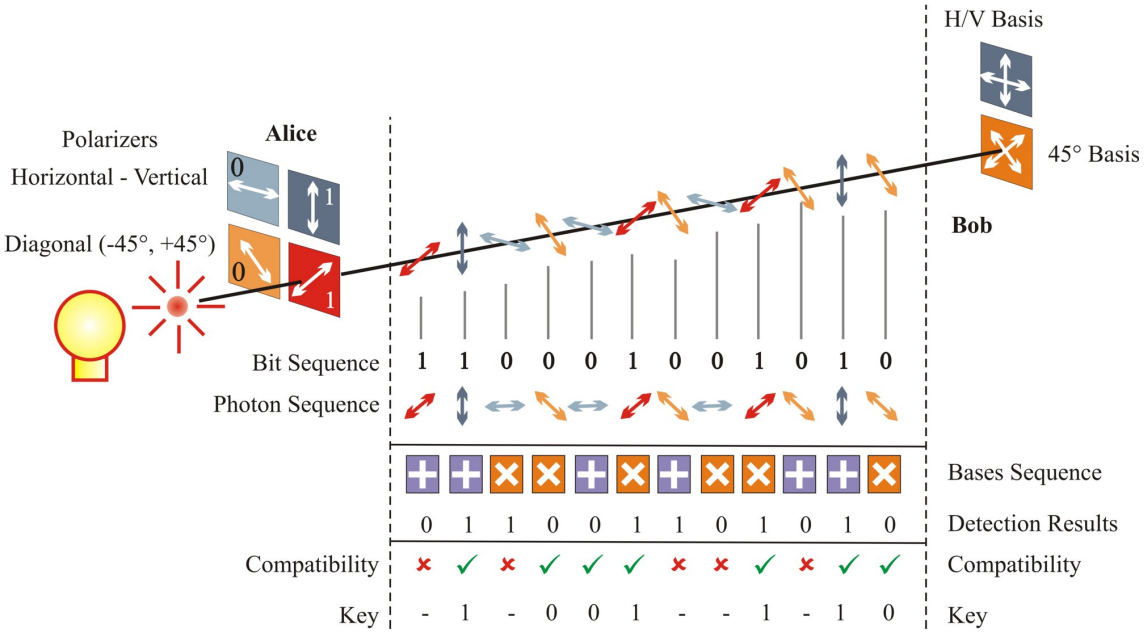

Figure 4.1: The key exchange in the BB84 protocol implemented with the polarization of photons [16].

While BB84 allows securely transmitting secret keys, it is vulnerable to man-in-the-middle attacks and requires an authenticated classical channel. Additionally, if the transmission of keys is constantly eavesdropped on (denial of service attack), it would be impossible to exchange keys since the qubits would permanently have altered quantum states. BB84 is also vulnerable to a photon-number-splitting attack since usually quantum states are sent using laser pulses that cannot precisely send one photon, thus allowing an eavesdropper to store extra photons in a quantum memory. These stored photons can be measured using the correct basis shared by the sender.

4.1.2 SARG04 Protocol

SARG04 (Scarani-Acin-Ribordy-Gisin 2004) is a quantum cryptography protocol derived from the BB84 scheme, and its main feature is protection against photon-number-splitting attacks [16]. It was initially defined as a prepare-and-measure implementation of QKD. Later an entanglement-based implementation was described. It differs from BB84 because it uses bases instead of quantum states to code the bits.

Algorithm 4.1.2 (SARG04 Protocol)**.** SARG04 is very similar to BB84 and is performed with the following steps [7] [19]:

1. Alice encodes two n bit long strings of bits, a and b, as a tensor product of n qubits:
$$|\psi\rangle = \bigotimes_{i=1}^{n} |\psi_{a_i b_i}\rangle$$
Therefore, $a_i b_i$ produces four qubit states, and b_i determines whether a_i is encoded in the computational basis or the Hadamard basis. The resulting qubits are not mutually orthogonal, making them impossible to distinguish without knowing b.

2. Alice sends the qubits $|\psi\rangle$ to Bob using a quantum channel ε.

3. Bob receives a state $\varepsilon(|\psi\rangle\langle\psi|)$, which may include noise in the channel and eavesdropping.

4. Bob generates a string of random bits b' with the same length as b and measures the received qubits using the basis b', obtaining a'.

5. Bob acknowledges the received transmission, but Alice never shares the basis b for each bit. Instead, she announces the choice of the set from which the state was selected.

6. Bob checks whether his measurement is consistent with the possible state he was shared with. If the measurement outcome is orthogonal to the state in the shared set, it is conclusive; otherwise, it is inconclusive, and Bob announces either result. Invalid bits are discarded.

7. Now they have remaining k bits, and Alice randomly chooses $\frac{k}{2}$ bits and shares them with Bob. If a certain number of them match, both parties obtain a shared secret key; otherwise, the procedure starts over.

While SARG04 is secure against a photon-number-splitting attack, it is still vulnerable to a denial of service attack using constant measuring, which modifies the quantum states and prevents the transmission of secret keys. Another example is the large pulse attack that can be performed in the SARG04 protocol. If a malicious third party flashes a bright light into the quantum channel, this light would be reflected inside the transmitting device and picked up by the internal modulator. It would get modulated, and the resulting pulse can be measured to obtain the settings of the modulator.

4.2 Quantum-Safe Cryptography

A quantum-safe (post-quantum) algorithm is secure against attacks by both a classical computer and a quantum computer. The abovementioned quantum key distribution methods BB84 and SARG04 produce a guaranteed quantum-safe shared secret if an ideal quantum channel is used [21]. However, analog physical devices used to perform QKD are imperfect. A photon-number-splitting attack is one example that exploits such imperfections. Additionally, quantum computers and quantum

channels are costly and challenging to implement. Therefore, there is a need for specialized algorithms that would be secure against various quantum attacks.

One of the candidates for post-quantum cryptography is lattice-based constructions of cryptographic primitives. Such constructions involve lattices, and some of them seem to be resistant to attacks by both classical and quantum computers since some computational lattice problems have no efficient solution.

"A lattice in $\mathbb{R}^n$ is a set $L = \left\{ \sum_{i=1}^{m} x_i b_i | x_i \in \mathbb{Z} \right\}$, where $b_1, \ldots, b_m$ are linearly independent over $\mathbb{R}$, and the matrix $B = [b_1, \ldots, b_m]$ is called a basis of the lattice L" [5]. The basis for a lattice is not unique.

4.2.1 NTRUEncrypt

The NTRU encryption algorithm is a lattice-based alternative to RSA. It is based on the short vector problem and is thought to be secure against quantum computer attacks (quantum-resistant). It operates in a truncated polynomial ring $\mathbb{Z}(X)/(X^N - 1)$. Unlike BB84 and SARG04, NTRU is not a quantum encryption algorithm but a classical one.

Algorithm 4.2.1 (NTRUEncrypt Key Generation)**.** [9] To generate a pair of public and private keys, we need two polynomials, f and g, with a degree at most $N - 1$ and with coefficients {-1, 0, 1}. The polynomial $f \in L_f$ must have inverses modulo q and modulo p, which means the following must be true:

$$f \cdot f_p = 1 \bmod p \text{ and } f \cdot f_q = 1 \bmod q$$

The private key consists of the polynomials f and f_p. The public key can be calculated using the formula:

$$h = (p \cdot f_q \cdot g) \bmod q$$

Algorithm 4.2.2 (NTRUEncrypt Encryption and Decryption)**.** [9] To encrypt a message, it must be first converted in the form of a polynomial m with coefficients in $[-\frac{p}{2}, \frac{p}{2}]$. Then choose randomly a polynomial r with small coefficients, and compute the encrypted message (ciphertext):

$$c = (r \cdot h + m) \bmod q$$

To decrypt a message, perform the following calculation:

$$a = (f \cdot c) \bmod q$$

since

$$a = (f \cdot (r \cdot h + m)) \bmod q = (f \cdot (r \cdot pf_q \cdot g + m)) \bmod q = (rpg + fm) \bmod q$$

where the coefficients of a are in $[-\frac{q}{2}, \frac{q}{2}]$. Next calculate

$$b = a \bmod p$$

since

$$b = fm \bmod p$$

and recover the message using the other part of the private key

$$d = f_p \cdot b$$

since

$$d = (f_p \cdot f \cdot m) \bmod p = m \bmod p$$

The resulting polynomial d represents the original message.

The security of NTRUEncrypt is based on the computational problem on lattices called the shortest vector problem, which is stated as "given a lattice basis B, find the shortest nonzero vector in $L(B)$" [17]. This problem is difficult to compute using both a classical and a quantum computer, making the NTRU encryption algorithm quantum-safe. It is NP-hard for randomized reductions, which means nondeterministic polynomial-time hardness. New cryptographic schemes undergo a series of cryptanalytic tests before their widespread adoption in any critical infrastructure, which takes years. There are attacks that can be used on NTRUEncrypt, such as brute-force attacks, man-in-the-middle attacks, multiple transmission attacks, and lattice-based attacks [9]. Still, evidence suggests that they do not pose a security threat to the encryption [9]. According to the paper by Mavroeidis et al., "as of today there is not any known attack for NTRU" [16].

Chapter 5

Conclusion

Nowadays, secure transmission of information plays a critical role in everyday life because of the widespread use of computers in finance, healthcare, science, government, and military applications. Every year quantum computers become more powerful and less expensive, which creates a problem with modern encryption algorithms, which can be broken using quantum algorithms. When quantum computers are powerful enough to utilize Shor's and Grover's algorithms, the conventional public-key cryptosystems such as RSA and key exchange methods such as Diffie-Hellman will become insecure. Therefore, there is a need for new encryption and key distribution methods that are resistant to quantum computing.

If quantum computers become ubiquitous and quantum channels are available to every user, quantum key distribution methods such as BB84 and SARG04 can be used for secure key distribution and, consequently, secure encryption. Such protocols are provably secure under certain conditions but require a sophisticated infrastructure and are currently infeasible. If powerful quantum computers become available only to specific groups of people, the quantum key distribution would not be viable for widespread use, and lattice-based cryptography can be used instead. Post-quantum cryptographic methods such as NTRUEncrypt can be implemented using classical computers and will provide sufficient security.

This paper has provided the relevant mathematical concepts and explanations for all mentioned algorithms to highlight the flaws of the widely-used encryption methods and the future advantages of the proposed post-quantum encryption methods.

Bibliography

[1] Adetokunbo Adedoyin, John Ambrosiano, Petr Anisimov, Andreas Bärtschi, William Casper, Gopinath Chennupati, Carleton Coffrin, Hristo Djidjev, David Gunter, Satish Karra, et al. Quantum algorithm implementations for beginners. *arXiv preprint arXiv:1804.03719*, 2018.

[2] Mirko Amico, Zain H Saleem, and Muir Kumph. Experimental study of shor's factoring algorithm using the ibm q experience. *Physical Review A*, 100(1):012305, 2019.

[3] Paulo SLM Barreto, Felipe Piazza Biasi, Ricardo Dahab, Julio César López-Hernández, Eduardo M de Morais, Ana D Oliveira, Geovandro CCF Pereira, and Jefferson E Ricardini. A panorama of post-quantum cryptography. In *Open Problems in Mathematics and Computational Science*, pages 387–439. Springer, 2014.

[4] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

[5] Johannes Buchmann and Jintai Ding. *Post-Quantum Cryptography: Second International Workshop, PQCrypto 2008 Cincinnati, OH, USA October 17-19, 2008 Proceedings*, volume 5299. Springer Science & Business Media, 2008.

[6] Gabriela Bucur, Otilia Cangea, and Cristina Popescu. Encryption algorithms for data transmission security in industrial environments. *Petroleum-Gas University of Ploiesti Bulletin, Technical Series*, 68(4), 2016.

[7] Chi-Hang Fred Fung, Kiyoshi Tamaki, and Hoi-Kwong Lo. On the performance of two protocols: Sarg04 and bb84. *arXiv preprint quant-ph/0510025*, 2005.

[8] Lov K Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pages 212–219, 1996.

[9] Jeffrey Hoffstein, Jill Pipher, and Joseph H Silverman. Ntru: A ring-based public key cryptosystem. In *International algorithmic number theory symposium*, pages 267–288. Springer, 1998.

[10] Lawrence Ip. Shor's algorithm is optimal. *University of California, Berkeley, USA*, 5, 2003.

[11] Richard Jozsa. Searching in grover's algorithm. *arXiv preprint quant-ph/9901021*, 1999.

[12] Noboru Kunihiro. Quantum factoring algorithm: Resource estimation and survey of experiments. In *International Symposium on Mathematics, Quantum Theory, and Cryptography*, pages 39–55. Springer, Singapore, 2021.

[13] Carlile Lavor, LRU Manssur, and Renato Portugal. Grover's algorithm: Quantum database search. *arXiv preprint quant-ph/0301079*, 2003.

[14] Fiona Yan Lee. *An End-to-End Identity-Based Email Encryption Scheme*. PhD thesis, Boise State University, 2014.

[15] Nan Li. Research on diffie-hellman key exchange protocol. In *2010 2nd International Conference on Computer Engineering and Technology*, volume 4, pages V4–634. IEEE, 2010.

[16] Vasileios Mavroeidis, Kamer Vishi, Mateusz D Zych, and Audun Jøsang. The impact of quantum computing on present cryptography. *arXiv preprint arXiv:1804.00200*, 2018.

[17] Daniele Micciancio and Oded Regev. Lattice-based cryptography. In *Post-quantum cryptography*, pages 147–191. Springer, 2009.

[18] Michele Mosca and Artur Ekert. The hidden subgroup problem and eigenvalue estimation on a quantum computer. In *NASA International Conference on Quantum Computing and Quantum Communications*, pages 174–188. Springer, 1998.

[19] Michael A Nielsen and Isaac Chuang. Quantum computation and quantum information, 2002.

[20] Bartłomiej Patrzyk. Review, analysis and simulation of quantum algorithms in cryptography. *Master of Science Thesis supervised by Katarzyna Rycerz*, 2014.

[21] Alasdair B Price, John G Rarity, and Chris Erven. A quantum key distribution protocol for rapid denial of service detection. *EPJ Quantum Technology*, 7(1):8, 2020.

[22] Benjamin Schumacher and Michael Westmoreland. *Quantum processes systems, and information*. Cambridge University Press, 2010.

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

[24] Alan Szepieniec. Mathematical and provable security aspects of post-quantum cryptography. 2018.

[25] Lieven MK Vandersypen, Matthias Steffen, Gregory Breyta, Costantino S Yannoni, Mark H Sherwood, and Isaac L Chuang. Experimental realization of shor's quantum factoring algorithm using nuclear magnetic resonance. *Nature*, 414(6866):883–887, 2001.