DEMOCRATIC REPUBLIC OF ALGERIA PEOPLE

MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH

UNIVERSITY MOHAMED BOUDIAF - M'SILA

FACULTY: Mathematics and Informatics

DEPARTEMENT of computer science

N° : ……..

DOMAIN: Mathematics and Informatics

BRANCH: Computer Science

OPTION: RTIC

A Dissertation in Fulfillment

For the Requirement of the Degree of MASTER

By: Marrok Abderrahmane || Boukhelef Safaa

Subject:

Integrating post-quantum cryptography into VPN protocol

Defended to the jury:

Mezrag Fares	University of M’sila	Chairman
Noureddine Chikouche	University of M’sila	Supervisor
Amroune Nacereddine	University of M’sila	Examiner

Academic year: 2022/2023

بسم الله الرحمن الرحيم

Acknowledgements

In the name of Allah the merciful the merciful

Our first thanks go to God the Almighty, who has allowed us to have the health, courage, and will to carry out this work.

We would like to extend our most distinguished thanks to our family and friends, who helped us continue our education and achieve our desired goal.

We would like to extend our deepest thanks to Professor Noureddine Chikouche for his significant contributions to the field and for his unwavering support and guidance.

We are grateful for the opportunities that Pr. Noureddine Chikouche has provided us to collaborate on research projects and present our work at conferences. His guidance and belief in our abilities have empowered us to overcome challenges and strive.

Abstract

In today's interconnected world, Virtual Private Networks (VPNs) are essential for secure communication and data protection. However, with the rise of quantum computing, the security landscape is rapidly changing, necessitating advanced measures to secure VPN infrastructures. By integrating post-quantum hybrid cryptography into VPNs, organizations can fortify their security measures and safeguard sensitive information against the threats posed by quantum computing. For that we propose a hybrid cryptography approach to integrate post-quantum security into Virtual Private Network protocols. Our approach utilizes the Kyber key encapsulation mechanism (KEM) in combination with the WireGuard VPN protocol to encrypt the public key before exchanging it with the server. We conducted a security analysis of our proposed approach and compared it to classical VPN protocols like OpenVPN and WireGuard without post-quantum security. Our findings demonstrate that the proposed hybrid approach provides a higher level of security against attacks by quantum computers. the performance evaluation of our approach demonstrates that it performs comparably to the original WireGuard protocol in terms of computational overhead and latency while outperforming OpenVPN in terms of key generation and encryption speed.

Keywords: Post Quantum Cryptography, Hybrid Public Key Encryption (HPKE), Virtual Private Networks (VPNs), Wireguard, Kyber.

الملخص

في عالم الاتصالات الحالي تعد الشبكات الافتراضية الخاصة ضرورية لضمان اتصال آمن، ولكن مع ظهور الحواسيب الكمية، يتغير المشهد الأمني للمعلومات بسرعة، مما يستدعي اتخاذ إجراءات متقدمة لتأمين بنية الشبكة الافتراضية الخاصة. حتى يتسنى للنظم القائمة تعزيز أمان وحماية المعلومات الحساسة من التهديدات التي قد تنجر عن الحواسيب الكمية. لذا، نقترح نهج تشفير هجين يدمج بين تقنيات أمان ما بعد الكم مع الشبكات الافتراضية الخاصة. حيث يعتمد هذا النهج على التغليف الخاص بخوارزميات Kyber بالاشتراك مع بروتوكول WireGuard للشبكات الافتراضية الخاصة من أجل تشفير المفتاح العام قبل تبادله مع الخادم. بعد ذلك، قمنا بتحليل أداء وأمان النهج المقترح وقارناه مع بروتوكولات أخرى بدون أمان الكم. تظهر النتائج أن النهج الهجين المقترح يوفر مستوى أعلى ضد حواسيب الكم والكلاسيكية، كما يوضح أداء هذا النهج أن التعقيد الحسابي له يبقى قابلاً للمقارنة مع بروتوكول WireGuard الأصلي، في حين يتفوق بشكل كبير من ناحية سرعة توليد المفاتيح وسرعة التشفير على البروتوكولات القائمة على OpenVPN.

الكلمات المفتاحية : تشفير ما بعد الكم، تشفير المفتاح العام الهجين، الشبكات الافتراضية OpenVPN، WireGuard.

Résumé

Dans le monde interconnecté d'aujourd'hui, les réseaux privés virtuels (VPN) sont essentiels pour la communication et la protection des données. Cependant, avec la montée de l'informatique quantique, le paysage de la sécurité évolue rapidement, ce qui nécessite des mesures avancées pour sécuriser les infrastructures VPN. En intégrant la cryptographie hybride post-quantique dans les VPN, les organisations peuvent renforcer leurs mesures de sécurité et protéger les informations sensibles contre les menaces posées par l'informatique quantique. À cet effet, nous proposons une approche de cryptographie hybride pour intégrer la sécurité post-quantique dans les protocoles de réseaux privés virtuels (VPN). Notre approche utilise le mécanisme d'encapsulation de clé Kyber (KEM) en combinaison avec le protocole VPN WireGuard pour chiffrer la clé publique avant de l'échanger avec le serveur. Nous avons effectué une analyse de sécurité de notre approche proposée et l'avons comparée à des protocoles VPN classiques tels que OpenVPN et WireGuard sans sécurité post-quantique. Nos résultats démontrent que l'approche hybride proposée offre un niveau de sécurité plus élevé contre les attaques des ordinateurs quantiques. L'évaluation des performances de notre approche montre qu'elle fonctionne de manière comparable au protocole WireGuard d'origine en termes de charge de calcul et de latence, tout en surpassant OpenVPN en termes de génération de clé et de vitesse de chiffrement.

Mots clés: Cryptographie post-quantique, Cryptographie Hybride à Clé Publique (HPKE), Réseaux privés virtuels (VPN), WireGuard, Kyber.

List of Algorithms

List of Tables

List of Figures

List of Code Snippets

Contents

GENERAL INTRODUCTION

In today's digital landscape, virtual private networks (VPNs) have become a critical tool for ensuring secure communication over the Internet. VPNs rely on cryptographic protocols to ensure the confidentiality, integrity, and authenticity of data transmitted over the network. However, the growing threat of quantum computing has raised concerns about the long-term security of existing VPNs.Quantum computers are expected to break classical cryptographic algorithms, such as RSA and ECC [8], which are widely used in VPNs and other secure communication systems. To address this challenge, there has been a growing interest in developing post-quantum cryptography protocols that are resistant to attacks by quantum computers. The motivation behind our proposed scheme stems from the increasing threats of cyberattacks and the limitations of traditional cryptographic protocols.

We contribute to the research community by providing a practical and efficient solution that addresses the security concerns of VPNs in the post-quantum era. The use of a hybrid approach offers several benefits, including protection against new attacks, easier migration and backward compatibility in complex environments, and compliance with regulatory requirements such as those set by the Federal Office for Information Security in Germany and the National Agency for the Security of Information Systems in France [9]. We selected the Kyber KEM scheme as our Post Quantum Computing because it's a lattice-based cryptographic algorithm that has been selected as one of the finalists for the National Institute of Standards and Technology (NIST) Post-Quantum Cryptography

Standardization project. According to the NIST report [7], Kyber has shown strong performance in terms of speed and size. It offers a high level of security against attacks by both classical and quantum computers, making it an ideal choice for our proposed hybrid scheme. Moreover, the Kyber KEM has been optimized for performance on a wide range of platforms, including CPUs, microcontrollers, and FPGAs, making it a practical and efficient solution for our proposed integration with the WireGuard VPN protocol.

We present our contribution to the exploration of hybrid solutions for cryptography. A comprehensive analysis of our proposed hybrid scheme and compare it with some existing VPN protocols. Additionally, we evaluate the performance of our proposed scheme in terms of security, efficiency, and complexity. Our focus is on enhancing the security of VPNs in the face of evolving threats posed by quantum computing. We propose a hybrid scheme to enhance the security of VPNs and offer the "best of both worlds" approach, providing protection against both classical and quantum attacks. Our proposed scheme (HK-Wireguard) provides quantum forward secrecy while maintaining classical wireguard handshake properties and limiting the amount of exchanged data. By integrating our proposed scheme into a wireguard VPN, we can protect against current classical attacks and potential quantum attacks in the future. One key aspect of our proposed scheme is the integration of the Kyber KEM algorithm, a post-quantum lattice-based cryptographic algorithm. Kyber has been selected as a finalist in the National Institute of Standards and Technology's (NIST) Post-Quantum Cryptography [7].

This dissertation is divided into four chapters: First of all, we provide an overview of cryptography and its importance in modern society. Additionally, we discuss the basic properties of cryptography, followed by an exploration of cryptography algorithms, which are the mathematical functions and protocols used to secure and protect data. Furthermore, we introduce the concept of post-quantum cryptography and highlight the differences between classical and quantum computing. Secondly, we present the fundamentals of VPN and review relevant works in the field of integrating post-quantum cryptography into VPNs and other protocols. Several research works have examined the integration of quantum-resistant algorithms into VPNs and protocols such as OpenVPN, WireGuard, IKEv2, TLS, and SSH. In the third chapter, we present our contribution and the results obtained. Finally, the last chapter covers the implementation of our recommended approach.

CHAPTER 1

CONCEPTS OF CRYPTOGRAPHY

1.1 Introduction

Cryptography is a technique that transforms readable and understandable data into an unreadable format, called ciphertext, to protect it from unauthorized access and interception. Cryptography has become an integral part of modern society, enabling secure communication and protection of sensitive information such as financial transactions, personal data, and government secrets.

In this chapter, we will discuss the basic concepts of cryptography, including its properties, algorithms, and applications. We will also explore the differences between symmetric and asymmetric cryptography, and delve into some of the most commonly used algorithms ,such as AES, RSA, SHA, HMAC, Diffie-Hellman, and Elliptic Curve Cryptography. We'll also talk about what "post-quantum" means and how it differs from "classical" computers.

1.2 Security properties

Security properties refer to the desirable characteristics and attributes of a system or environment that contribute to its overall security.

Here are some commonly recognized security properties:

- **Confidentiality :** is the act of keeping information hidden, especially from those who are not supposed to have access to it .
- **Data integrity :** is the assurance that data is accurate and consistent throughout its lifecycle, ensuring reliability in storage, processing, and retrieval systems.
- **Authentication :** is the process of confirming someone's identity or verifying the legitimacy of a system.
- **Non-repudiation :** is an information security concept that prevents the sender of a message from denying its transmission and the recipient from denying its receipt .

1.3 Cryptography algorithms

These are several algorithms and protocols that are used to secure and protect communication and data from unauthorized access or interception. Cryptography involves the use of cryptographic techniques to ensure confidentiality, integrity, and authenticity requirements. Encryption is the process of turning plaintext into

ciphertext, which is a version of the original data that cannot be read or understood without the right decryption key. On the other hand, decryption is the process of converting ciphertext back into plaintext.

There are several types of cryptography algorithms, including symmetric-key algorithms, hash functions, digital signatures, and key exchange protocols.

Each type of algorithm has unique features, strengths, and weaknesses and is used in different applications to ensure communication and data security. Some commonly used cryptography algorithms include AES, RSA, SHA, HMAC, Diffie-Hellman, and elliptic curve cryptography.

1.3.1 Symmetric Cryptography

and also known as a symmetric key or single key because we use the same key for the encryption and decryption processes; during this process, both parties exchange that key. In symmetric cryptography, the plaintext is transformed into ciphertext using the secret key, and the ciphertext can only be decrypted back to the original plaintext using the same key. Some common symmetric-key encryption algorithms include Advanced Encryption Standard (AES), Data Encryption Standard (DES), Triple DES (3DES), Blowfish, and Twofish.

Data Encryption Standard(DES)

In data encryption, the process involves transforming data into an unintelligible form called ciphertext, while decryption reverses this process and converts the ciphertext back into its original form, known as plaintext. This standard describes an algorithm that facilitates both encryption and decryption operations, utilizing a binary number known as a key[10].DES is no longer recommended for general use,but it may still be encountered in legacy systems or certain specific situations.

Advanced Encryption Standard (AES) is a highly prevalent symmetric-key encryption algorithm renowned for being one of the most secure encryption standards currently in use. It was introduced as a replacement for the outdated Data Encryption Standard (DES) algorithm.

AES employs a block cipher technique, encrypting data in fixed-length blocks of 128 bits. It supports key sizes of 128, 192, or 256 bits, and the number of rounds in AES varies based on the chosen key length. AES is designed to withstand all known attacks and is characterized by its simplicity and efficient computational speed [11].

1.3.2 Asymmetric Cryptography

Asymmetric cryptography, commonly known as public-key cryptography, is a kind of encryption that encrypts and decrypts data using a pair of keys. They use different keys for encryption and decryption [12], The two keys are mathematically similar, but one is kept hidden (the private key), while the other is made public (the public key). Asymmetric cryptography's fundamental tenet is that anybody can encrypt a message using the public key, but only the owner of the associated private key is able to decode it. This provides safe communication between two parties without requiring them to exchange a secret key first. Among the most common asymmetric cryptography algorithms are RSA, Diffie-Hellman, and Elliptic Curve Cryptography (ECC).

Rivest-Shamir-Adleman RSA is a common public-key cryptography method for encryption, digital signatures, and key exchange.The RSA algorithm uses public and private keys. The public and private keys encrypt and decode data. The public key and private key have the same modulus but different exponents. The exponent and modulus make up the public key. The modulus' prime factors—often a massive integer that is the product of two prime numbers—determine the exponents.RSA encrypts data using the recipient's public key and decrypts it using their private key. When signing a message with RSA, the sender generates a digital signature with their private key, which anyone can verify with their public key.

Diffie-Hellman is a key exchange method that allows two parties to create a shared secret key via an unsecured channel without exchanging any sensitive information. The Diffie-Hellman algorithm creates a shared secret key between two parties based on their unique private keys and a common public key by employing modular arithmetic. While the private keys are kept private, the public key is exchanged between the two parties (see Figure 1.1).

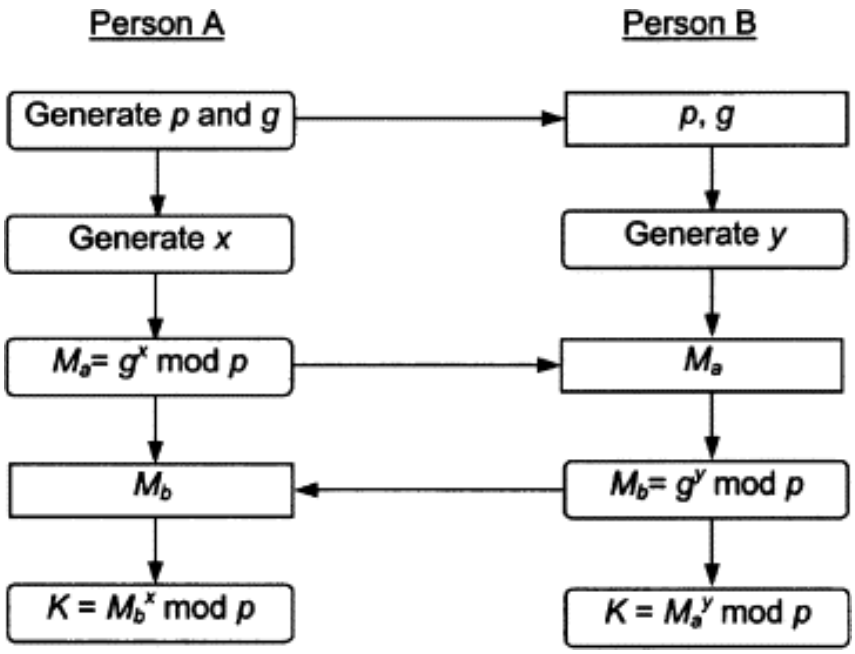

Figure 1.1: Diffie-Hellman-key-exchange-for-authentication [2]

. .

Elliptic Curve Cryptography ECC is based on the algebraic structure of elliptic curves, which are a type of curve defined by a mathematical equation. ECC uses a public key and a private key, exactly like other public-key cryptography methods. In contrast to the private key, which is needed to calculate the public key, the public key is a point on the elliptic curve. When two parties want to create a shared secret key, ECC follows a similar procedure to Diffie-Hellman.

1.3.3 Hash Function

Hash functions are mathematical functions that accept any length input and give a fixed-length output. The output is a condensed version of the input, and every little change in the input produces an entirely different outcome. The hash function should be developed in such a way that determining the original input from the hash value is computationally impossible. SHA-256, SHA-3, and BLAKE2 are a few of the well-known hashing algorithms.

1.3.4 Digital signature

A digital signature is an electronic mechanism that establishes a connection between an entity and a data record. Its primary function is to validate and authenticate electronic

documents [13] .

1.3.5 Key exchange protocols

Key exchange protocols are cryptographic techniques used to establish a secure session key between two users communicating over a public and potentially insecure channel. These protocols ensure that the session key is securely shared between the users, despite the presence of potential adversaries or eavesdroppers on the communication channel [14].

1.4 Post Quantum Cryptography

1.4.1 Definitions

Post-quantum cryptography is a branch of cryptography where encryption methods are made that can't be broken by an opponent with a quantum computer. All 10 of the current asymmetric methods (RSA, ECC, DH, and DSA) can be broken with a quantum computer. They are based on the Prime Factoring Problem or the Discrete Logarithm Problem, which is easy to solve on quantum computers with Shor's Algorithm[15].

Post-quantum cryptography (PQC) is a field of cryptography that focuses on developing new cryptographic algorithms and protocols that are resistant to attacks by quantum computers. These algorithms use mathematical problems that are believed to be too difficult for even the most advanced quantum computers to solve.

Classic Cryptography "Classic cryptography" refers to the traditional cryptographic techniques that have been used for centuries. These techniques rely on mathematical algorithms and keys to encrypt and decrypt data. Classic cryptographic systems include symmetric-key algorithms like the Data Encryption Standard (DES) and Advanced Encryption Standard (AES) and asymmetric-key algorithms like RSA and Diffie-Hellman.

Quantum Cryptography Quantum cryptography is a branch of cryptography that leverages the principles of quantum mechanics to provide secure communication channels. Unlike classic cryptography, which relies on computational complexity, quantum cryptography utilizes the laws of physics to ensure the security of information. Quantum key distribution (QKD) is a prominent example of quantum cryptography, where encryption keys are shared using quantum properties such as entanglement and

the uncertainty principle.

1.4.2 Types of post-quantum Cryptography

Post-quantum cryptography refers to the development of cryptographic algorithms that can effectively resist attacks from quantum computers. These algorithms can be categorized into different types, as shown in Figure 1.2:

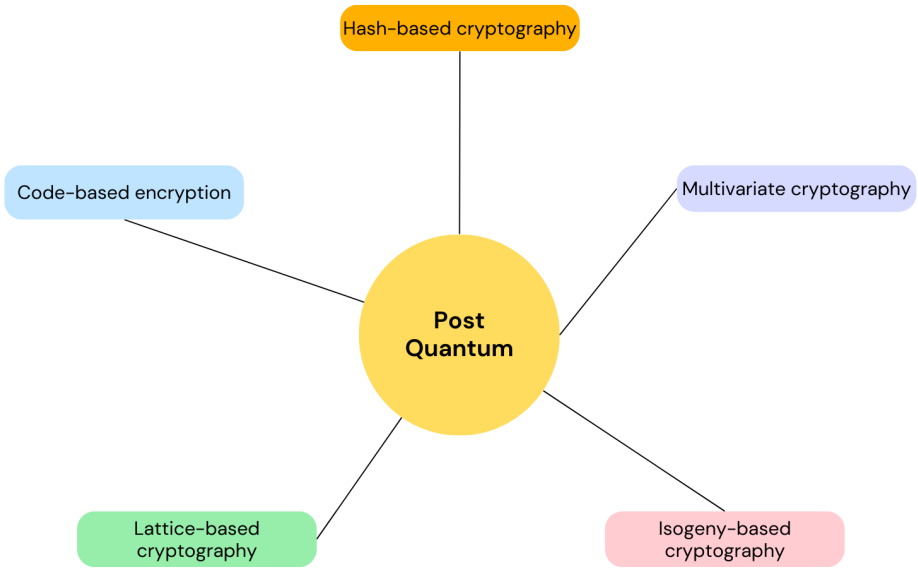

Figure 1.2: Basic types of PQC .

1. **Lattice-based cryptography**: is a form of encryption that relies on the difficulty of certain mathematical problems related to lattices. A lattice is a discrete grid-like structure formed by a set of basis vectors. In two dimensions, as shown in Figure 1.3, the lattice is represented by a grid of points that can be reached by integer combinations of two basis vectors, denoted as $\mathbf{b}_1$ and $\mathbf{b}_2$.

   In lattice-based cryptography, the Shortest Vector Problem (SVP) and the Closest Vector Problem (CVP) play significant roles. The SVP involves finding the shortest non-zero vector, denoted as $\mathbf{v}$, within a given lattice L. Mathematically, it can be stated as finding $\mathbf{v} \in L$ such that $|\mathbf{v}| \leq \lambda \cdot \lambda(L)$, where $|\mathbf{v}|$ denotes the Euclidean norm of vector $\mathbf{v}$ and $\lambda(L)$ represents the length of the shortest vector in lattice L.

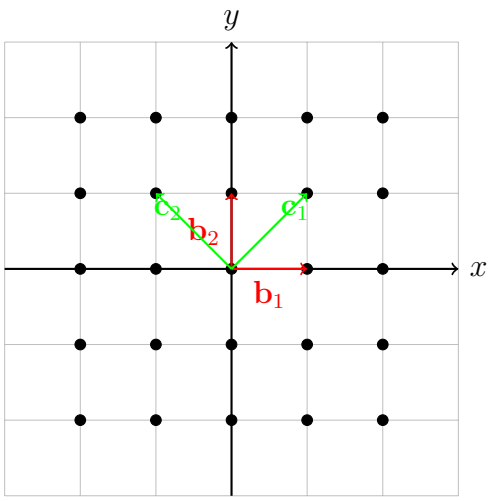

Figure 1.3: Two-dimensional lattice with two possible bases

In the lattice diagram above, $\mathbf{v}$ would correspond to the shortest vector connecting the lattice points.

On the other hand, the CVP deals with finding the lattice point within L that is closest to a specified target vector $\mathbf{t}$. it can be formulated as finding $\mathbf{v} \in L$ such that $|\mathbf{v} - \mathbf{t}|$ is minimized, where $|\cdot|$ represents the Euclidean distance. The CVP is concerned with finding the lattice point that best approximates the target vector $\mathbf{t}$.

Both the SVP and CVP are computationally challenging problems, particularly as the dimensions of the lattice increase. However, their hardness forms the foundation of lattice-based cryptography. By leveraging the difficulty of these problems, lattice-based encryption schemes provide a basis for secure communication and cryptographic primitives. One notable advantage of lattice-based cryptography is its resilience against attacks by quantum computers, offering post-quantum security. This property makes lattice-based cryptography a promising area of research for developing secure communication protocols in the presence of emerging quantum technologies.[16].

The **Learning With Errors (LWE)** problem is another important aspect of lattice-based cryptography. It involves estimating a set of random linear equations with some error. Specifically, given a secret vector $\mathbf{s}$ and a random matrix $\mathbf{A}$, the LWE problem entails finding a vector $\mathbf{e}$ such that $\mathbf{A} \cdot \mathbf{s} + \mathbf{e}$ is indistinguishable from random. Solving the LWE problem is challenging, especially when the noise

term **e** is large, making it difficult to recover the secret vector **s**.

2. **Code-based encryption**: Code-based encryption is a cryptographic technique that is predicated on the difficulty of decoding specific error-correcting codes. It relies on the concept that certain decoding problems, such as the Syndrome Decoding Problem, are computationally hard to solve.

   In code-based encryption, the security of the scheme is based on the hardness of decoding random linear codes. The underlying idea is that decoding problems, such as finding the original message from the received noisy ciphertext, are computationally infeasible without possessing certain secret information.

3. **Hash-based cryptography**: Hash-based cryptography, originally developed by Leslie Lamport and Ralph Merkle in the late 1970s, is a cryptographic scheme that relies on the use of hash functions. The Merkle signature scheme and the XMSS (Extended Merkle Signature Scheme) serve as notable examples of hash-based cryptography. In the Merkle signature scheme, a binary tree structure, known as a Merkle tree, is employed to create a digital signature. On the other hand, the XMSS utilizes a binary tree structure with multiple levels for digital signature generation. Both schemes are built upon the fundamental properties of hash functions, namely one-wayness and collision resistance, and are widely regarded for their potential to provide post-quantum security [17].

   The security of both the Merkle signature scheme and the XMSS is derived from the one-wayness and collision resistance properties of hash functions. These properties make it exceedingly difficult to forge a signature or discover two messages that produce the same signature. Hence, hash-based cryptography offers robust security features for various cryptographic applications.

4. **Isogeny-based cryptography**: is a cryptographic framework that relies on the computational hardness of finding isogenies, which are mappings between elliptic curves. These mappings preserve certain algebraic properties and are fundamental in isogeny-based cryptography. The security of isogeny-based cryptography is based on the difficulty of computing isogenies between elliptic curves over finite fields.

   The difficulty lies in identifying and computing isogenies, which are mappings between elliptic curves that preserve certain algebraic properties. In particular,

finding an isogeny between two given elliptic curves can be computationally challenging, especially when the curves are defined over large finite fields.

5. **Multivariate cryptography**: is a type of cryptography that exploits the difficulty of solving systems of multivariate polynomial equations. These equations typically take the form of quadratic equations involving multiple variables. The coefficients and variables in these equations belong to a finite field, which consists of a set of finite numbers used for mathematical operations.

   The security of multivariate cryptography is rooted in the hardness of a specific mathematical problem known as **Problem MQ**(Multivariate Quadratic polynomials) .

   It is crucial to note that Problem MQ is widely recognized as an NP-hard problem, meaning that it is considered challenging to solve unless the complexity classes P and NP are equivalent. The security of multivariate cryptography schemes heavily relies on the hardness of this problem, as it ensures protection against attacks based on solving systems of multivariate equations [18, 19].

1.4.3 Classical vs Quantum Computing

Classical computing operates on classical bits, which are either in the state 0 or 1. Quantum computing, on the other hand, uses quantum bits, or qubits, which can be in a superposition of states, meaning that they can exist in a combination of 0 and 1 at the same time. This property of qubits enables quantum computers to perform certain computations exponentially faster than classical computers [20].

The difference between bits and qubits can be expressed as follows:

A classical bit can be in one of two states:

$$|0\rangle = \begin{bmatrix} 1 & 0 \end{bmatrix}, \quad |1\rangle = \begin{bmatrix} 0 & 1 \end{bmatrix} \tag{1.1}$$

where $|0\rangle$ represents the state of a classical bit being in the 0 states, and $|1\rangle$ represents the state of a classical bit being in the 1 state.

In contrast, a qubit can be in a superposition of the 0 and 1 states:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle \tag{1.2}$$

where α and β are complex amplitudes such that $|\alpha|^2 + |\beta|^2 = 1$. This means that

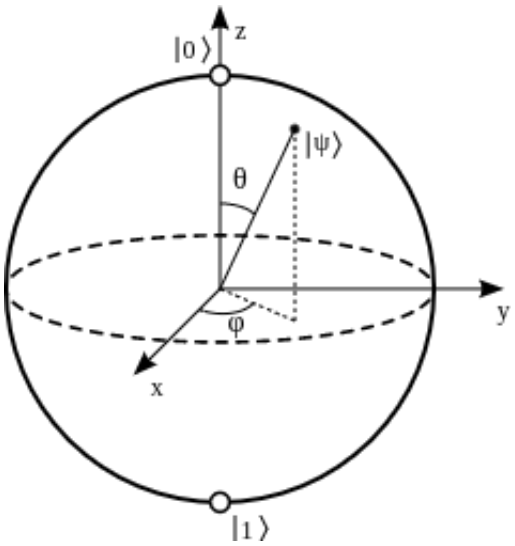

Figure 1.4: surface of the Bloch sphere

when a qubit is measured, the probability of observing it in the 0 state is $|\alpha|^2$ and the probability of observing it in the 1 state is $|\beta|^2$.

Additionally, quantum computing also allows for entanglement, where two or more qubits can become correlated in such a way that the state of one qubit depends on the state of the other(s), even if they are physically separated from each other. This property enables quantum computers to perform certain types of computations even faster than with just superposition alone.

However, while quantum computing offers advantages over classical computing in terms of speed and the ability to solve certain problems more efficiently, it also poses a threat to classical cryptographic algorithms. This is because quantum computers can use a quantum algorithm called Shor's algorithm to efficiently factor large numbers, which is the basis for many common public-key cryptography algorithms such as RSA. Therefore, it is important to develop and use post-quantum cryptography to protect against attacks by quantum computers.

Shor's and Grover's Algorithms

In the field of factorization, the General Number Field Sieve (GNFS) is the most efficient classical algorithm for large numbers. Its complexity is subexponential, approximately $O\left(\exp\left(\left(\frac{64}{9}\right)^{\frac{1}{3}} \cdot (\ln N)^{\frac{1}{3}} \cdot (\ln \ln N)^{\frac{2}{3}}\right)\right)$, where N represents the number factored. However, despite this subexponential complexity, the runtime of the GNFS escalates quickly with the size of the input number, rendering it impractical for extremely large values [21].

Contrastingly, Shor's algorithm [22] operates on an entirely different level, where N is the number undergoing factorization, The key to Shor's algorithm's efficiency lies in its ability to exploit quantum parallelism and the properties of quantum entanglement, so it presents a remarkable breakthrough in exponential factorization speed with a complexity of $O\left((\log N)^3\right)$, Its exponential speedup over classical factorization algorithms highlights the power of quantum computing and its potential to revolutionize various fields, including cryptography and number theory.

In the realm of unstructured search problems, classical algorithms typically have a linear time complexity, with an average of $O(N)$ steps to locate a specific item within an unsorted database of size N. However, Grover's algorithm [23] provides a significant speedup, operating at a complexity of $O(\sqrt{N})$, when executed on a quantum computer. This quadratic speedup makes Grover's algorithm substantially faster than the best-known classical algorithms.

Table 1.1 compares classical cryptography , quantum cryptography and post quantum cryptography.

Aspect	Classic Cryptography	Quantum Cryptography	Post-Quantum Cryptography
Security	Vulnerable to quantum attacks	Offers unconditional security based on quantum principles	Provides long-term security against quantum attacks
Implementation	Well-established and widely understood	Practical implementation challenges due to specialized quantum hardware	Compatible with classical computing infrastructure
Key Exchange	Depends on mathematical algorithms and key distribution	Secure key exchange through quantum properties (QKD)	Secure key exchange using quantum-resistant algorithms
Performance	Efficient encryption and decryption processes	Practical implementation challenges and performance considerations	Performance and efficiency considerations compared to classic algorithms

Table 1.1: Comparison of Cryptographic Approaches.

1.4.4 The PQC Hybrid approach

The hybrid approach to cryptography is a promising solution for ensuring security in the post-quantum era. It involves using both classical and post-quantum cryptographic algorithms in a complementary way to address the limitations of each. Post-quantum algorithms are expected to be secure against attacks from both classical and quantum computers [24].

Strengths of Classical and Post-Quantum:

One of the main advantages of the hybrid approach is that it allows for the use of both classical and post-quantum cryptographic algorithms. Classical algorithms have been extensively studied and tested, and they are generally faster than post-quantum algorithms. On the other hand, post-quantum algorithms are believed to be secure against attacks from both classical and quantum computers. By using both types of algorithms, the strengths of each can be leveraged while mitigating their weaknesses [25].

Higher Level of Security:
The hybrid approach provides a higher level of security than classical cryptography alone. By using post-quantum algorithms alongside classical algorithms, the system is protected against both classical and quantum attacks. This means that even if an attacker has access to a quantum computer, they will still be unable to break the encryption. Furthermore, the hybrid approach can serve as a fallback in case of a breach in post-quantum cryptography. This ensures that the system remains secure even in the face of new attacks and vulnerabilities [25].

Easy Transition to Post-Quantum Cryptography:
Another advantage of the hybrid approach is that it allows for an easy transition to post-quantum cryptography. As post-quantum algorithms become more standardized and widely adopted, the hybrid approach can be modified to include a greater proportion of post-quantum algorithms. This makes it easier for organizations to transition to post-quantum cryptography without having to completely overhaul their existing cryptographic systems [26].

Compatibility :
Finally, the hybrid approach is compatible with existing cryptographic standards and protocols. This means that it can be integrated into existing systems and infrastructure without requiring significant changes. This is important for organizations that have invested heavily in their cryptographic systems and want to ensure that their systems remain compatible with new cryptographic technologies [26].

In summary, the hybrid approach to post-quantum cryptography provides a higher level of security than classical cryptography alone while still leveraging the strengths of classical algorithms. It also enables an easy transition to post-quantum cryptography, allows for performance optimization, and is compatible with existing cryptographic standards and protocols.

1.5 Conclusion

Cryptography plays a crucial role in securing communication and data in today's digital world. Cryptography algorithms provide different levels of security and are used for various purposes, such as confidentiality, data integrity, authentication, and non-repudiation. However, with the emergence of quantum computing, there is a

growing concern that some of the commonly used cryptography algorithms, particularly those based on asymmetric cryptography, may become vulnerable to attacks. This has led to the development of post-quantum cryptography, which involves using new cryptographic algorithms resistant to quantum attacks.

CHAPTER 2

OVERVIEW ON VIRTUAL PRIVATE NETWORK AND RELATED WORKS

2.1 Introduction

Virtual Private Networks (VPNs) have become increasingly popular in recent years due to their ability to provide a secure and private connection over the Internet. VPNs allow users to connect to a private network from anywhere in the world, enabling them to access resources and information that would typically only be available within the network's physical boundaries.

This chapter provides an overview of VPNs, including their definitions, topologies, and tunneling protocols. It also includes a summary of related work and research.

2.2 Definition

VPN is a communications environment that allows peer connections only within a defined community of interest[27]. A VPN creates a logical safe channel for communication between two entities over the Internet by employing the tunneling technique, which encapsulates the IP datagram into a tunneling protocol to protect the original data from attackers or hackers who are present in nearly all networks [28] as in Figure 2.1. VPNs provide the following security features [29]:

1. **Privacy**: Prevents intermediate users from viewing, changing, or deleting the data in a packet that's being carried over the channel.

2. **Authentication**: confirms that a legitimate user sent the VPN packet.

3. **Data Integrity**: Determines whether the data packets are unchanged during transmission.

4. . **Anti- Replay**: Stops intermediate users from duplicating and resending packets sent by a legitimate user.

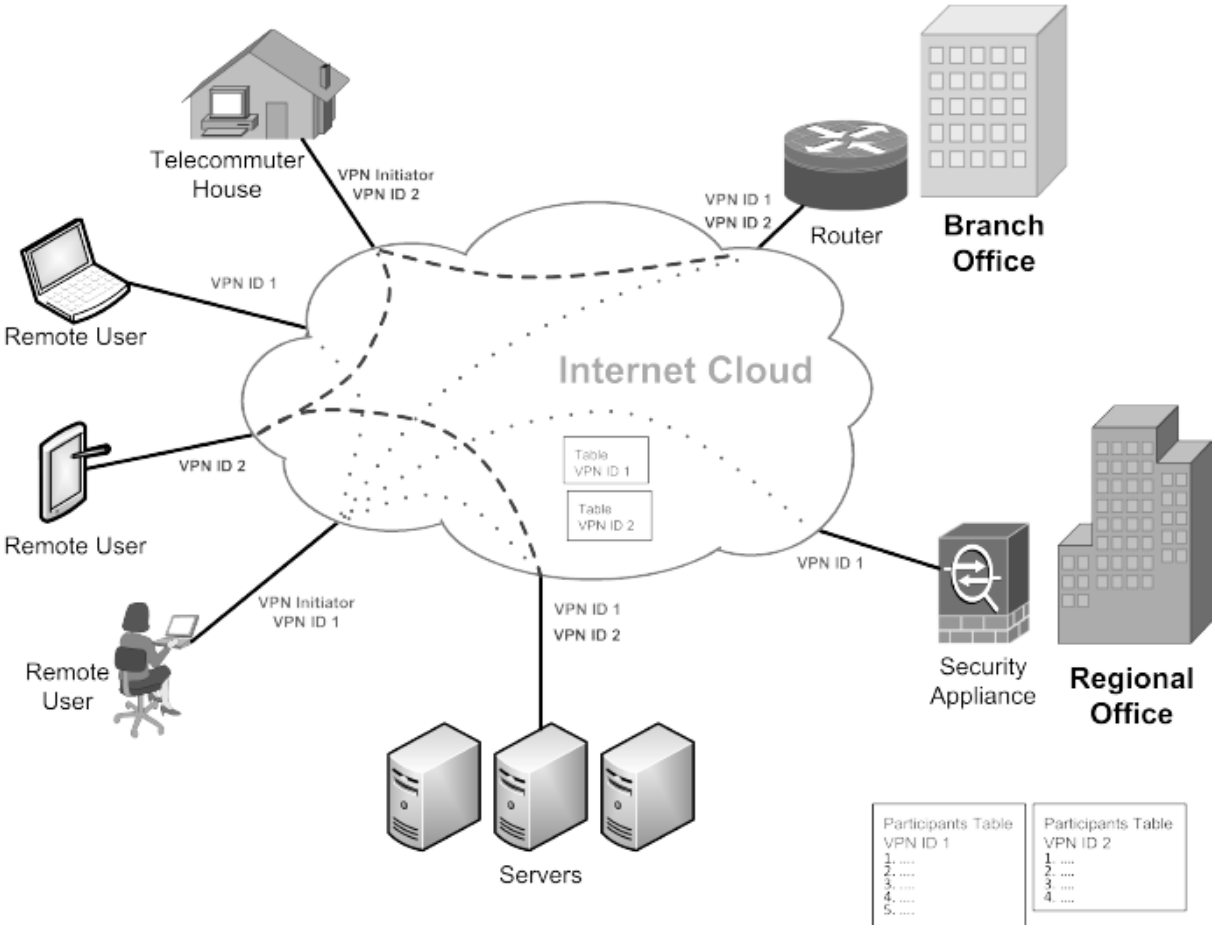

Figure 2.1: Virtual Private Network Architecture [3].

2.3 VPN Topology

There are various types of VPN topologies.

2.3.1 Site-to-Site VPN

One of the most frequent situations is this one. This entails connecting the websites of two different businesses, or the websites of businesses and their clients, suppliers, or customers. However, it is also essential that all or a portion of the machines on the two networks be able to connect to those on the distant network using the private addresses of each network. Typically, to set up this form of VPN, two hardware components (routers or firewalls) that are situated at the site's internal network and public network boundaries are connected. These pieces of hardware support packet authentication, routing, and encryption. Site to-site VPNs allow connectivity between an organization's geographically

dispersed sites (such as a head office and branch offices). Figure 2.2 illustrates a typical site-to-site VPN[30].

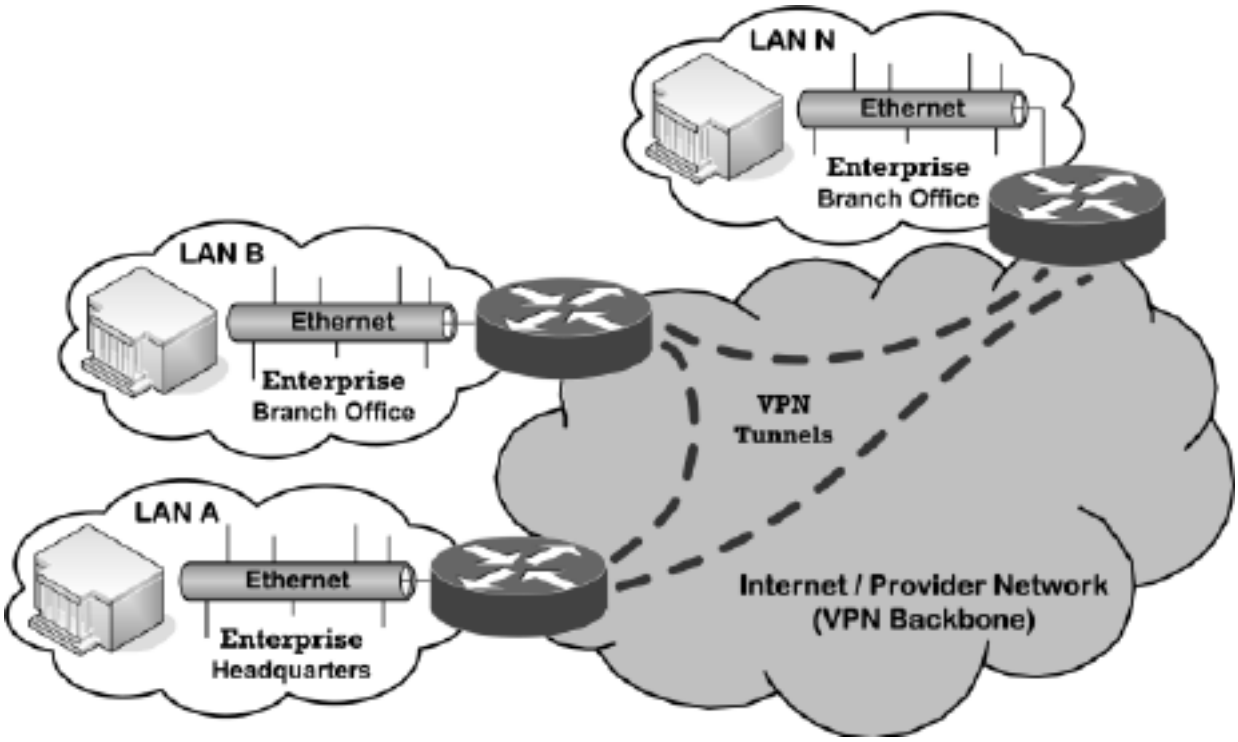

Figure 2.2: Typical site-to-site VPN [4].

2.3.2 Remote Access VPN

With a VPN called a remote access VPN, remote users can safely connect to a company network online, as we see in Figure 2.3. It enables users to access resources like files, programs, and email by establishing a secure and encrypted connection between their device and the corporate network. For Remote Access VPN to function, a secure tunnel must be built between the user's device and the company network. The user's device connects to the business network's VPN server, which establishes a secure tunnel and verifies the user's identity. Once the tunnel has been built, the user can use the corporate network resources as if they were physically present in the workplace.

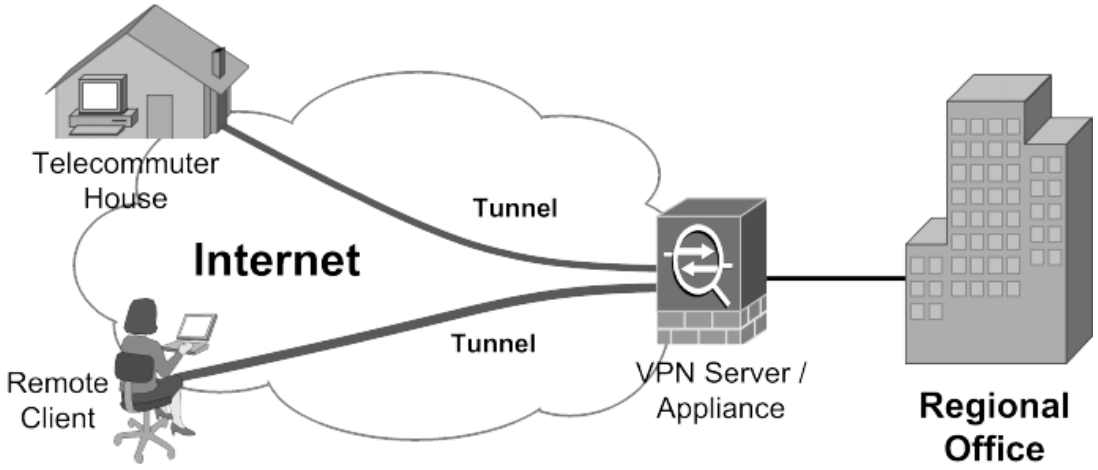

Figure 2.3: Remote-access VPN [3] .

2.3.3 Client-to-Site VPN

This topology is similar to the remote access VPN, except it requires the user's device to have client software installed. This enables the user to connect to the business network safely from any location. We can say that the selection of a VPN topology will be based on the organization's particular requirements, including the number of sites, the amount of security needed, and the types of resources that need to be accessed.

2.4 Tunneling protocols

A tunnel is a means of forwarding data across a network from one node to another as if the two nodes were directly connected [31]. This connection is created using a tunneling protocol, which encapsulates the data being transmitted and sends it through the tunnel. Tunneling is commonly used in VPNs to create a secure connection between a user's device and a private network, as shown in Figure 2.4.

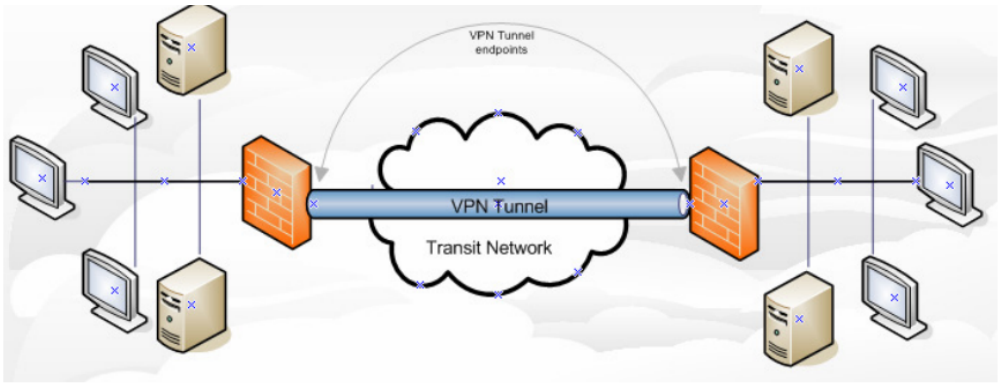

Figure 2.4: Tunneling VPN mode [5].

2.4.1 Point-to-Point Tunneling Protocol

Is among the oldest and most popular tunneling protocols. Although it is easy to start up and offers quick data rates, its security has come under scrutiny. PPTP is an extension of point-to-point protocol (PPP) [5].

2.4.2 Layer 2 Tunneling Protocol

L2TPv3 (Layer 2 Tunneling Protocol version 3) is a protocol designed for tunneling layer 2 information across a layer 3 network. As such, it is not surprising that the protocol is particularly useful for layer 2 VPNs. Happily, for the VPN provider, scalability is good – tunnels only consume resources at the tunnel endpoints, and the tunnels can be multiplexed. Although L2TP does not have any built-in security, L2TP can be run over transport mode IPsec, providing a healthy level of security for the Layer 2 VPN[31].

2.4.3 Internet Protocol Security

is an open standard framework for securing network communication through the use of cryptographic security services. It operates at OSI Layer 3 (network layer) and provides various security features such as peer authentication, data origin authentication, data integrity, data confidentiality, and replay protection.IPSec supports the establishment of Virtual Private Network (VPN) tunnels, which allow secure communication between two networks or between a remote user and a network. It can be implemented in two encryption modes: Transport mode and Tunnel mode. In Transport mode, only the data portion (payload) of each packet is encrypted, while the header remains untouched. This mode is typically used to secure communication within a network.

On the other hand, Tunnel mode encrypts both the header and the payload of each

packet. It is commonly used when communication needs to traverse unknown or untrusted third-party networks, providing a higher level of security. Tunnel mode is typically used for network-to-network communication. Is a widely used protocol for internet traffic encryption that is frequently used in VPN systems to safeguard data transmission over open networks. Another tunneling protocol that is very popular for building VPNs is tunnel mode IPsec,which has several desirable features. Since IPsec has been developed for security reasons[31].

2.4.4 WireGuard

The WireGuard protocol is a secure method for two parties, the initiator and responder, as shown in 2.5, to establish a virtual private network (VPN) over an insecure communication channel. The protocol operates at layer 3 and is implemented as a kernel virtual network interface for Linux. It aims to replace traditional VPN solutions like IPsec and OpenVPN with a more secure, performant, and user-friendly alternative. It uses the NoiseIK [32] key exchange protocol, which is a single round-trip. It also uses ChaCha20Poly1305 [33] authenticated encryption for the encapsulation of packets in UDP and IP-binding cookies for mitigating denial-of-service attacks[34].

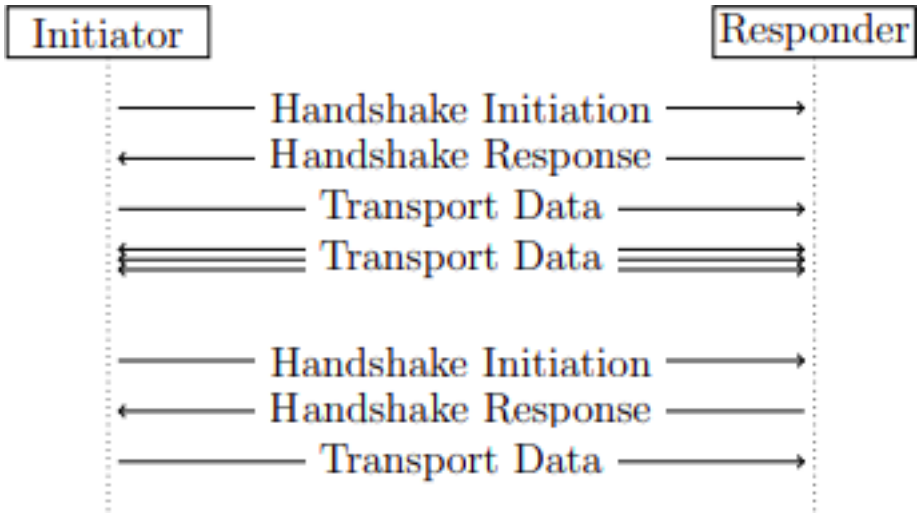

Figure 2.5: WireGuard Protocol overview [6].

2.4.5 Multi-Protocol Label Switching

Multi-Protocol Label Switching (MPLS) is a technology being standardized by the IETF. It enables high-speed data forwarding and bandwidth reservation. MPLS operates by forwarding packets through a designated tunnel using attached labels, without examining the contents of the IP header. When a packet enters an MPLS tunnel at the ingress node, a label is added to it. Subsequent nodes forward the packet based on the incoming interface and label, passing it along to the next node with a new label value. Finally, the last node in the MPLS tunnel removes the label and forwards the packet to its intended destination. The route taken by the data through the network is referred to as a" Label Switched Path" (LSP) [31].

2.4.6 OpenVPN

An open-source VPN protocol called OpenVPN offers a safe and adaptable method for establishing VPN connections. It is renowned for being highly secure, simple to use, and able to function on a variety of platforms [35].

2.5 VPN Architecture

A VPN architecture typically consists of three main components:

2.5.1 VPN client

A program that establishes a secure connection while running on the end user's device.

2.5.2 VPN server

is a remote server program that encrypts and decrypts data for a VPN client. It is a critical component of any VPN setup, as it provides a secure connection that allows users to access private networks from anywhere in the world.

2.5.3 VPN tunnel

A private, encrypted link between the VPN client and server. The VPN tunnel, which offers a secure channel for data transmission, is built using the VPN protocol and encryption algorithm. By encrypting data and authenticating users, a VPN's architecture is created to offer secure, private communication over public networks.

Users can connect to private networks, such as business or home networks, from far-off locations while maintaining the confidentiality and integrity of their data by using a VPN.

2.6 VPN Security Algorithms

There are several algorithms used in VPN communication to ensure the security and privacy of the data being transmitted. These algorithms are used to encrypt and decrypt the data that is sent between devices over the VPN connection. Here are some of the common algorithms used in VPN communication:

2.6.1 Encryption algorithms

Encryption algorithms are used to secure data that is transmitted over the VPN connection. Some commonly used encryption algorithms are:

- Advanced Encryption Standard (AES).
- Triple Data Encryption Standard (3DES).
- Blowfish.

2.6.2 Hashing algorithms

These algorithms are used to generate a unique value (hash) for the data that is sent over the VPN connection. This hash is used to verify the integrity of the data and ensure that it has not been tampered with during transmission. Among the frequently employed hashing algorithms are: SHA (Secure Hash Algorithm) and MD5 (Message Digest 5).

2.6.3 Key exchange algorithms

Key exchange algorithms are used to establish a secure key that is used for encrypting and decrypting data transmitted over the VPN connection. Some commonly used key exchange algorithms are: Rivest–Shamir–Adleman (RSA), Diffie-Hellman, and Elliptic curve diffie-hellman. These algorithms cooperate to ensure that the data being transmitted over the VPN connection is secure, private, and cannot be intercepted or altered by unauthorized third parties.

2.7 Related works

In the field of integrating post-quantum cryptography into VPNs and other protocols, several research works have been conducted.

2.7.1 Integrating Post Quantum into VPNs

Heesch et al. [36] incorporate updated OpenSSL libraries into OpenVPN and test several quantum-safe algorithms for TLS versions 1.2 and 1.3 utilizing HTTPS and OpenVPN separately.

Simon de Vries[37] proposed a scheme to integrate the Niederreiter cryptosystem into the OpenVPN protocol in order to achieve 128-bit security against quantum attacks and gave a study of how much Grover's algorithm can speed up current attacks on Niederreiter and McEliece and how to code parameters to protect against these attacks.

The choice of OpenVPN over WireGuard makes it more difficult to implement and maintain, and it also makes it more vulnerable to security vulnerabilities.

Hülsing et al. [38] present PQ-WireGuard, a postquantum variant of the handshake in the WireGuard VPN protocol. To achieve this,they replace the Diffie-Hellman-based handshake with a more generic approach only using key-encapsulation mechanisms (KEMs). The paper does not consider the use of a hybrid scheme,which would be more secure than the proposed variant of WireGuard because it would combine the strengths of both classical and post-quantum cryptographic algorithms. Classical cryptographic algorithms are well-understood and have been well tested.

2.7.2 Integrating Post Quantum into other protocols

Scott et al. [39] present delineates an expansion of the Internet Key Exchange version 2 (IKEv2) protocol, which enables it to exhibit resistance against quantum computing by leveraging preshared keys. Douglas Stebila [40] works on integrating post-quantum cryptographic algorithms into communication protocols and applications, such as TLS and SSH. The authors of. The authors of Alkim et al. [41] proposed a hybrid key exchange protocol for SSL/TLS that uses a combination of classical Diffie-Hellman and the NTRUEncrypt post-quantum algorithm.

Eric et al. [42] said The adoption of post-quantum cryptography will rely on the integration of algorithms for quantum-resistant key exchange and digital signature schemes into standards for communication protocols and other components of the IT

infrastructure, following their selection by standards bodies. This paper delves into adapting two significant Internet security protocols, namely the Transport Layer Security (TLS) and Secure Shell (SSH) protocols, to incorporate post-quantum cryptography.

Sebastian et al .[43] involves the integration of mixed certificate chain authentication with the utilization of the lattice-based key encapsulation mechanism (KEM) CRYSTALS-Kyber, which serves as a representative for post-quantum cryptography (PQC) KEMs.The objective is to assess a completely post-quantum and mutually authenticated TLS 1.3 handshake.

Marcin et al. [44] suggest that quantum key sharing be used to help end-user verification. Quantum protocols, like the BB84 protocol, can be used to share pre-shared keys, which are codes that prove who the end users are. This way, the pre-shared keys are set up using the safest transmission methods available today.

2.8 Conclusion

This chapter provides a comprehensive overview of VPNs, including their definitions, topologies, tunneling protocols, architecture, security algorithms, and related research. It highlights the importance of VPNs in establishing secure and private connections and showcases the ongoing efforts to incorporate post-quantum cryptography into VPNs and other communication protocols.

CHAPTER 3

DESIGN AND SECURITY ANALYSIS

3.1 Introduction

In this chapter, we will present the design of a hybrid approach that combines contemporary VPN technologies with post-quantum cryptography to enhance the security of encrypted communication. Specifically, we propose to encrypt the Wireguard public key using the Kyber shared secret with Kyber KEM. This approach ensures that even if a quantum attacker intercepts the communication, they cannot acquire the private key or shared secret key using Shor's or Grover's algorithms.

Moreover, by incorporating Kyber KEM into the Wireguard protocol, our design adds an extra layer of protection for encrypted communication while maintaining the outstanding security and performance of the original Wireguard protocol. We will provide an informal security analysis of our proposed scheme and compare it with other protocols to evaluate its effectiveness.

3.2 Selected VPN Protocol

We selected WireGuard as our VPN protocol due to its simplicity, efficiency, and high security. WireGuard is a modern VPN protocol that offers excellent performance and security features. It is designed to be fast, lightweight, and easy to configure. WireGuard uses state-of-the-art cryptographic algorithms to provide strong encryption and authentication.

Compared to other VPN protocols such as OpenVPN and IPsec, WireGuard has a smaller codebase, making it easier to audit and less prone to security vulnerabilities. WireGuard also uses fewer computational resources, making it an ideal choice for low-power devices such as smartphones and routers[34].

WireGuard offers several advantages over other VPN protocols, including:

1. High-performance: WireGuard is designed to be fast and efficient, providing excellent performance even on low-power devices.

2. Simplicity: WireGuard has a small codebase and a simple configuration, making it easy to set up and use.

3. Security: WireGuard uses state-of-the-art cryptographic algorithms to provide strong encryption and authentication.

4. Cross-platform compatibility: WireGuard is available on a wide range of platforms,

including Linux, macOS, Windows, Android, and iOS.

5. Transparency: WireGuard is an open-source protocol, meaning that its source code is available for review and audit.

   Overall, we believe that WireGuard is an excellent choice for our VPN solution, and we are confident that it will provide the necessary security and performance for our application.

3.3 Chosen Cryptographic Algorithms

After careful consideration, we have decided to use the Kyber cryptographic algorithm for our VPN implementation. Kyber is a post-quantum cryptographic algorithm that offers high security and performance. It has been designed to resist attacks from both classical and quantum computers, making it an ideal choice for secure communications in the age of quantum computing.

Kyber is based on lattice-based cryptography, which is considered to be one of the most promising approaches for post-quantum cryptography. Kyber has also been selected as one of the finalists for the NIST Post-Quantum Cryptography Standardization Project [7], which is a strong endorsement of its security and performance. It's renowned for its exceptional security and performance characteristics. It has been purposefully designed to withstand attacks from both classical and quantum computers, making it an excellent choice for ensuring secure communications in the era of quantum computing. Kyber demonstrates strong performance across various platforms, making it suitable for diverse application scenarios. The NIST report highlights Kyber's impressive speed and efficiency, as demonstrated by the benchmarks shown in Figure 3.1 [7]. These factors contribute to its practicality and effectiveness in real-world deployments.

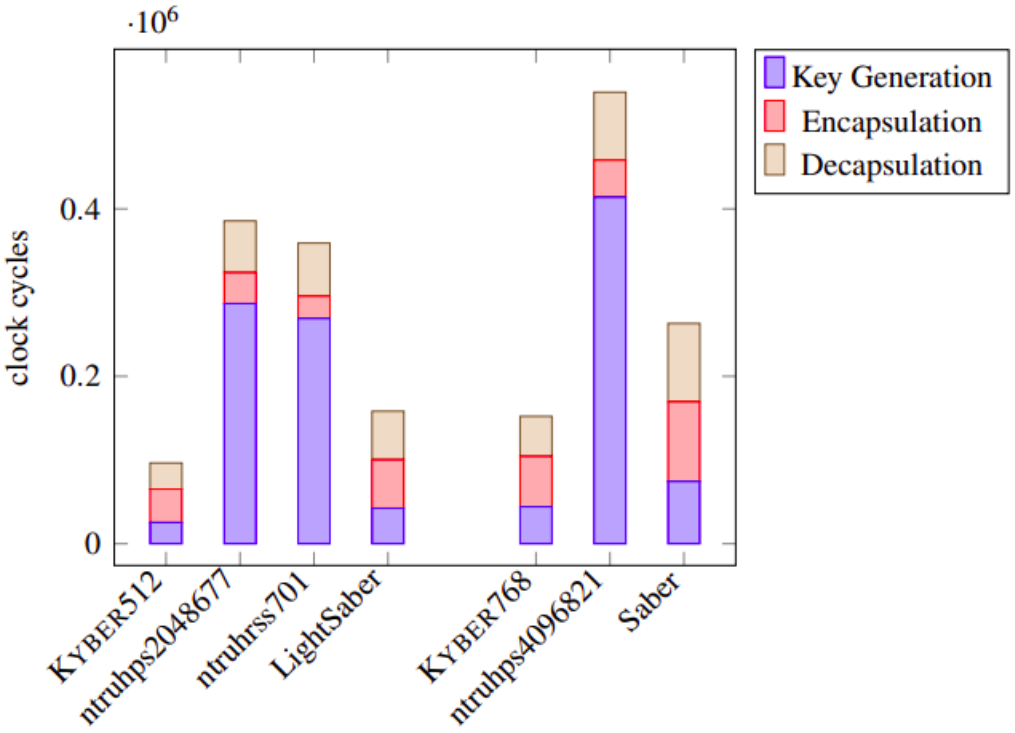

Figure 3.1: Kyber Key Encapsulation Mechanism (KEM) benchmarks on x86-64 processors with AVX2 extensions [7].

3.3.1 Kyber Algorithms

Kyber is a post-quantum cryptography technique based on the hardness of lattice-based mathematics problems. It is a key encapsulation mechanism (KEM) used to securely exchange a secret key between two parties to build a secure communication channel. We will see how in Algorithms 1 ,2 and 3. It is defined by a tuple: KeyGen; Encaps; Decaps.

The adoption of CRYSTALS-Kyber by both NIST and the NSA as a standardized public-key encryption and key encapsulation mechanism highlights its reliability and security features. This selection marks a significant milestone in the field of cryptography, as it reinforces the potential of CRYSTALS-Kyber to safeguard sensitive information in national security systems. The robustness of CRYSTALS-kyber's encryption algorithm and key encapsulation mechanism makes it an ideal choice for secure communication[45].

Algorithm 1 Key Generation for Kyber.CPA

1: $\rho, \sigma \leftarrow \{0,1\}^{256}$
2: $A \leftarrow R_q^{kk} \leftarrow Sam(\rho)$
3: $(s, e) \leftarrow \beta_{\eta k}\beta_{\eta k} \leftarrow Sam(\sigma)$
4: $t \leftarrow Compress_q(As + e, d_t)$
5: **return** $(pk \leftarrow (t, \rho), sk \leftarrow s)$

Algorithm 2 Kyber.CPA.Enc$(pk = (t, \rho), m \in M)$: encryption

1: $r \leftarrow \{0,1\}^{256}$
2: $t \leftarrow \text{Decompress}_q\ (t, d_t)$
3: $A \leftarrow R_q^{kk} := \text{Sam}(\rho)$
4: $(r, e_1, e_2) \sim \beta_\eta^k \beta_\eta^k \beta_\eta$
5: $u \leftarrow \text{Compress}_q\ (A^T r + e_1, d_u)$
6: $v \leftarrow \text{Compress}_q\ (t^T r + e_2 + \lfloor \frac{q}{2} \rceil m, d_v)$
7: **return** $c = (u, v)$

Algorithm 3 Kyber.CPA.Dec$(sk = s, c = (u, v))$: decryption

1: $u := \text{Decompress}(u, d_u)$
2: $u := \text{Decompress}(v, d_v)$
3: **return** $\text{Compress}_q(v - s^\intercal u, 1)$

Algorithm 1 is the Key Generation for Kyber.CPA. It starts by generating two random 256-bit strings, ρ, and σ. Then, it generates a matrix A from the ring R_q^{kk} using the function Sam(ρ). Next, it generates two vectors (s, e) from the distribution $\beta_{\eta k}\beta_{\eta k}$ using the function Sam(σ). The public key pk is then computed as the tuple (t, ρ), where t is the result of the function $\text{Compress}_q(As + e, d_t)$, and the secret key sk is s. Algorithm 2 is the encryption algorithm Kyber.CPA.Enc. It takes as input the public key $pk = (t, \rho)$ and a message $m \in M$. It starts by generating a random 256-bit string r. Then, it decompresses t and generates the matrix A using the function Sam(ρ). Next, it generates three vectors (r, e_1, e_2) from the distribution $\beta_\eta^k \beta_\eta^k \beta_\eta$. The ciphertext c is then computed as the tuple (u, v), where u is the result of the function $\text{Compress}_q(A^T r + e_1, d_u)$ and v is the result of the function $\text{Compress}_q(t^T r + e_2 + \lfloor \frac{q}{2} \rceil m, d_v)$. Algorithm 3 is the decryption algorithm Kyber.CPA.Dec. It takes as input the secret key $sk = s$ and the ciphertext $c = (u, v)$. It starts by decompressing u and v. Then, it returns the result of the function $\text{Compress}_q(v - s^\intercal u, 1)$.

The CCA-secure KEM

The CCA-secure Key Encapsulation Mechanism (KEM) is a cryptographic protocol that provides confidentiality and integrity for data transmission. The Fujisaki-Okamoto transformation is used to convert an IND-CPA [46] secure public-key encryption scheme into a CCA-secure KEM. This transformation allows the sender to produce an encrypted message and a key encapsulation like in the algorithm 4, which the receiver can then use to derive a shared secret key. This shared key can be used to encrypt and decrypt data, as explained in 5th algorithm. The CCA security ensures that the attacker cannot manipulate the ciphertext to extract the secret key.

Algorithm 4 Kyber.Encaps($pk = (t, \rho)$)

$m \leftarrow \{0,1\}^{256}$
$(\hat{K}, r) := G(H(pk), m)$
$(u, v) :=$ Kyber.CPA.Enc$((t, \rho), m; r)$
$c := (u, v)$
$K := H(\hat{K}, H(c))$
return (c, K)

Algorithm 5 Kyber.Decaps($sk = (s, z, t, \rho)$, $c = (u, v)$)

$m\prime :=$ Kyber.CPA.Dec$(s, (u, v))$
$(\hat{K}, r) := G(H(pk), m)$
$(u\prime, v\prime) :=$ Kyber.CPA.Enc$((t, \rho), m, r)$
if $(u\prime, v\prime) = (u, v)$ **then**
  return $K := H(\hat{K}, H(c))$
else
  return $K := H(z, H(c))$
end if

Algorithm 4 is the encapsulation algorithm Kyber.Encaps. It takes as input the public key $pk = (t, \rho)$. It starts by generating a random 256-bit string m. Then, it computes $(\hat{K}, r)$ using the function $G(H(pk), m)$. Next, it computes (u, v) using the encryption function Kyber.CPA.Enc with inputs $(t, \rho), m$ and r. The ciphertext c is then computed as the tuple (u, v). Finally, it computes the key K using the function $H(\hat{K}, H(c))$ and returns the tuple (c, K). Algorithm 5 is the decapsulation algorithm Kyber.Decaps. It takes as input the secret key $sk = (s, z, t, \rho)$ and the ciphertext $c = (u, v)$. It starts by decrypting $m\prime$ using the decryption function Kyber.CPA.Dec with inputs s and (u, v).

Then, it computes $(\hat{K}, r)$ using the function $G(H(pk), m)$. Next, it computes $(u\prime, v\prime)$ using the encryption function Kyber.CPA.Enc with inputs $(t, \rho), m\prime$ and r. If $(u\prime, v\prime)$ equals (u, v), it computes the key K using the function $H(\hat{K}, H(c))$ and returns K. Otherwise, it computes the key K using the function $H(z, H(c))$ and returns K.

3.4 Proposed Scheme

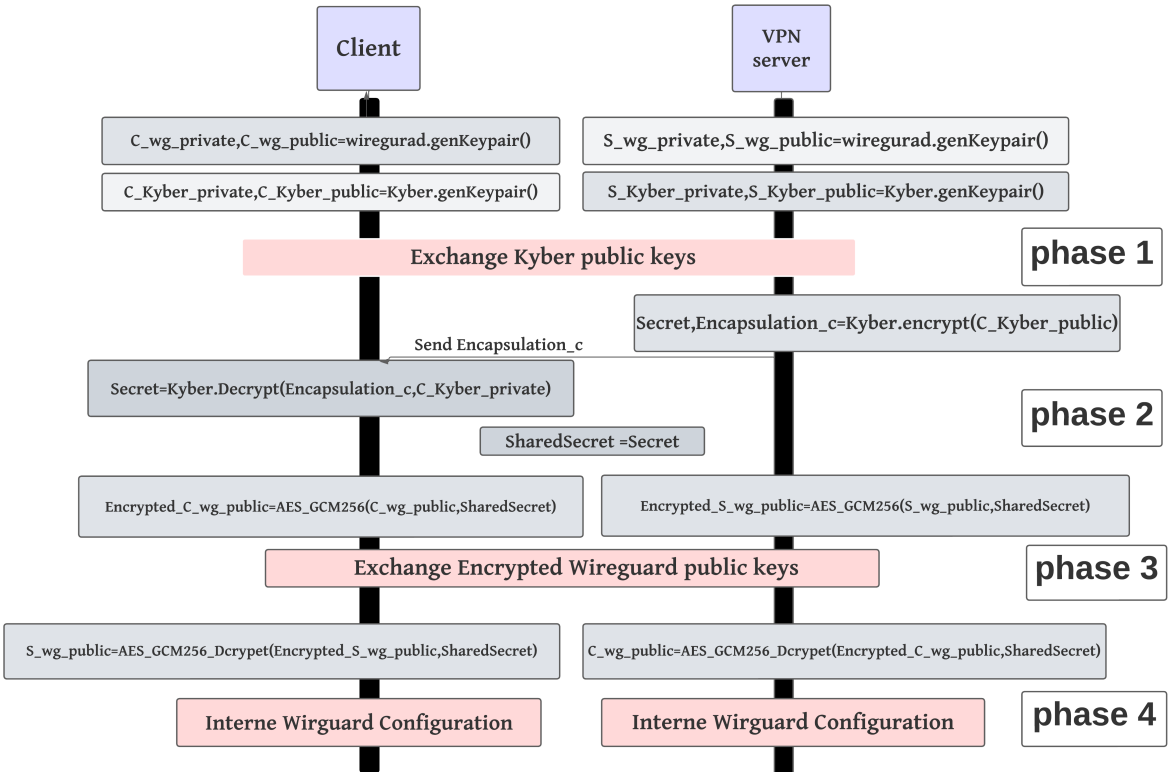

Figure 3.2: HK-WireGuard protocol.

The proposed HK-Wireguard protocol has four phases, as presented in Figure 3.2, and is described as follows:

Phase 1: In the first phase, key pairs are generated for Kyber and WireGuard using the functions $Kyber.genKeypair()$ and $wireguard.genKeypair()$. The Kyber public key exchange takes place between the client and server. The client initiates the communication by sharing its Kyber public key C_kyber_public with the server. The server responds by sharing its Kyber public key S_kyber_public with the client.

Phase 2: In the second phase, the server generates a shared secret and encapsulation using the client's Kyber public key with the function $Kyber.encrypt(C\_kyber\_public)$. The server then shares the encapsulation with the client. The client uses its own Kyber private key $C\_kyber\_private$ and the encapsulation from the server to recover the shared secret.

Phase 3: In the third phase, both the client and server encrypt their own WireGuard public keys using the function $AES\_GCM256(SharedSecret)$. They then exchange the encrypted WireGuard public keys with each other.

Phase 4: In the final phase, both sides decrypt the other's WireGuard public keys using the function $AES\_GCM256(TheOther\prime sPK, SharedSecret)$. Finally, an internal configuration is established based on the shared WireGuard public keys.

3.5 Informal Security analysis

In this section, we will analyze the security of the proposed solution, the HK-Wireguard protocol.

The HK-Wireguard protocol enhances the security of the WireGuard VPN protocol by adding an extra layer of encryption using an encrypted public key based on the Kyber Shared Secret. This feature is post-quantum secure, which means it is resistant to attacks by quantum computers.

The encryption layer ensures that even if a quantum attacker intercepts the public key, the private key, or the shared secret key, it cannot be easily obtained using quantum algorithms such as Shor's algorithm or Grover's algorithm. This increases the security of the protocol and reduces the risk of compromising the symmetric key used for the encryption and decryption of communication.

HK-Wireguard maintains the **forward secrecy** [47] feature of the original WireGuard protocol, which means that past communication remains secure even if the

long-term private key is compromised in the future. This ensures that past communication remains confidential and provides an additional layer of protection against potential attacks.

The identity-hiding feature of the original WireGuard protocol is also retained in HK-WireGuard, which helps protect users' privacy by keeping their real IP addresses hidden. This prevents potential adversaries from identifying the source and destination of communication, enhancing the overall privacy of the protocol[47]
.

HK-WireGuard uses constant-time implementations of Kyber and WireGuard to protect against **side-channel attacks**, such as timing attacks, cache attacks, and power analysis attacks. It avoids secret-dependent branches, table lookups, and non-constant-time operations and employs techniques such as Montgomery and Barrett reductions to prevent timing leakage. This ensures that the protocol is resistant to most side-channel attacks.

The protocol is resistant to **Man-in-the-Middle (MITM)** attacks due to the use of public key cryptography in the Kyber encryption scheme and mutual authentication provided by the Noise Protocol. [32] Any attempt by an attacker to intercept the communication between the client and the server and impersonate either party would require compromising the private keys used in the Kyber encryption scheme and/or the authentication keys used in the Noise Protocol. Since the Kyber encryption scheme is post-quantum secure and the Noise Protocol uses strong authentication, the probability of a successful MITM attack is extremely low.

The security analysis of the HK-Wireguard protocol confirms that it provides an enhanced level of security by integrating post-quantum cryptography into the WireGuard VPN protocol while maintaining the strong security features and performance efficiency of the original protocol. It offers protection against potential quantum attacks, ensures the confidentiality, integrity, and privacy of communication, and provides forward secrecy and identity-hiding capabilities.

However, proper implementation, configuration, and ongoing security evaluations are

crucial to ensuring the effectiveness and security of the HK-Wireguard solution in protecting sensitive communications.

3.6 Conclusion

The HK-Wireguard scheme presents a promising combination of the lightweight and fast WireGuard protocol with the post-quantum Kyber cryptographic algorithm. It offers a compelling balance between performance and security for VPN applications. By leveraging the efficiency of WireGuard and the strong security of Kyber, HK-Wireguard addresses the challenges posed by quantum computing. Future work can focus on further optimizing the scheme and exploring its potential in various use cases.

CHAPTER 4

IMPLEMENTATION AND RESULTS

4.1 Introduction

Chapter 4 presents the implementation phase of the research study, which involves putting into action the design developed in Chapter 3. The main objectives of this phase are to develop the software application based on the WireGuard protocol and test its performance in a real-world setting. In this chapter, we will describe the implementation details of the HK-Wireguard Client application.

4.2 Implementation

4.2.1 Environment

Development The project was developed on an Ubuntu 20.04.2 AMD64 system with kernel version 5.15.0-60-generic. The server used had 8GB of RAM, an Intel Core i7-8550U CPU @ 1.80GHz with 4 cores, and an Ethernet interface of 10 Mbit/s. The client was a KVM machine with 2GB of RAM, a Skylake-Client-IBRS CPU at 2GHz with one core, and an Ethernet interface.

- **WireGuard:** is a new, modern, and secure VPN protocol that is quickly gaining popularity. It is designed to be faster, simpler, and more efficient than traditional VPN protocols such as IPsec and OpenVPN. WireGuard is also much lighter, making it a good choice for devices with limited resources.
- **Kyber AVX2 implementation:** The Kyber AVX2 implementation is based on the Kyber reference implementation, but it has been improved to take advantage of the AVX2 instruction set addition. This includes leveraging AVX2 intrinsics to conduct operations on data vectors as well as AVX2-specific instructions to execute operations on polynomials.
- **KVM:** Kernel-based virtual machines are full virtualization technologies for Linux on x86 hardware. KVM uses the Linux kernel as a hypervisor to allow multiple operating systems to run on the same physical machine. KVM is a popular choice for virtualization because it is free and open source, and it offers good performance and scalability.

- **DigitalOcean VPS hosting:** We used DigitalOcean to host our VPN server in the cloud. DigitalOcean is a popular cloud hosting provider that offers scalable and affordable virtual private servers (VPS) with high-performance SSD storage and global data center locations.
- **NodeJs:** It's an open-source runtime environment that allows developers to run JavaScript on the server side. It uses the V8 engine from Google Chrome to execute JavaScript code and provides access to various modules for building web applications.
- **ElectronJs:** It's a framework for building cross-platform desktop applications using web technologies (HTML, CSS, and JavaScript). It is based on Chromium and Node.js, and it allows developers to create applications that look and feel native on Windows, macOS, and Linux.

4.2.2 Implemantation phases

Phase 1

```
#include <stdlib.h>
#include <stddef.h>
#include <stdio.h>
#include <string.h>
#include "kem.h"
#include "randombytes.h"
int main(void)
{ uint8_t pk[CRYPTO_PUBLICKEYBYTES];
  uint8_t sk[CRYPTO_SECRETKEYBYTES];
  uint8_t ct[CRYPTO_CIPHERTEXTBYTES]; size_t i;
  crypto_kem_keypair(pk, sk);
  for (i = 0; i < KYBER_PUBLICKEYBYTES; i++)
  { printf("%02hhx", pk[i]);} printf("@@");
  for (i = 0; i < KYBER_SECRETKEYBYTES; i++)
  { printf("%02hhx", sk[i]);} return 0;
  }
```

Listing 4.1: Kyber key pair generation

```
exec(‘wg genkey |
     tee privatekey |
      wg pubkey > publickey‘,
( err ? console.log(err) : console.log(stdout))
   );
   //
```

Listing 4.2: Wireguard key pair generation

```
//The client start by
 const client = net.createConnection({ host: "206.189.18.1", port: 8080 },
 async () => {
       // client starts sending the kyber public key to the server
     client.write(Buffer.from(client_publickeyKyber));
   });

//Then the server

 client.on('data', (data) => {
       if (data.length == 800) {
         // Put the client kyber public key in the variable
         kyberClientPublicKey = data;
         //Send the server kyber public key to the client
         client.write(Buffer.from(ServerKyberPublicKey));
```

Listing 4.3: Exchange kyber public Keys

Phase 2

```
      // server side
        // generate the encapsulation and the shared secret
  [encapsulation_c, secret_key] =
handleEncapsulation_and_SecretKey(kyberClientPublicKey);
    //Put the shared secret in the variable
         SharedSecret = secret_key;
      // client side
          //If the data length is 768 then it is the encapsulation
      if (data.length == 768) {
      Encapsulation_C = data; // put the encapsulation in the variable
       /* generate the shared secret using the encapsulation and
       the client's private key*/
      SharedSecret =
      kyber.Decrypt512(Encapsulation_C, client_privatekeyKyber);
```

Listing 4.4: Generate and Exchange the Encapsulation

Phase 3

```
//the Client side
   // encrypt the Client wg public key using the shared secret
    const encryptedClientWgPublicKey =
    AES_GCM.encryptAES_GCM(ClientWgPublicKey, SharedSecret);
    //  put the encrypted data, iv, and tag in one buffer
   const encryptedClientWgPublicKeytoSend =
    Buffer.concat([encryptedClientWgPublicKey.encrypted,
    Buffer.from("|:"),
       encryptedClientWgPublicKey.iv, Buffer.from("|:"),
       encryptedClientWgPublicKey.tag]);
       //send encrypted Wireguard public key to Server
       client.write(Buffer.from(encryptedClientWgPublicKeytoSend));
//the Server side
   //If the data length is 80 then it is the encrypted wg public key
   if (data.length == 80) { const [encrypted, iv, tag] = Bufferspliter(data);
         // encrypt the wg public key using the shared secret
         const encryptedServerWgPublicKey =
         AES_GCM.encryptAES_GCM(ServerWgPublicKey, SharedSecret);
```

```
        //  put the encrypted data, iv, and tag in one buffer
    const encryptedServerWgPublicKeytoSend =
Buffer.concat([encryptedServerWgPublicKey.encrypted,Buffer.from("|:"),
  encryptedServerWgPublicKey.iv, Buffer.from("|:"),
      encryptedServerWgPublicKey.tag])
        //send encrypted Wireguard public key to Client
        client.write(Buffer.from(encryptedServerWgPublicKeytoSend));
```

Listing 4.5: Encrypt and Exchange WireGuard public keys

Phase 4

```
// the Server side

   // decrypt the wg public key using the shared secret
ClientWgPublicKey =
AES_GCM.decryptAES_GCM(encrypted, SharedSecret, iv, tag);
configureWireguardServer(ServerWgPrivateKey,
'10.160.0.1/24', //server address range
'52533', // the port
'PqWg1', //the wireguard interface name
'wlp2s0',//the server Connected Interface name
ClientWgPublicKey)

// the Client side
   if (data.length == 80) {
       const [encrypted, iv, tag] = Bufferspliter(data);
       // decrypt the wg public key using the shared secret
       const ServerWgPublicKey =
       AES_GCM.decryptAES_GCM(encrypted, SharedSecret, iv, tag);
configureWireguardClient(ClientWgPrivateKey,
ServerWgPublicKey,
"206.189.18.1:52533",//example of server endpoint
"52533",//clientListenPort example
"0.0.0.0/0");// allowed IPs example
```

Listing 4.6: Decrypt Wireguard public key and intern configuration

4.2.3 HK-Wireguard Application

Our HK-Wireguard client app 4.1 is an open-source VPN client application that implements the HK-Wireguard protocol. The application interface is developed using ElectronJS, which allows for cross-platform compatibility with Windows, MacOS, and Linux operating systems. The application is available for download on the official HK-Wireguard GitHub repository at HK-Wireguard [1].

As shown in Figure 4.1, the HK-Wireguard client app has a simple interface that displays the status of the VPN connection. Users can also switch between different HK-Wireguard server configurations and enable or disable the VPN connection with just one click.

HK-Wireguard provides users with a secure and easy-to-use VPN solution. It offers better security than the original WireGuard due to its post-quantum cryptography, while its performance is similar to that of WireGuard. Furthermore, HK-Wireguard is cross-platform compatible and provides users with an intuitive and user-friendly interface.

[1] https://github.com/QuntumWorld/HK-Wireguard

Figure 4.1: HK-Wireguard Client App

4.2.4 HK-Server

HK-Server [2] is a Vpn server application that allows HK-Wireguard clients to securely connect to the server from anywhere in the world.

Deployment Steps

For deployment, we used the host machine as the VPN client and the DigitalOcean VPS machine as the VPN server. The VPN server was hosted on a DigitalOcean VPS with 1GB of RAM, 1 vCPU, and 25GB of disk space, running Ubuntu 22.10 x64 with a public IPv4 address.

[2] https://github.com/QuntumWorld/HK_Server

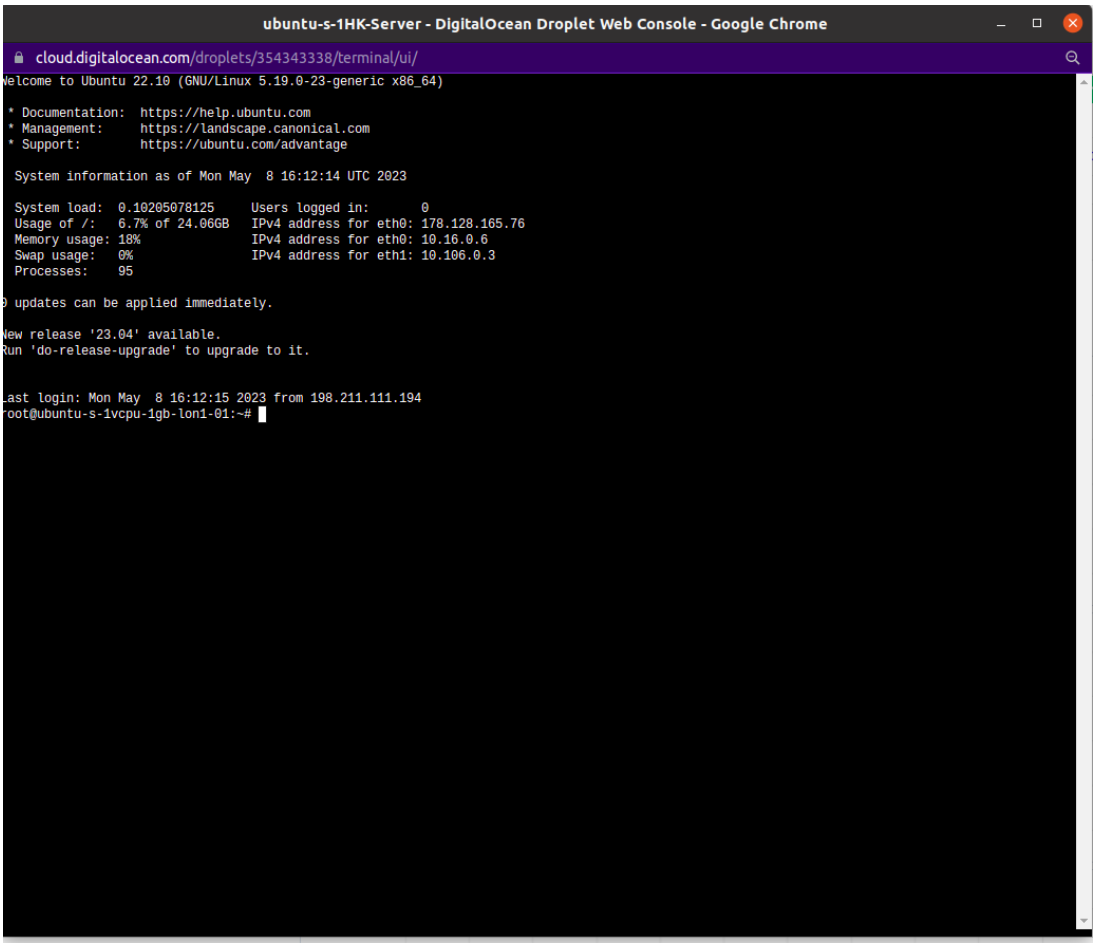

Figure 4.2: droplet web console

1. Creating the DigitalOcean VPS droplet. like in Figure 4.3 to get in the droplet provided web console 4.2 and then

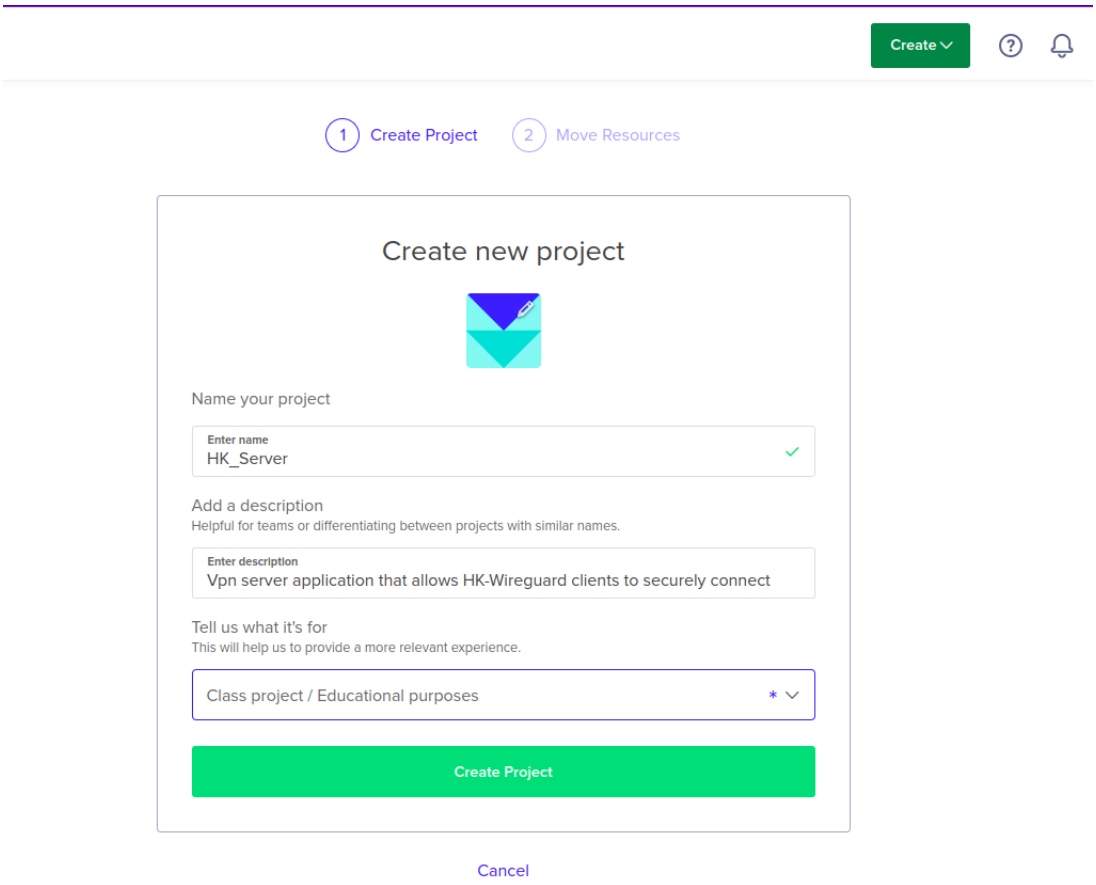

Figure 4.3: Creating the HK_ServerVPS droplet

2. Updating the packages using the following commands:

```
sudo apt update
sudo apt-get install Node.js
sudo apt install npm
```

3. Installing WireGuard tools using the following command:

```
apt install wireguard-tools
```

4. Cloning the HK_Server our repository using the following command:

```
git clone https://github.com/QuntumWorld/HK_Server
```

5. Enabling the firewall by uncommenting the following lines in the `server.js` code:

```
/*
exec(‘sudo ufw allow ${serverListenPort}/udp‘);
exec("sudo ufw allow ssh");
exec("sudo ufw allow http");
exec("sudo ufw enable");
*/
```

6. Installing the sha3 package using the following command:

```
npm i sha3
```

7. We Run the server using the following command:

```
sudo node server
```

After running the server we see the message "VPN server listening on port 8080!" if the server has started successfully.

8. On the client side, we configure the HK_Wireguard client app using the IPv4 address of the droplet, and run the app using the following command: and click on the connect button.

```
npm start
```

9. On the server side, we should see the message "VPN client connected! WireGuard server configured on PqWg1" if the client has connected successfully.

10. On the client side, we can check the connection status using the following command:

```
sudo wg
```

we should see output similar to the following:

```
interface: PqWg1
  public key: pSJnnE4Kut0EEWcBCBfg8JHwq4lsOOOgswuZ8a7SiHg=
  private key: (hidden)
  listening port: 52533
  fwmark: 0xca6c

peer: Tyqeh4cvUlsS/FZJqGjTuzqkU5Y36xcCR3ySEI+E6HM=
  endpoint: 178.128.165.76:52533 # this droplet ipv4
  allowed ips: 0.0.0.0/0
  latest handshake: 42 seconds ago
  transfer: 252 B received, 53.55 KiB sent
```

Listing 4.7: Client Wireguard configuration

11. On the Server side, we should see the following:

```
interface: PqWg1
  public key: Tyqeh4cvUlsS/FZJqGjTuzqkU5Y36xcCR3ySEI+E6HM=
  private key: (hidden)
  listening port: 52533

peer: pSJnnE4Kut0EEWcBCBfg8JHwq4lsOOOgswuZ8a7SiHg=
  endpoint: 105.235.134.162:20254 # the client machine ipv4
  allowed ips: 10.106.0.0/24
  latest handshake: 32 seconds ago
  transfer: 109.10 KiB received, 600 B sent
  persistent keepalive: every 25 seconds
```

Listing 4.8: Server Wireguard configuration

4.3 Results and Comparison

4.3.1 Comparison between Protocols

We provide a comparison between several protocols commonly used in VPNs,as shown in Figure 4.4, focusing on the performance of our proposed HK-Wireguard compared to the original WireGuard and OpenVPN linked to the OpenSSL library.

To conduct our comparison, we utilized the original C implementation of the protocols and measured their performance using the C standard library, time.h. Additionally, we employed the Kyber AVX2 implementation of the Kyber cryptographic algorithm, leveraging the AVX2 instruction set found in recent Intel processors to achieve higher performance.

We investigated four specific VPN protocols in terms of computational cost, including key generation time, configuration time, and the total time required to establish a connection.

Table 4.1 presents the sizes of the public key (Pk Size) and secret key (sk Size) for various cryptographic schemes, each with a security level equivalent to AES-128. These schemes are employed by different VPNs. For instance, OpenVPN can use RSA 3072 or ECC 256, HK-Wireguard uses Kyber 512, and WireGuard uses Curve 25519.

Version	Security Level	sk Size	Pk Size
Kyber512	AES128	1632	800
RSA3072	AES128	384	384
ECC256	AES128	32	64
Curve25519	AES128	32	32

Table 4.1: Key sizes and ciphertext size of studied schemes [1]

The selection of these protocols was based on their security level, which is a crucial factor in ensuring the privacy and safety of data transmission.

Table 4.2 displays the computational cost of each protocol. The original WireGuard implementation exhibits the quickest key generation and overall time, closely followed by our proposed HK-WireGuard solution. On the other hand, OpenVPN P-256 and OpenVPN RSA 2048 have much slower key generation speeds, making them less efficient for establishing VPN connections quickly.

Table 4.2: Computational Cost of VPN Protocol

Protocol	Key gen (ms)		Config (ms)		Total (ms)
	Server	Client	Server	Client	
Original WireGuard	4.88	5.08	8.95	14.56	24.52
HK-Wireguard	4.90	5.10	8.95	14.56	24.56
OpenVPN RSA 2048	185709.46	18	1.72	3.79	185731.25
OpenVPN ECC 256	200207.40	67	3.12	7.17	200281.57

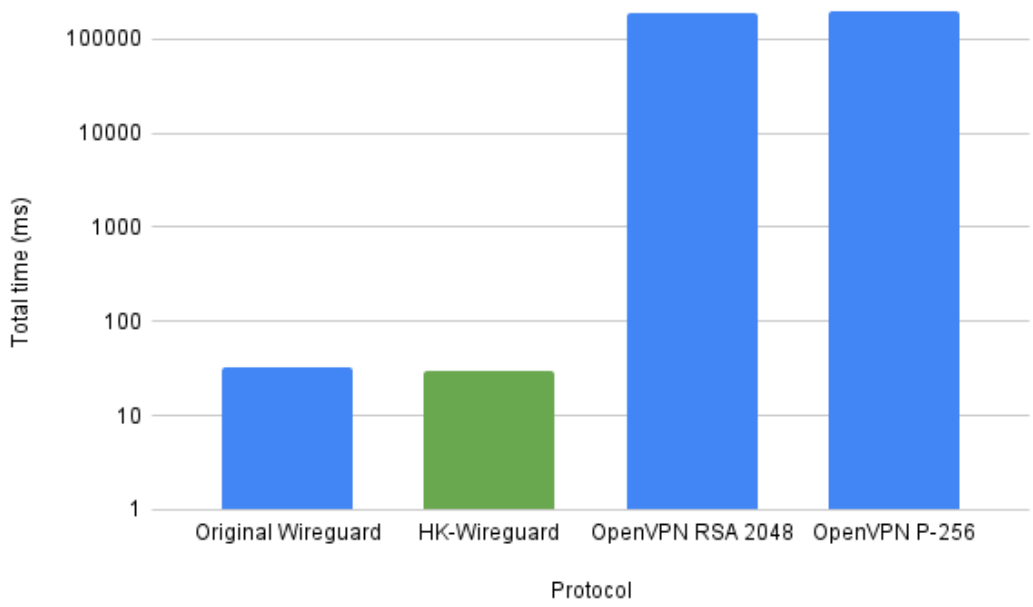

Figure 4.4: Computational Cost

In terms of security, the HK-Wireguard solution combines the high-performance Wireguard protocol with the post-quantum Kyber cryptographic algorithm, which provides great resistance to quantum computer threats. However, it is important to note that the key generation time for OpenVPN RSA 2048 and OpenVPN ECC 256 is significantly slower compared to the WireGuard-based protocols.

Additionally, we investigated the size of exchanged data for each protocol, as shown in Table 4.3. These values represent the total size of the exchanged data between the server and the client during the VPN connection setup.

protocol	exchanged data Size(bit)
Wireguard	88
HK-Wireguard(kyber512)	2528
OpenVPN RSA 2048	19880
OpenVPN P-256	18352

Table 4.3: exchanged data Size

From the comparison, it is evident that the HK-Wireguard solution provides a good balance between performance and security, offering efficient key generation and a reasonable amount of exchanged data during the connection setup. OpenVPN RSA 2048

and OpenVPN P-256, while slower in terms of key generation, still provide viable options with their own security characteristics.

4.4 Conclusion

In this chapter, we have presented the implementation of HK-Wireguard, a post-quantum VPN protocol that is based on the WireGuard protocol. We have discussed the design of HK-Wireguard, its security features, and its performance. We have also presented the HK-Wireguard client app and the HK-Wireguard server.

GENERAL CONCLUSION

The research study aims to enhance the security of the WireGuard VPN protocol by integrating post-quantum cryptography using the Kyber encryption scheme. The proposed solution, HK-WireGuard, maintains the security features and performance efficiency of the original protocol while providing protection against potential quantum attacks and ensuring the confidentiality, integrity, and privacy of communication.
The implementation details of the HK-WireGuard Client application were described, including the generation of Kyber and WireGuard key pairs, the exchange of Kyber public keys between the client and server, and the testing of the application's performance in a real-world setting.
Overall, the research study presents a promising solution to the security challenges faced by traditional VPN protocols, and it could potentially benefit users who require high levels of security and privacy in their communications. However, ongoing security evaluations and proper implementation and configuration are crucial to ensuring the effectiveness and security of the HK-Wireguard solution.
Limitations and Future Work The HK-Wireguard client app is currently only available for Ubuntu, limiting its compatibility with other operating systems. Future work could focus on expanding HK-Wireguard's compatibility with other operating systems,(such as Windows and MacOS) and devices(smartphones, tablets, and others).
We believe that HK-Wireguard is a promising post-quantum VPN protocol that offers strong security and good performance. We are currently working on improving the

security and performance of HK-Wireguard, and we plan to release a public beta version of HK-Wireguard in the near future. Furthermore, we aim to integrate this protocol into the WireGuard Android app, thereby expanding its reach. We hope that HK-Wireguard will serve as a valuable tool for users seeking a secure and reliable VPN solution.

Bibliography

[1] National Institute of Standards and Technology (NIST). Call for proposals for post-quantum cryptography standardization, 2016.

[2] Burak Ozfidan and Kemal Bicakci. A survey on post-quantum cryptography for secure communications in the iot-wsn environment. *Electronics*, 7(12):370, 2018.

[3] Zornitsa Yakova. A new virtual private networks access model. In *Proceedings of the 2013 International Conference on Computer Science and Applications*, 2013.

[4] Sebastian Rosu, George Dragoi, and Luminita Roşu. Virtual enterprise networks solutions to support the virtual teams work. *Annals of the University of Oradea: Economic Science*, 19(1):1227–1232, 2010.

[5] Shaneel Narayan, Samad S Kolahi, Kris Brooking, and Simon de Vere. Performance evaluation of virtual private network protocols in windows 2003 environment. In *2008 International Conference on Advanced Computer Theory and Engineering*, pages 69–73. IEEE, 2008.

[6] Peter Wu. Analysis of the wireguard protocol. *Master's Thesis, Analysis of the WireGuard protocol, Eindhoven University of Technology*, 2019.

[7] National Institute of Standards and Technology. Post-quantum cryptography standardization, 2022.

[8] Andrew Rukhin, Juan Soto, James Nechvatal, Miles Smid, Elaine Barker, Stefan Leigh, Mark Levenson, Mark Vangel, David Banks, Alan Heckert, and James Dray. NIST special publication 800-22: A statistical test suite for random and pseudorandom number generators for cryptographic applications, 2010.

[9] Mike Ounsworth. PKI Consortium – PQC Conference. [PDF], March 3 2023.

[10] FIPS Pub. Data encryption standard (des). *FIPS PUB*, pages 46–3, 1999.

[11] Prakash Kuppuswamy and Saeed Q. Y. Al-Khalidi. Hybrid encryption/decryption technique using new public key and symmetric key algorithm. *Int. J. Inf. Comput. Secur.*, 6:372–382, 2014.

[12] Nicolas Gisin, Grégoire Ribordy, Wolfgang Tittel, and Hugo Zbinden. Quantum cryptography. *Reviews of modern physics*, 74(1):145, 2002.

[13] Mrs Pooja and Mamta Yadav. Digital signature. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, 3(6):71–75, 2018.

[14] Michel Abdalla and David Pointcheval. Simple password-based encrypted key exchange protocols. In *Topics in Cryptology–CT-RSA 2005: The Cryptographers' Track at the RSA Conference 2005, San Francisco, CA, USA, February 14-18, 2005. Proceedings*, pages 191–208. Springer, 2005.

[15] Ritik Bavdekar, Eashan Jayant Chopde, Ashutosh Bhatia, Kamlesh Tiwari, Sandeep Joshua Daniel, et al. Post quantum cryptography: Techniques, challenges, standardization, and directions for future research. *arXiv preprint arXiv:2202.02826*, 2022.

[16] Daniele Micciancio and Oded Regev. Lattice-based cryptography. *Lecture Notes in Computer Science*, 5203:2–2, 2008.

[17] Vikas Srivastava, Anubhab Baksi, and Sumit Kumar Debnath. An overview of hash based signatures. Cryptology ePrint Archive, Paper 2023/411, 2023.

[18] Jintai Ding and Bo-Yin Yang. Multivariate public key cryptography. *International Journal of Applied Cryptography*, 5(2):87–105, 2011.

[19] Christopher Wolf. Multivariate quadratic polynomials in public key cryptography. *Cryptology ePrint Archive*, 2005.

[20] Gregg Jaeger. *Classical and quantum computing*, pages 203–217. Springer, 2007.

[21] Matthew Edward Briggs. *An Introduction to the General Number Field Sieve*. PhD thesis, Virginia Polytechnic Institute and State University, 1998.

[22] Peter W. Shor. Algorithms for quantum computation: Discrete logarithms and factoring. *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134, 1994.

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

[24] Lily Chen, Lily Chen, Stephen Jordan, Yi-Kai Liu, Dustin Moody, Rene Peralta, Ray A Perlner, and Daniel Smith-Tone. *Report on post-quantum cryptography*, volume 12. US Department of Commerce, National Institute of Standards and Technology, 2016.

[25] https://www.linkedin.com/advice/0/how-does-classical-cryptography-compare-quantum-terms. Accessed on 09/05/2023.

[26] https://www.nccoe.nist.gov/sites/default/files/2022-07/pqc-migration-project-description-final.pdf. Accessed on 09/05/2023.

[27] Paul Ferguson and Geoff Huston. What is a vpn? Rfc 2764, Internet Engineering Task Force, 1998.

[28] Kuwar Kuldeep VV Singh and Himanshu Gupta. A new approach for the security of vpn. In *Proceedings of the Second International conference on Information and Communication Technology for Competitive Strategies*, pages 1–5, 2016.

[29] Dnyanesh Deshmukh and Brijesh Iyer. Design of ipsec virtual private network for remote access. In *2017 International Conference on Computing, Communication, and Automation (ICCCA)*, pages 716–719. IEEE, 2017.

[30] Si Thu Aung and Thandar Thein. Comparative analysis of site-to-site layer 2 virtual private networks. In *2020 IEEE Conference on Computer Applications (ICCA)*, pages 1–5. IEEE, 2020.

[31] Matthew Finlayson, Jon Harrison, and Richard Sugarman. Vpn technologies: A comparison. *Data Connection Limited*, February 2003.

[32] T. Perrin, M. Thomson, and C. Wood. The noise protocol framework. Online, 2017.

[33] Adam Langley. Chacha20 and poly1305 for ietf protocols, 2015.

[34] Jason A Donenfeld. Wireguard: Next generation kernel network tunnel. In *NDSS*, pages 1–12, 2017.

[35] https://openvpn.net/. Accessed on 16/04/2023.

[36] Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Cédric Fournet, Aseem Gollamudi, Georges Gonthier, Nadim Kobeissi, Nataliia Kulatova, Aseem Rastogi, Thomas Sibut-Pinote, Nikhil Swamy, and Santiago Zanella-Béguelin. Towards quantum-safe vpns and internet. Cryptology ePrint Archive, Paper 2019/1277, 2019.

[37] Simon de Vries. Achieving 128-bit security against quantum attacks in openvpn. Master's thesis, University of Twente, 2016.

[38] Andreas Hülsing, Kai-Chun Ning, Peter Schwabe, Florian Weber, and Philip R. Zimmermann. Post-quantum wireguard. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 304–321, 2021.

[39] Scott Fluhrer, Panos Kampanakis, David McGrew, and Valery Smyslov. Mixing Preshared Keys in the Internet Key Exchange Protocol Version 2 (IKEv2) for Post-quantum Security. RFC 8784, June 2020.

[40] Douglas Stebila. Prototyping post-quantum key exchange. *Second PQC Standardization Conference*, 2020.

[41] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum key exchange for the tls protocol from the ring learning with errors problem. *International Journal of Applied Cryptography*, 6(3):237–279, 2016.

[42] Eric Crockett, Christian Paquin, and Douglas Stebila. Prototyping post-quantum and hybrid key exchange and authentication in tls and ssh. Cryptology ePrint Archive, Paper 2019/858, 2019.

[43] Sebastian Paul, Yulia Kuzovkova, Norman Lahr, and Ruben Niederhagen. Mixed certificate chains for the transition to post-quantum authentication in tls 1.3. Cryptology ePrint Archive, Paper 2021/1447, 2021.

[44] Marcin Niemiec and Petr Machnik. Authentication in virtual private networks based on quantum key distribution methods. *Multimedia Tools and Applications*, 75:10691–10707, 2016.

[45] Elena Dubrova, Kalle Ngo, and Joel Gärtner. Breaking a fifth-order masked implementation of crystals-kyber by copy-paste. Cryptology ePrint Archive, Paper 2022/1713, 2022.

[46] Joppe Bos, Leo Ducas, Eike Kiltz, T Lepoint, Vadim Lyubashevsky, John M. Schanck, Peter Schwabe, Gregor Seiler, and Damien Stehle. Crystals - kyber: A cca-secure module-lattice-based kem. In *2018 IEEE European Symposium on Security and Privacy*, pages 353–367, 2018.

[47] Benjamin Dowling and Kenneth G. Paterson. A cryptographic analysis of the wireguard protocol. Cryptology ePrint Archive, Paper 2018/080, 2018.