Quantum Resistant Authentication Methods for Quantum Key Distribution

A thesis presented for the degree of Master in Quantum Science and Technology by

Paula Alonso Blanco

Supervised by

Dr. Marc Manzano
Dr. David Joseph

Faculty of Physics
SandboxAQ, Universitat de Barcelona and others
July 11, 2022

Abstract

Quantum Key Distribution (QKD) can distribute keys securely, even in the era of quantum computers, only if the classical channel has been authenticated. This master thesis investigates several methods and optimal parameters for authenticating the classical channel as quickly as possible in the QKD protocol BB84. We utilized quantum-resistant signature algorithms for authentication as they withstand attacks from quantum adversaries. We introduce a novel authentication approach, mono-authentication, which comprises authenticating only at the end rather than the traditional multi-authentication after each stage of communication.

We first simulated a simile of what would be performed in classical cryptography to distribute a key, where we ask the QKD for a determined number of security bits. Next, we studied how four different signature algorithms performed in a noisy quantum channel and found the optimal cases for implementing these algorithms. Then, we obtained a frequency of authentications for three payloads. Finally, we used the previous results to calculate the minimum period for each post-quantum algorithm needs for authentication in terms of the key rate. Results show that the mono-authentication style is at least twice faster than the multi-authentication case. We conclude that in noisy channels, the mono-case reduces its cost significantly. Regarding the performance of the signatures, *CRYSTALS-DILITHIUM* is shown to be the fastest overall, and in contrast to the other algorithms, its number of signatures per second fluctuates with the key rate while being consistently low for the others.

Keywords: quantum key distribution, quantum resistant algorithms, authentication, noise

Acknowledgements

I would like to express my gratitude to my supervisors, Marc Manzano and David Joseph, who guided me throughout this project. Thank you also to Carlos Aguilar and James Howe for their assistance, which was crucial to completing this work and provided in-depth knowledge of the topic. Finally, I would also like to thank my friends and family for their unconditional support.

Contents

1 Introduction

Quantum computing is gaining popularity due to its promising ability to outperform classical computers in solving complex computational tasks. For example, Grover's algorithm [Gro96] can search in an unstructured database quadratically faster than the classical counterpart. Another example is Shor's algorithm [Sho97], which can factor large integers exponentially quicker than the best-known classical algorithm, as well as solve the elliptic-curve discrete logarithm problem [PZ04]. Simulating the dynamics of complex systems is another promising application of quantum devices [Llo96].

Recently a significant milestone in this field was achieved when a quantum computational advantage over classical computers was accomplished at executing a task exponentially faster than its classical counterpart [Pre12, AA19]. However, the implementation of Shor's algorithm endangers internet security as modern cryptography relies on the difficulty of finding solutions to both the integer factorization problem and the discrete logarithm of a random elliptic curve element [RSA78, Mil86, Kob87]. Even though there is currently no quantum computer powerful enough to implement this algorithm in a realistic setting, we must prepare for what the future holds.

The scientific community began several paths to tackle this problem when they became aware of the situation. The first one was to open a new branch in cryptography dedicated to developing novel algorithms based on problems that take exponential time to solve, both for classical and quantum computers. These algorithms belong to the named quantum-resistant or Post Quantum Cryptography (PQC) [Ber09, BL17]. There is an alternative mechanism to counter Shor's algorithm from a quantum angle, lest these mathematical problems are solved in the future. This second solution exploits some of the unique properties of quantum mechanics [WZ82] to exchange some bits through an insecure quantum channel. This communication method is known as Quantum Key Distribution (QKD) because we distribute a key between parties [BB84, Eke91].

In classical cryptography, we need to ensure confidentiality, identity authentication, and integrity of the transmitted messages. However, with the existing protocols, QKD can only guarantee the first property. By combining QKD and PQC, it is possible to implement the remaining two properties.

For the authentication of the classical channel, we propose using post-quantum algorithms. Although it has been suggested to use existing authentication methods for QKD, we want to assure the most significant possible quantum resistance. Thus we use algorithms that have not been yet compromised by algorithms deployed in quantum computers. This thesis investigates how to combine PQC and QKD in two different situations and through different approaches. We introduce a novel approach where these signatures, typically performed during the communication, are performed at the end. We begin with an examination of an analogy to what is done in classical cryptography, where one wants to complete a transaction having x bits of security, and the key exchange generates them. After using the previous study to fix some variables, we obtain the results for the realistic experimental setup of QKD, in which there is a continuous flow of bits that the QKD must authenticate continuously. For each algorithm implemented in this work, we determine the minimum authentication period given a specific key rate.

This work is organized as follows: Section 2 provides the fundamental definitions of quantum mechanics and cryptography and the essential concepts of classical cryptography, which will complement the quantum section. Then, we present the methodology in Section 3 before showing the results in Section 4. Finally, the work comes to a close in Section 5, where we discuss the conclusions of the work and prospects.

2 Preliminaries

This chapter lays the groundwork for the results of this thesis. We will first introduce the fundamental concepts of quantum mechanics, which are essential for quantum cryptography, before exploring how cryptography works and how it can make communications more secure. Then, towards the end of the section, we introduce the essential concepts from classical cryptography to perform authentication and explain how we integrate them into the chosen QKD protocol.

2.1 The foundation of quantum cryptography: quantum mechanics

All the information of an isolated quantum mechanical system is completely encoded in its quantum state $|\Psi\rangle = \sum_i^d \psi_i |\psi_i\rangle$, where $\{|\psi_i\rangle\}_i^d$ H is the basis of its Hilbert space with dimension d. If $|\psi\rangle$ and $|\phi\rangle$ are possible pure states of a quantum system, then $|\Psi\rangle = \alpha|\psi\rangle + \beta|\phi\rangle$ only represents a possible physical pure state if $|\alpha|^2 + |\beta|^2 = 1$, where α, β C. The coherent superposition of both pure states resulting from this linearity is known as the principle of superposition. We represent mixed states, in contrast to pure states, with a density matrix, $\rho = \sum p_i |\psi_i\rangle\langle\psi_i|$, that is a probability distribution over a set of pure states.

We will be dealing with more than one physical system later in this work, and we will describe each by their corresponding Hilbert space, H_A and H_B. A linear combination of two possible states of the global system, $|\psi_A\psi_B\rangle$, $|\phi_A\phi_B\rangle$ H, returns a state of the composite system, i.e. $|\Psi\rangle$ H, where $|\Psi\rangle = \alpha\,|\psi_A\psi_B\rangle + \beta\,|\phi_A\phi_B\rangle$. When it is impossible to write a state of the global system as a tensor product of the states of each subsystem, $|\Psi\rangle = |\varphi_A\varphi_B\rangle$, we say that it is entangled. Two entangled subsystems are related to the other, and any modification in one results in an alteration in the other, providing quantum mechanics a uniqueness that classical mechanics lacked. In the following section, we discuss how we can modify quantum states.

2.1.1 Measurements

Quantum channels are the mathematical representation of any possible modification in quantum states. In Appendix A there is a thorough explanation of a quantum channel. In this work, we modify the quantum states with quantum measurements, which are the set quantum channels that follow the following properties. Any measurement of m outcomes on a system whose associated Hilbert space has dimension d can be represented by the so-called Positive Operator Valued Measure (POVM), defined by r positive operators $\{M_i \geq 0\}$, where $i = 1, \ldots, r$, such that $\sum_i M_i = \mathrm{I}$. Each measurement is the sum of the projectors, $P_m = |\psi_m\rangle\langle\psi_m|$, into the m outcomes where $M_i = \sum P_m$.

Measurements destroy the prepared states as they force these states to project into the measurement basis to obtain one of the m possible outputs of the chosen measurement. When the initial preparation of a state is unknown, selecting the wrong measurement will output a random value unrelated to the initial preparation. In Appendix B we prove mathematically why randomness emerges in the outcome when we do not measure an eigenstate of the measurement basis. This randomness will be the fundamental concept behind the confidentiality of quantum communications, as we will see in the following sections.

Qubits are the states we measure in this work, but measurements are not the only modification these qubits undergo. The environment sometimes induces undesired modifications that occur naturally to these quantum states, where we have to deal with open

quantum system dynamics. The interaction of the states with the environment gives rise to the so-called incoherent noise described by quantum channels. In Appendix C we find a complete explanation of what a qubit is.

2.1.2 Qubits in noisy channels

The environment perturbs quantum pure states into mixed states as the states get entangled with it. In a realistic setup these fluctuations with the environment are of striking importance as they introduce incoherent noise. This noise can be modelled through quantum channels in many ways. In this work we consider a random bit flip under a probability p, such that after the measurement of the states, we will obtain the opposite expected outcome knowing the result in beforehand with a probability p. The possible prepared qubits in this work, either in the X or Z basis, are $\rho_0 = |0\quad 0|$, $\rho_1 = |1\quad 1|$, $\rho_+ = |+\quad +|$ and $\rho_- = |-\quad -|$. Then, the quantum channel Λ_{noise} acts on them as

$$\rho_i|_{i=0,1,+,-} \longrightarrow \Lambda_{noise}(\rho_i) = K_0\rho_i K0^\dagger + K_1\rho_i K_1^\dagger = \begin{cases} \Lambda(\rho_0) = (1-p)\,|0\quad + p\,|1 \\ \Lambda(\rho_1) = (1-p)\,|1\quad + p\,|0 \\ \Lambda(\rho_+) = (1-p)\,|+\quad + p\,|- \\ \Lambda(\rho_-) = (1-p)\,|-\quad + p\,|+\quad , \end{cases} \tag{1}$$

where $K_0 = \sqrt{1-p}\cdot \mathrm{I}, K_1 = \sqrt{p}\cdot X$ are the Kraus operators, which are one of the multiple mathematical representations of a quantum channel. In particular, the probability of obtaining the undesired flip when applying those operators in our qubits is $p_1 = Tr[K_1\rho_i K_1^\dagger]$.

After outlining all the fundamental ideas underlying quantum cryptography, we can now introduce what it is.

2.2 Quantum cryptography and Quantum Key Distribution

Cryptography aims to send secure information through an insecure channel where an eavesdropper may try to intercept it. It consists of defending protocols against saboteurs and eavesdroppers looking to take advantage of any leak. To secure this information, we need to encrypt it. It was shown that it is possible to send private information in a completely secure way using a pre-shared secret key $\vec{k}$, which used on the sent message ensures its secrecy. In Appendix D there is a deeper explanation of how these keys are applied to the messages.

However, a problem arises that is to distributing the key securely. Shannon showed the impossibility of generating secure keys using an untrusted classical channel [Sha49]. This problem is known as the *key distribution problem*. However, this limitation does not apply if the channel is quantum, where we use quantum states and operations. Due to the unique properties of quantum mechanics, we know we can send bits, ensuring confidentiality. This concept is encapsulated by the no-cloning theorem [WZ82], which states that quantum states whose initial preparation is unknown cannot be exactly copied. We can find the proof of this theorem in Appendix E. This theorem is connected to the fact that measurements collapse quantum states, leading to a random output if the wrong measurement is chosen, which would have a detectable e□ect on the sent state if an eavesdropper was trying to tamper with the communication. This property gave rise to quantum cryptography, which in fact refers to any cryptographic protocol using quantum resources.

The protocols allowing this key distribution are grouped under Quantum Key Distribution (QKD) protocols. These protocols can be classified on prepare-measurement protocols such as BB84 [BB84], and entanglement based (or prepare-only) protocols such as E91 [Eke91]. In this thesis, we simulate the BB84 protocol, which is explained hereafter.

2.3 The BB84 protocol

The BB84 protocol consists of the following. One can suppose the existence of two users willing to exchange some key $\vec{k}$. In this communication, to exchange the key, they will use two di□erent channels: a quantum and a classical channel. We name the communications that happen through the latter as post-processing. The protocol begins in the quantum setup. To ease the comprehension of the protocol, we give an example with initially 10 bits.

2.3.1 Quantum communication

First, Alice randomly generates a n_1-bit-string, Ψ^A_{dAB}, and prepares the quantum states given by a random basis string of length n_1, that is Ψ^A_{bA}. When a bit is 0, she can either prepare $|0\rangle$ or $|+\rangle$, while when she has 1, she will prepare either $|1\rangle$ or $|-\rangle$, associating the eigenvectors with positive eigenvalues to 0 and the ones with negative eigenvalues to 1. Each bit of information is encoded into photons with di□erent polarisations, representing the preparation of these four states. Then, she sends them through an untrusted noisy quantum channel, which means we assume that an eavesdropper may be trying to tamper with the communication. An example with $n_1 = 10$ can be seen in Table 1.

$n_1 = 10$ random data bits	Ψ^A_{dAB}	$=$	0	1	1	0	0	0	0	1	0	1
$n_1 = 10$ random basis	Ψ^A_{bA}	$=$	Z	X	Z	X	Z	X	Z	Z	X	X
prepares 10 states			$\|0\rangle$	$\|-\rangle$	$\|1\rangle$	$\|+\rangle$	$\|0\rangle$	$\|+\rangle$	$\|0\rangle$	$\|1\rangle$	$\|+\rangle$	$\|-\rangle$

Table 1: Alice randomly generates $n_1 = 10$ bits, Ψ^A_{dAB}, and $n_1 = 10$ basis choices, Ψ^A_{bA}. Then, she prepares quantum states such that when we have a zero we associate the eigenstate with positive outcome and the one with negative outcome otherwise.

Once Alice has sent her quantum states to Bob, he randomly generates a string with random basis choices, Ψ^B_{bB}, and uses it to measure and obtain a string of outcomes $1, -1$. One remark is that dealing with 1 and -1 during communication is complex; it is preferable to work on the binary basis. Therefore, we transform the measurement outcomes to a binary string, Ψ^B_{dAB}, as $1 \dashrightarrow 0$ and $-1 \dashrightarrow 1$. In Table 2, we can observe this process in detail, continuing with the previous example.

$n_1 = 10$ random basis	Ψ^B_{bB}	$=$	Z	X	X	Z	X	X	Z	X	X	Z
measurement outcome			1	-1	-1	-1	-1	1	1	1	1	-1
$n_1 = 10$ data bits	Ψ^B_{dAB}	$=$	0	1	1	1	1	0	0	0	0	1

Table 2: Bob generates $n_1 = 10$ random basis choices, Ψ^B_{bB}, and measures each received qubit in the corresponding basis choice, obtaining a string of $1, -1$. Then he transforms them into a $0, 1$ bit-string to obtain Ψ^B_{dAB}.

Ultimately, Alice and Bob aim to obtain $\Psi^B_{dAB} = \Psi^A_{dAB}$, so they have to continue with the protocol. This is the only step of the protocol in which they communicate through the quantum channel. From this point, they start the post-processing.

2.3.2 Basis sifting: Alice

Bob notifies Alice that he is ready by sending her Alice the basis in which he has measured each bit, Ψ^B_{bB}, and then Alice begins with the process called basis sifting. The process starts with Alice checking which random basis choice does not match Bob's. Alice updates her

bit-string with the ones with the same basis choice and discards the other bits, changing the size of Ψ^A_{dAB} from n_1 to n_2, being $n_1 > n_2$. Then Alice generates another random bit-string, Ψ^A_{indAB}, of length n_2, to decide which bits is she going to make public, Ψ^A_{chkAB}, and which ones is she going to keep secret as a key, Ψ^A_{kAB}. In particular, Ψ^A_{chkAB} is the substring of Ψ^A_{dAB} for which the bits of Ψ^A_{indAB} are 1, and Ψ^A_{kAB} when 0. The size of the sifted key Ψ^A_{kAB} will be n_3, such that $n_1 > n_2 > n_3$. After doing this process, Alice makes public Ψ^A_{bAB}, Ψ^A_{indAB}, Ψ^A_{chkAB}. Notice that Alice is making the basis choice public in this step. This is of special importance since Bob has already measured the sent qubits; an eavesdropper can no longer tamper with the quantum communication at this step. In Table 3 we detail the process with the same example.

$n_1 = 10$ random basis: Bob	Ψ^B_{bB}	=	Z	X	X	Z	X	X	Z	X	X	Z
$n_1 = 10$ random basis: Alice	Ψ^A_{bA}	=	Z	X	Z	X	Z	X	Z	Z	X	X
matching basis			Z	X				X	Z		X	
$n_2 = 5$ random data bits	Ψ^A_{dAB}	=	0	1				0	0		0	
$n_2 = 5$ random data bits	Ψ^A_{indAB}	=	0	0				1	0		1	
$n_3 = 3$ secret bits	Ψ^A_{kAB}	=	0	1					0			
$n_4 = 2$ revealed bits	Ψ^A_{chkAB}	=						0			0	

Table 3: Alice first compares both basis and discards the bits in positions (in red) with outmatching choices. After this process there are $n_2 = 5$ remaining bits, $\Psi^A_{dAB'}$. Then Alice generates another random bit-string, Ψ^A_{indAB}, of the same length, 5. This string is used as a reference to divide $\Psi^A_{dAB'}$ into the final secret key Ψ^A_{kAB} of length $n_3 = 3$ choosing the positions of the bits in Ψ^A_{indAB} which are 0, and into the revealed bit-string Ψ^A_{chkAB}, of length $n_4 = 2$, where 1.

2.3.3 Basis sifting: Bob

Similarly, Bob compares Ψ^A_{bAB} to Ψ^B_{bAB} and discards the bits in his measured bit-string Ψ^B_{dAB} in which the basis choice was di□erent. He then using Ψ^A_{indAB} performs the same procedure as Alice to obtain Ψ^B_{chkAB} and Ψ^B_{kAB}. It is in this step where Bob compares Ψ^B_{chkAB} with Ψ^A_{chkAB} to determine how many bits are matching and then estimate the Quantum Bit Error Rate (QBER) in the key. For this protocol, the maximum allowed of errors is 11% [CRE04], and if higher, there may be an eavesdropper trying to tamper with the communication. In the communication, the QBER is calculated as

$$QBER = \frac{(\Psi^{B0}_{chkAB} - \Psi^{A0}_{chkAB}) + (\Psi^{B1}_{chkAB} - \Psi^{A1}_{chkAB})}{n_4}, \tag{2}$$

where Ψ^{B0}_{chkAB} is the number of 0 bits in Ψ^B_{chkAB}, and similarly for the other cases. The length of the string with the revealed bits is $n_4 = \text{len}\left(\Psi^i_{chkAB}|_{i=\text{A or B}}\right)$.

Finally, Bob sends Alice, which is the value of the $QBER$, and she calculates it as well to check if they have the same values. If the process is successful, QBER $< 11\%$, we obtain the sifted key. The length of the key, n_3, should be at least the size of the message m we want to encode to ensure its security, as explained in Appendix D. Finally, we talk about the QBER in-depth and how this bound of 11% was obtained in Section F.

2.3.4 Error correction and privacy amplification: Alice

Having reached this point, we must remind ourselves that this protocol aims to exchange a secret key between Alice and Bob. The secret key in the previous process is Ψ^A_{kAB} and Ψ^B_{kAB}. However, since we are working in a noisy channel, we expect the communication to

have up to a certain number of incorrect bits, hence having $\Psi^A_{kAB} = \Psi^B_{kAB}$ and of course, we aim to distribute the same key. In the original work of $BB84$, the noise was not considered in the game, so error correction was added later.

After making sure that there was no eavesdropper during the quantum communication, the parties initiate the error correction process to obtain the same string, which mathematically traduces as having the maximal mutual information between Alice and Bob. Error correction can be performed in many ways. In this work, we perform the following. Alice generates the hash of her secret key Ψ^A_{kAB} such that

$$F: \quad F \leftarrow F(\Psi^A_{kAB}) \tag{3}$$

where F is the *hash* or *digest*. Since hash functions are deterministic, we always get the same output for a given input. However, for di□erent inputs, we always obtain di□erent outputs of the same length, so it is impossible to distinguish which was the input. The hash function we implement in this work is $F = SHA$-3 512.

Hereafter, we need to perform privacy amplification as well, which consists of making the key more private. Mathematically this traduces as minimizing the mutual information between Alice and Eve as much as possible. To this end, Alice generates a random permutation P P of length n_3, such that P is the set of permutations of the n_3 elements of Ψ^A_{skAB}. She applies this permutation to her key, and then she hashes the output using the same hash function as we introduced in the error correction step, obtaining $\Psi^A_{skAB} = F(P(\Psi^A_{kAB}))$, being Ψ^A_{skAB} the same final secret key. The scheme of the process is

$$\begin{array}{ll} \text{Alice} & \text{Bob} \\ P \quad P & \xrightarrow{F,P} \quad \Psi^B_{skAB} := F(P(\Psi^B_{kAB})). \end{array}$$

After this point, Alice finished with the protocol. Finally, she sends Bob the necessary information about the error correction and the privacy amplification, that is, the output of the hash of the private key F, the random permutation P, and the hash function she has used F.

2.3.5 Error correction and privacy amplification: Bob

This is the last step of the key exchange. Bob receives the data and begins with the error correction process using F, the hash from Alice. He can calculate the hash of his key as well to obtain $F = F(\Psi^B_{kAB})$. If $F = F$, there are some errors in the key. The error correction consists of Bob trialing bit-flips in his key until the hashes match. For an eavesdropper, they would have to brute-force search a pre-image of the hash, however, which is completely infeasible.

To avoid calculations of long periods, what we perform in the implementation is to limit how many errors we will tolerate in our key. We call it the maximum corrected errors, which is a fundamental parameter in this work. Summing up, the process consists of Bob trying di□erent combinations, first flipping only one of the bits of his key in every position and hashing each option to then compare it to F. If the search is unsuccessful, he repeats the process with two errors. He will perform the brute force search up to a maximum number of errors. If he goes through all the combinations and does not find a matching hash, he will discard this key and restart the QKD process again, aborting the process. For the cases he finds a matching hash, he will correct these errors and generate Ψ^B_{kAB}. Finally, Bob has to perform the privacy amplification as well, to finally obtain the common

secret key $\Psi^B_{skAB} = F(P(\Psi^B_{kAB}))$, where $\Psi^B_{skAB} = \Psi^A_{skAB}$ meaning they have successfully exchanged a secret key using quantum states.

Finally, this thesis aims to explore several ways to authenticate the classical channel using classical cryptography, so we explain what it is in the next section.

2.4 Classical cryptography to improve QKD: Post-Quantum Cryptography

We have seen that the post-processing occurs in the classical channel, and it is public; therefore, the only thing the users need to ensure is that they talk to the right person and that there are no modifications in the public messages. To fix this lack of security in the protocol, we introduce authentication.

There are several approaches to authenticate the classical channel in the QKD protocol. However, some of them rely on having pre-distributed keys in the initial round [KMG+20, FMC10], which is not scalable. Ideally, we would use a Public Key Infrastructure (PKI) in which not all the users need to be directly trusted but trust an authority, the Certificate Authority (CA), which distributes public and private keys associated with each user. These keys are subsequently utilized, together with signature and verification algorithms, to authenticate the communication.

Digital signatures are primitives used to verify the authenticity of what is being transmitted. The signature scheme has three components: first, the key-pair, composed of the public and secret key, which comes from the key generation algorithm, the signing algorithm, which produces the signature; and finally, the verifying algorithm that takes the public key and the message and returns either success or failure. These signatures guarantee *integrity* and *origin*. Signatures ensure *authenticated key exchange* (AKE) when they are valid. Provided that the authentication is not broken during the communication, QKD is information-theoretically secure and thus cannot be cracked by future algorithmic advances, even if the public key signature is later deciphered [MSU12]. So we only need to rely on the Public Key Signature once. One last remark is that when we refer to signatures, we also include the verification process, as explained at the beginning of the paragraph.

As we have seen, digital signatures rely upon the complexity of a mathematical problem which, when combined with the correct public and private keys, is easy but becomes a hard problem otherwise. In this work, we will only use the algorithms relying on mathematical problems that are hard to solve for a crypt-analytic attack by a quantum computer, named post-quantum algorithms.

We can then introduce the mathematical problems that are also hard for quantum computers and the algorithms we will use in this work. All these algorithms are finalists of the National Institute of Standards and Technology (NIST) competition which aims to find the most optimal algorithms to standardize as soon as possible to fight the threat of powerful quantum computers by undergoing many rounds of analysis. The con is that these algorithms are still relying on being hard to break, but future cryptanalysis could undermine their security, which recently happened to one of them.

The first problem is the lattice-based cryptosystems, which are believed to have good all-round performance. Some post-quantum algorithms based on this problem are *CRYSTALS-DILITHIUM* [DKL+18] and *FALCON* [DLP14], and NIST announced they would standardize both on 5 July.

Another problem is multivariate-based cryptography, which is based on the difficulty of solving systems of multivariate polynomial equations. The candidate for this type of problem is *Rainbow* [DS05], but it was broken and is now out of the process as of 5 July. A cryptographic primitive is considered broken when an attack is found to have less than its advertised level of security.

Finally, we have hash-based cryptosystems which offer one-time signature schemes based on hash functions, the security assumptions of one-way functions. The example for this problem is *SPHINCS+*[ABB+15], which NIST announced they would standardize on 5 July.

The last property before finishing this section is that we need to ensure that nobody can break the algorithms used for the authentication for at least the number of bits exchanged. The security level, which in our case is an indicator of the effectiveness of the signature algorithms, embraces this idea. For an exchanged key of n bits, an attacker would have to perform 2^n operations to guess the key. The post-quantum signature algorithms used in this work are *CRYSTALS-DILITHIUM*, *SPHINCS+*, *FALCON* and *Rainbow*. Each of them can ensure different security levels. We will study the security levels for 128, 192, and 256 bits of security. *CRYSTALS-DILITHIUM* and *SPHINCS+* have algorithms for all these security levels while *FALCON* and *Rainbow* only for two. In Table 4 we show the names of each algorithm depending on the security level.

Security bits	DILITHIUM	SPHINCS+	FALCON	Rainbow
128 bits	Dilithium2	sphincs-128f-simple	falcon512	rainbowIIIc-classic
192 bits	Dilithium3	sphincs-192f-simple	-	rainbowVc-classic
256 bits	Dilithium5	sphincs-256f-simple	falcon1024	-

Table 4: This table collects the names of the post-quantum algorithms with respect to its security level.

Having introduced all the essential concepts, we present the methodology and contributions in this work. Next, we will be interested in comparing the performance of the four post-quantum algorithms that have just been introduced. In Appendix H, we find a table with some differences in sizes between the different parameters generated by these algorithms.

3 Methodology and contributions

In this work, we aim to determine the optimal way to authenticate the classical channel in the BB84 protocol using post-quantum signature algorithms. Furthermore, we study specific cases to give a series of recommendations to be used in a BB84 experimental QKD setup. To this end, we simulate in Python the protocol procedure following the steps detailed in Section 2.3. In Appendix G we explain in depth the technical details of this simulation.

First, we begin with the study of a simile of what would be performed in classical cryptography with a key exchange, where we ask the QKD protocol for a specific number of bits to be used later as a symmetric key. Then, to study how these algorithms perform in the authentication of the classical channel in BB84, we first check some variables for the different security levels. Hereafter we examine the variables studied in this work.

Mono- and multi-authentication

The first problem we study is which steps of the protocol are optimal to authenticate. In the only existing authenticated QKD experimental implementation [WZW+21], the signature scheme implemented is to sign and verify in several steps of the protocol [MSU12]. We define multi-authentication as the process in which the authentication is performed in several steps during the communication. In red in Fig. 1, we indicate where exactly these signatures are performed. However, this is not the only way to perform the signatures. We

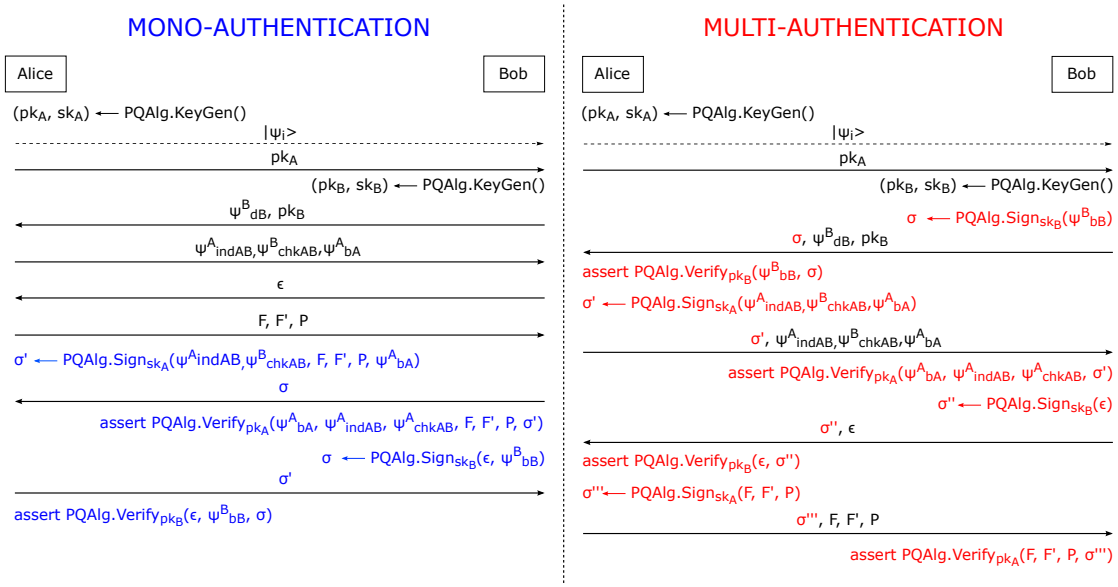


Figure 1: High-level overview of the two signature styles in the BB84 protocol. In red on the right, we detail the multi-authentication scheme [MSU12]. In blue on the left, we propose the mono-authentication style. In continuous black arrows, we indicate what is sent through the classical channel, while the dotted black arrows represent the communication through the quantum one. On top of the diagram, PQAlg.KeyGen() generates the secret sk and public keys pk for both users, Alice and Bob. PQAlg indicates the post-quantum algorithm used for the authentication. Sigmas, σ represent the signatures, which are the output of the PQAlg.Sign$_{sk}$() functions. Then with the PQAlg.Verify$_{pk}$() we determine if the authentication was successful, and the process continues.

propose a mono-authentication style, which implies generating the signature after exchanging the key and ending communication. It consists of Alice signing all the information she has sent to Bob, then sending it to Bob for verification and vice versa. In Fig. 1 we can find a detailed scheme of how this is performed.

The difference between both styles is that four signatures are performed in the multi-case, while for the mono case, we only perform two at the end. Another critical difference is that as the exchanged bit-string goes through a process of error correction where sometimes the key is discarded and the process restarted, some signatures will be discarded. In the case of multi-signature, these signatures are performed uselessly, and the calculation time will be lost, while for the mono-signature case, the only time that counts will be the non-aborted ones.

QBER, maximum number of corrected errors, aborts, and overhead

We refer to the probability p of having a random flip due to the noise in the quantum channel as QBER since they are similar. The QBER indicates an estimation of the error in the quantum channel and p is the probability of having an error. We assume they are the same to make the notation easier. The QBER is one of the most relevant variables in this work. We use as a reference the QBER estimated experimentally in the authenticated QKD implementation [WZW+21]. We study the QBER up to an error of 1.1% as it is the maximum experimental value obtained. We also study the cost of the authentication with respect to the time it takes to execute the protocol, which we name overhead. In some cases, we split the authentication into signature and verification to better compare how each algorithm performs compared to the others. In those cases, we explicitly refer to the

multi/mono-signatures/verifications.

Another parameter is the number of corrected errors. We correct up to 3 errors because a brute force search of 4 takes too much time. In Appendix I, we show how much time it takes to correct the errors at worst. Every time the number of errors is higher than the number of corrected errors, the key is discarded, and we abort that exchange and restart it again. The number of aborts to distribute a specific number of keys is another parameter we study.

After studying all the introduced parameters, we wanted to see how the amount of exchanged keys affected the performance of the algorithms. It consists of a study of a realistic QKD setup with a continuous photon stream. We decide to periodically authenticate the exchanged bits within a fixed time frame given a key rate rather than asking for a specific number of bits at the start of the round.

Small, medium, and large bins

For the case where we have a continuous photon stream in the QKD protocol, we decided to study how the key size affects the signature time for each algorithm. First, we explored how much time it takes every post-quantum algorithm to perform the authentication for a range of key sizes between 0 and 2000 bits. Then, we averaged these results into three different bins. When the range is between 0 and 100 bits, we refer to it as small, between 100 and 500 bits as medium, and between 500 and 2000 bits as large. The aim of grouping the results into three bins is to study how every algorithm performs under situations where the key length needed is smaller, medium, or larger.

Recommendations for different key rates

Finally, we study how a given key rate affects the cost of the authentication. There is a continuous basis sifting process, with a defined rate of the sifted key. The difficulty is knowing when and how often to authenticate and how this varies for each post-quantum algorithm. We calculate the optimal period so that every signing algorithm performs best in this continuous period: the minimum execution time of the signatures. We provide a table with the minimum signature time recommendations for each algorithm, such that whenever there is a given key rate, we know the minimum time we have to wait between signatures.

4 Results

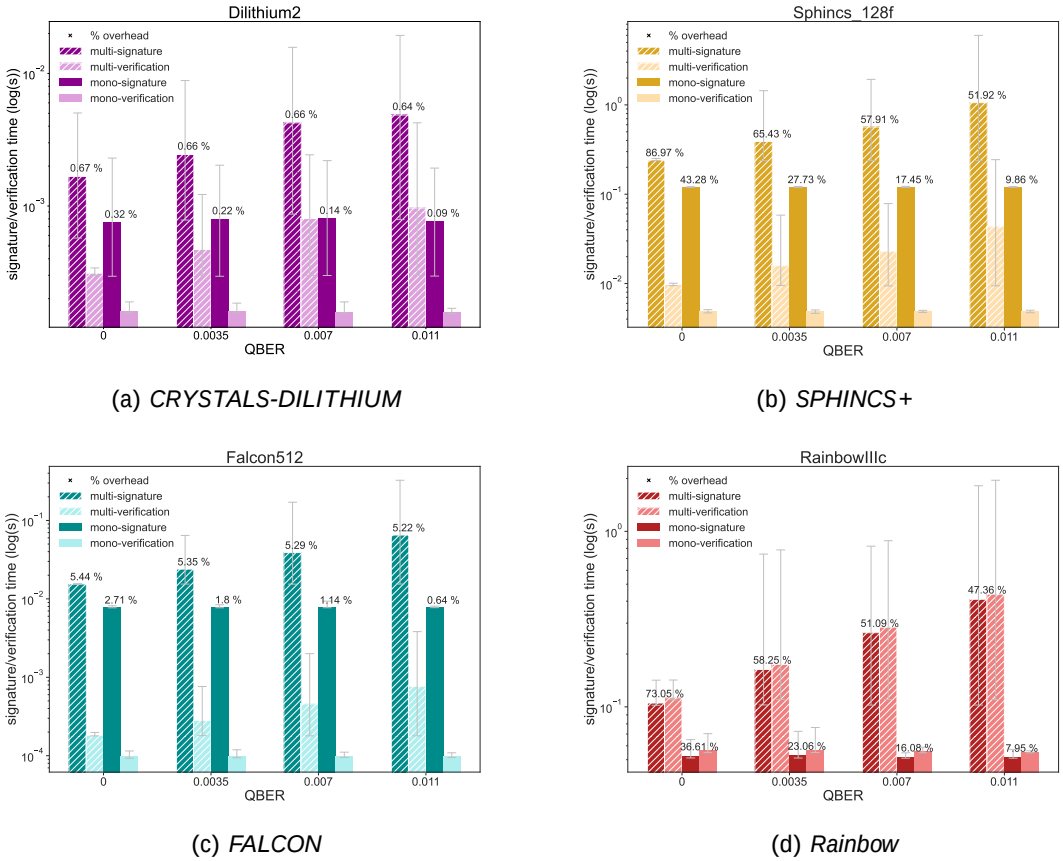
In this section, we show the results for the case where the transmission is per blocks and then as a continuous photon stream. We perform 100 repetitions for each algorithm and represent the deviation as thin grey bars in the plots.

4.1 Photon transmission per blocks: 128, 192 or 256 bits

We test the performance of the different post-quantum algorithms in both ways of authentication under different parameters. We begin studying the dependence of the results on the amount of noise in the quantum channel.

4.1.1 Noise dependence, QBER

The following results are for an exchange of 128 bits of security. In Fig. 2 we observe for every algorithm that the higher the QBER, the higher the dashed bars, which means



(a) *CRYSTALS-DILITHIUM* (b) *SPHINCS+*

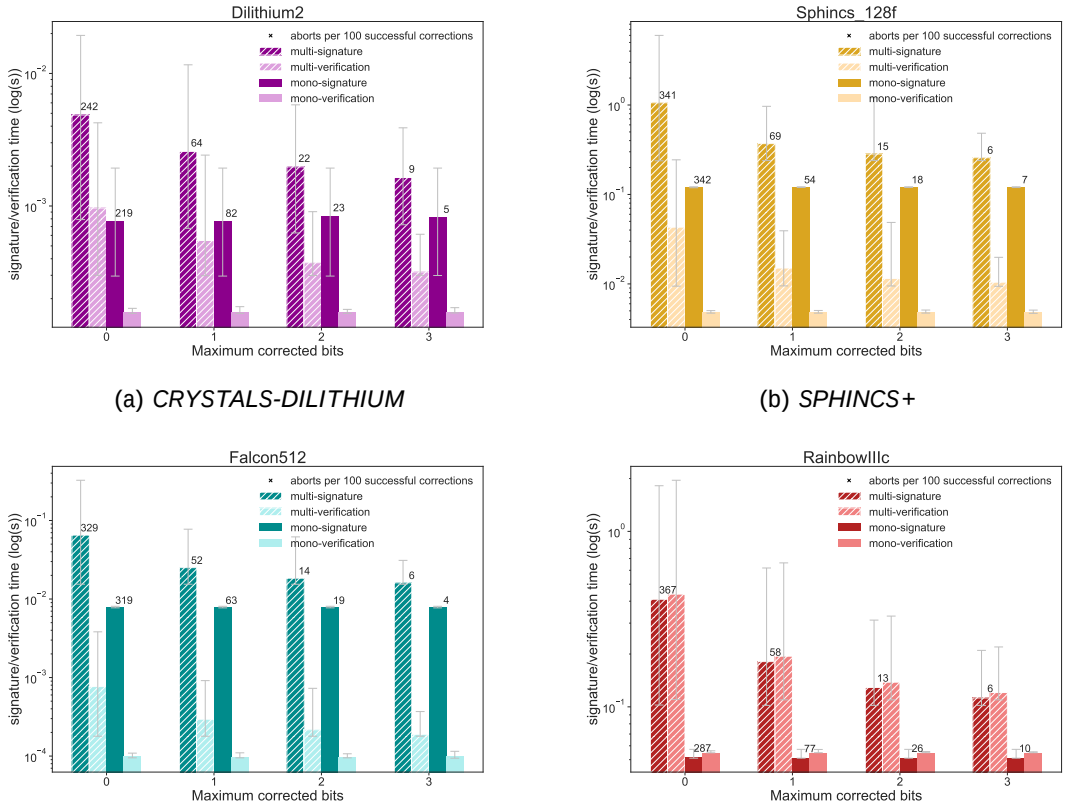
(c) *FALCON* (d) *Rainbow*

Figure 2: Noise dependence in BB84 under multi- and mono-authentication schemes. In each plot, di□erent post quantum algorithm is studied. The the darker bars represent the signature time and the lighter ones the verification. The dashed bars indicate the use of the multi-authentication scheme, while the plain ones represent the mono-authentication case. In the plots the grey line is the deviation of the results, as they are the output of an average of 100 repetitions. The overhead of the authentication with respect to the time of the communication is plotted on top of each pair of bars. Notice that the time axis is plotted in logarithmic scale. To consult the plots in linear scale check Appendix J.1.

more time for multi-authentication, whereas the plain ones remain constant. We observe with *CRYSTALS-DILITHIUM* in Fig. 2a that both dark and light purple plain bars have a constant value independently of the QBER, while the for dashed bars increase with it. If we label the overhead for the multi-signature t_{multi} and the overhead for the mono-signature t_{mono}, we observe in Fig. 2b (*SPHINCS+*) that they are related such that $t_{multi} \quad 2 \cdot t_{mono}$ for the case where QBER$= 0$, and the same happens with the other plots. However, when the QBER increases, this proportion is lost.

Regarding the authentication methods, we observe in all the plots that the multi-signature/verification times are always slower than the mono-signature/verification cases. For example, in Fig. 2d (*Rainbow*), we can see how the dashed dark red bars, multi-signature, are always significantly higher than the plain dark red bars, mono-signature, and the same happens for the salmon bars, which represent the verification.

Another interesting result is that the overhead of the authentication is nearly constant for the multi-case setting both with *CRISTALS-DILITHIUM* and *FALCON* as we observe in Fig. 2a and Fig. 2c, which is around 0.66% and 5.3% respectively, while it decreases significantly for the mono-cases. Even though for *SPHINCS+* and *Rainbow* the overhead



(a) *CRYSTALS-DILITHIUM* (b) *SPHINCS+*

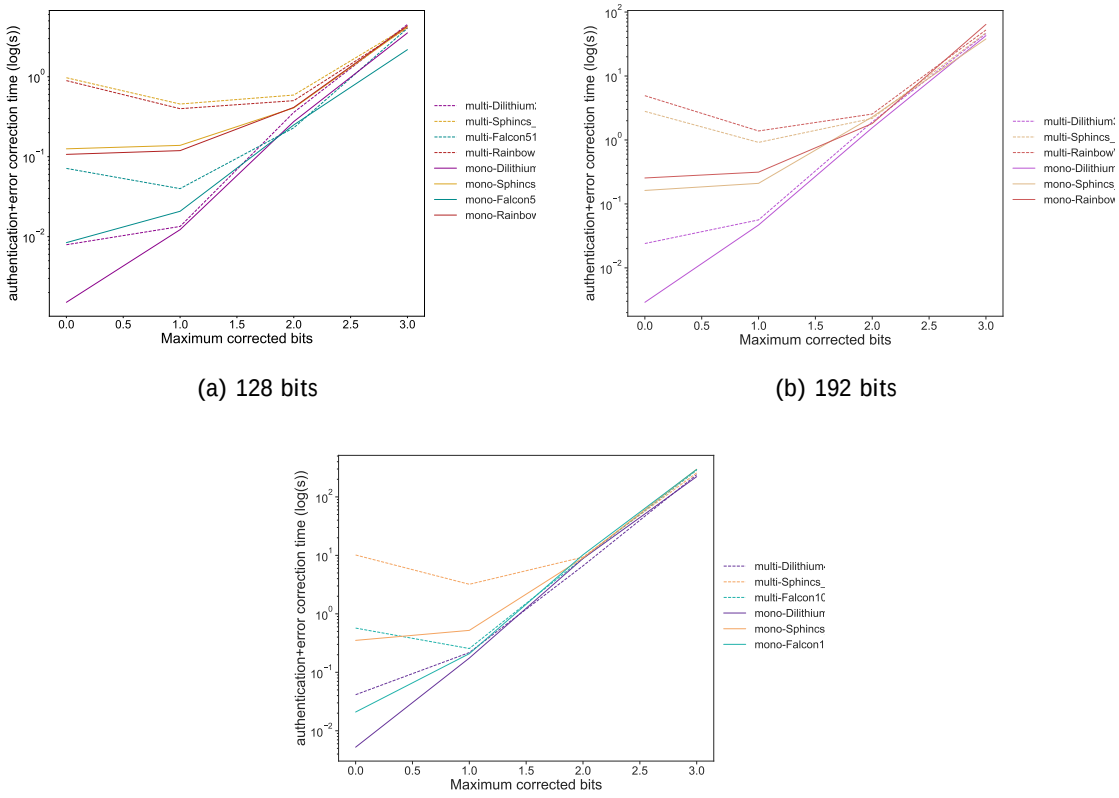
(c) *Falcon* (d) *Rainbow*

Figure 3: Dependance on the amount of *maximum corrected bits* in BB84 under multi- and mono-authentication schemes. In each plot, di□erent post quantum algorithm is studied. The the darker bars represent the signature time and the lighter ones the verification. The dashed bars indicate the use of the multi-authentication scheme, while the plain ones represent the mono-authentication case. In the plots the grey line is the deviation of the results, as they are the output of an average of 100 repetitions. The overhead of the authentication with respect to the time of the communication is plotted on top of each pair of bars. Notice that the time axis is plotted in logarithmic scale. To consult the plots in linear scale check Appendix J.2.

in the multi-case decreases, from 86% to 51.92% for *SPHINCS+* in Fig. 2b, it is still really high compared to the results for the mono-case, 9.86% and 7.95% as we can see in respectively Fig.2b and Fig.2d for *Rainbow*. Finally, for all the cases but *Rainbow* the signature is more expensive than the verification. For the rest of the results in this work we fixed the $QBER$ to 1.1% as it is the worst case experimentally seen.

Maximum number of corrected bits

The second studied parameter is how the maximum number of corrected bits a□ects the authentication performance. We observe that the mono-authentication time, plain bars in Fig. 3, remains constant with respect to the number of maximum corrected bits, while the multi-authentication time, the dashed bars, decreases. This decrease results from error-correcting up to a more significant number of errors, which means fewer aborts. The numbers on top of the purple bars in Fig. 3a are the number of aborts to exchange 100 keys of 128 bits. We plot this value on the other three subplots as well. Finally, since the number of aborts is a parameter unrelated to authentication, it is una□ected by the type



(a) 128 bits

(b) 192 bits

(c) 256 bits

Figure 4: Plots of the authentication time (signature time+verification time) plus the time it takes to conduct the error correction, with respect to the number of maximum corrected bits. These are plotted for three different security bits under a QBER of 0.011. The continuous line represents authenticating at the end of the communication (mono), while the dotted one represents authenticating at every step of the communication (multi). Notice that the time axis is plotted in logarithmic scale. To consult the plots in linear scale check Appendix J.3

of signature specified.

Then, we wanted to study the optimal amount of maximum corrected errors for each algorithm. We have seen in Fig. 3 that the authentication time decreases for the multi-case when the tolerance in the error correction increases. We also calculated the time to correct a certain number of errors given a key size and observed that it increases with the number of maximum corrected bits. A table with the maximum times it takes for each security level to correct between 0 and 3 errors can be found in the Appendix I. Putting both information together, we find the optimal values, which are the minimums for each curve in Fig. 4. All the algorithms for the mono-authentication, the continuous dashed lines in Fig. 4, have the minimum in origin at 0 corrected bits. However, a displaced minimum may appear in some algorithms, such as in the case of 128 bits in Fig. 4a with the multi-cases of *Rainbow*, discontinuous brown line, *FALCON*, discontinuous yellow line, and *SPHINCS+*, discontinuous orange line. There is a minimum because the optimal amount of bits corrected at maximum in those cases is not 0. We find that, in the 128 bits case, it is optimal to correct between 1 and 2 bits at maximum for the multi-authentication with *SPHINCS+* and *Rainbow* and to correct 1 with *FALCON*. For the case of 192 bits, we clearly see in Fig. 4b that the optimal number of corrected bits is 1 for the multi-

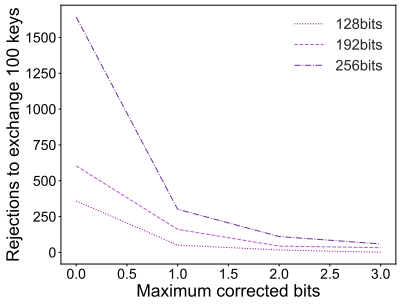


Figure 5: Plot to compare how many times are the different security levels (128, 192 and 192) rejected to exchange 10 keys, with respect to number of bits that are chosen to be corrected. These results are for a noisy channel with a 1% probability of having an error.

authentication in *SPHINCS+* and *Rainbow* and 0 bits for the rest of the cases. Finally, in Fig. 4c, we obtain a similar result as the previous case, but instead of *Rainbow* with *FALCON*, dotted-blue line.

We can check Fig. 5 to understand better why the authentication is more expensive for cases with no corrections. We can see that the greater the security level, dark purple, the more keys are rejected if no errors are corrected. There are more rejections because we insert a QBER of 0.011, and the longer the key, the more probable it is to have an error. We have found that the probability of having n errors in a key of length k_l is the probability of having $(k_l - n)$ bits correct multiplied by the probability of having n bits wrong multiplied by all the possible combinations that include n errors, that is

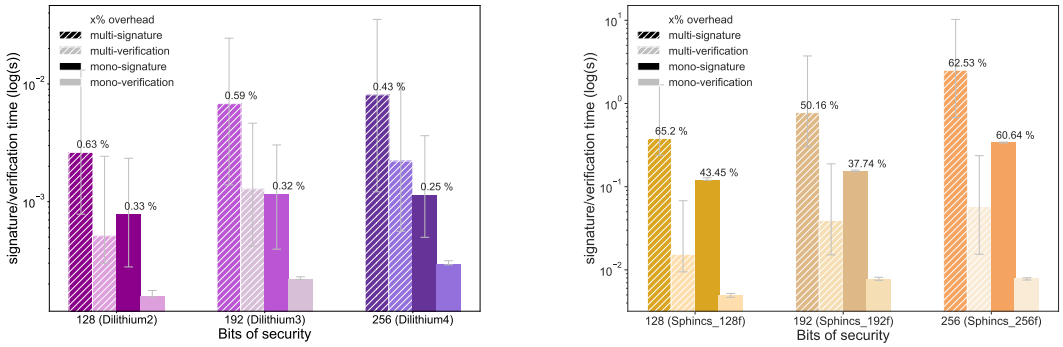
$$p_{n_{err}} = 0.99^{(k_l - n)} \cdot 0.01^n \cdot \frac{k_l!}{n!(k_l - n)!}. \tag{4}$$

In Appendix L we find these probabilities for every specific case.

Different security levels for each signature algorithm

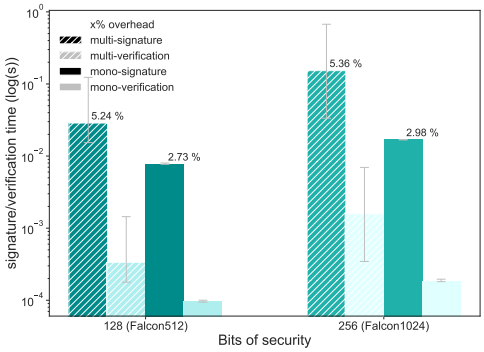
The next step is to determine how to do these algorithms work for transactions of 128, 192, and 256 bits of security. In Fig. 6 the both the dashed and plain bars are higher than the preceding ones, which means that the cost of the authentication increases with the security level. Dilithium in dark purple on top left, Fig. 6a, shows that the multi-verification times, light dashed bars, are greater than the mono-signature times, dark plain bars, whereas *FALCON* and *SPHINCS+* in Fig. 6c and 6b behave in the opposite way. For Figs. 6a, 6b and 6c as the security gets higher, the differences in mono/multi increase, which is what we expect. Falcon and Dilithium, Fig. 6c and 6a differences from 128 to 256 are quite small (for mono/multi).

Finally, in Fig. 7 we study how the different post-quantum algorithms perform compared to each other. We observe that the performance of the algorithms between the different security levels is similar. For the signature time we always have *DILITHIUM* ahead as the fastest, then *FALCON*, followed by *Rainbow* and finally *SPHINCS+*. Then for the verification time, we find that *FALCON* is the fastest, overtaking *DILITHIUM*. The latter is followed by *SPHINCS+* and finally *Rainbow*.

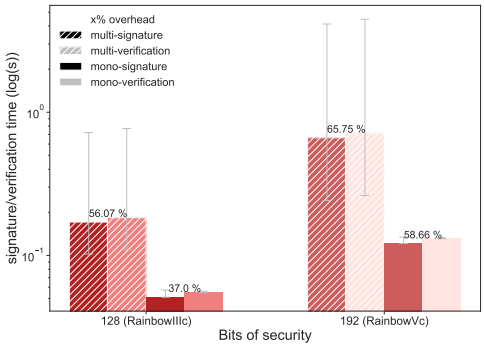


(a) CRYSTALS-DILITHIUM

(b) SPHINCS+



(c) FALCON



(d) Rainbow

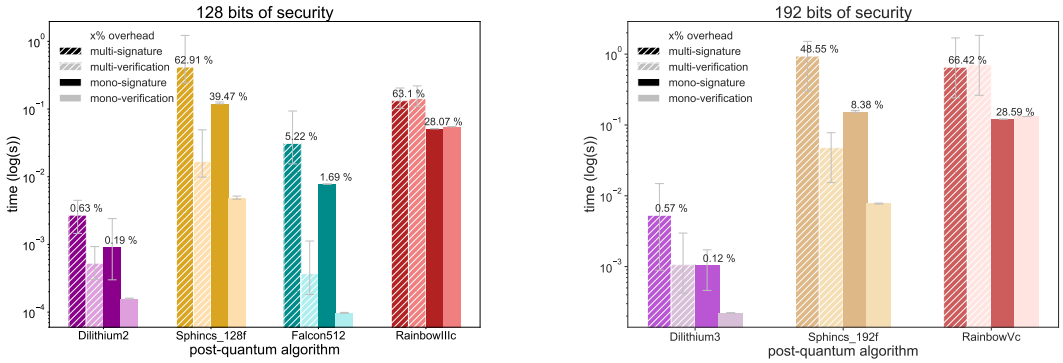
Figure 6: These plots analyze how the number of *security bits* impacts the time for the classical channel authentication in BB84 under multi- and mono-signature authentication schemes, for di□erent algorithms. The the darker bars represent the signature time and the lighter ones the verification. The dashed bars indicate the use of the multi-authentication scheme, while the plain ones represent the mono-authentication case. In the plots the grey line is the deviation of the results, as they are the output of an average of 100 repetitions. The overhead of the authentication with respect to the time of the communication is plotted on top of each pair of bars. Notice that the time axis is plotted in logarithmic scale. To consult the plots in linear scale check Appendix J.4

4.2 Continuous photon stream

In this section, we investigate which are the minimum times needed to wait to restart the QKD authentication for a given bit rate of photons transmitted in the quantum channel. But before doing so, we needed to determine how much time it takes for each algorithm to authenticate a certain number of bits. In the following results, we are not taking into account the multi-signature case anymore as we have already seen that the performance of the mono-signature surpasses the case with several steps.

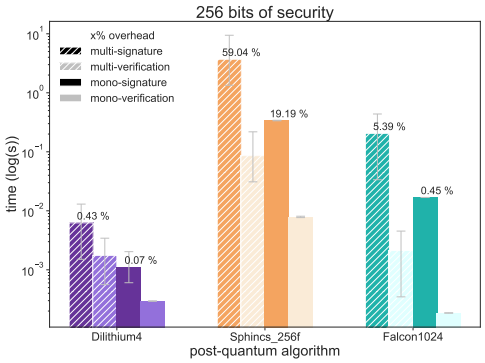
In Fig. 8 we observe that *CRYSTALS-DILITHIUM*, all the purple triangles, has the fastest authentication rate, while *Rainbow*, red squares, and *SPHINCS+*, yellow diamond, are the worst. In most cases, we observe that the authentication performance is almost independent of the number of bits being signed, at least when we sign up to 2000. The last remark is that to reach these results we first studied how much time every algorithm takes to authenticate from ≈ 0 to 2000 bits before averaging them to the results we just saw. They can be found in Appendix K.

In Table 5 we determine the number of signatures per second each algorithm can



(a) 128 bits

(b) 192 bits



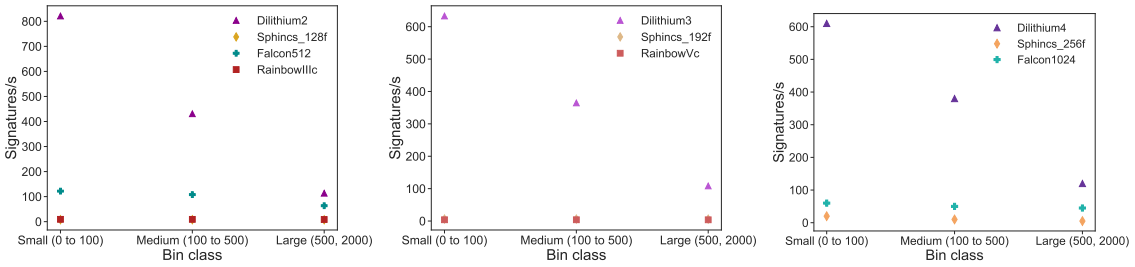
(c) 256 bits

Figure 7: Plots for three di□erent security bits. In each plot the the signature (dark bars) and verification times (light bars) for every post-quantum algorithm are calculated for two di□erent signature schemes (multi, dashed bars, and mono, plain bars). Each algorithm is represented in the same color palette in the several security levels. On top of the bars we represent the overhead, that is the cost of the authentication with respect to the communication. The grey bars represent the deviation of the result as they come from an average of 100 repetitions of the protocol. Notice that the time axis is plotted in logarithmic scale. To consult the plots in linear scale check Appendix J.5

perform concerning a given key rate. As there is not an exchange of more than 2000 bits in the indicated times for each algorithm, in the third column in the table, we can keep authenticating in that period. We observe that *CRYSTALS-DILITHIUM* is the fastest, while *Rainbow* is the slowest. One example of the possible recommendations to be given is that for a key rate of 50 kbps, the minimum period for authenticating what has been transmitted during that last period is 0.0013 s. The same would apply to the rest of the algorithms and key rates.

4.3 Discussion

Having compared multi-authentication and mono-authentication for di□erent parameters, we find that the novel authentication mode outperforms the standard one. The mono case improves the authentication as it performs the results in less time than the multi one, at least twice as fast. Noise is an important parameter that increases the di□erence in performance between both signature styles, observing that, for noisy channels, mono-authentication is the optimal implementation as it does not depend on any parameter of the QKD.



(a) 128 bits (b) 192 bits (c) 256 bits

Figure 8: Plots of the authentication rate per bin type for the post-quantum algorithms, each represented with a di□erent figure. The bin class is the averaged group of keys to be signed divided in three di□erent bins: small, medium and large.

PQ algorithm	Security level	50 kbps	300 kbps	1250kbps
CRYSTALS-DILITHIUM	Dilithium2	0.0013 s	0.0024 s	0.0089 s
	Dilithium3	0.0015 s	0.0028 s	0.0093 s
	Dilithium4	0.0016 s	0.0028 s	0.0092 s
SPHINCS+	sphincs-128f	0.1271 s	0.1280 s	0.1344 s
	sphincs-192f	0.1606 s	0.1619 s	0.1684 s
	sphincs-256f	0.2231 s	0.2245 s	0.2268 s
Rainbow	rainbowIIIc	0.1079 s	0.1091 s	0.1158 s
	rainbowVc	0.2546 s	0.2561 s	0.2621 s
FALCON	falcon512	0.0081 s	0.0093 s	0.0158 s
	falcon1024	0.0174 s	0.0185 s	0.0251 s

Table 5: Minimum period each post-quantum signature algorithm needs for authentication in terms of key rate.

Regarding the performance of the post-quantum algorithms, we have seen that *CRYSTALS-DILITHIUM* outperforms the rest in terms of signature speed, while *FALCON* does in terms of verification speed. However, the algorithm with the minimum overhead is *CRYSTALS-DILITHIUM*, which is always under 1%.

Checking closer into the several levels of security, we observe that the cost of the authentication is more expensive, which means it takes more time to be performed. Therefore, the lower the number of bits of security per transaction, the higher the authentication speed.

Another parameter we have studied is how many errors we correct in Bob's key. We have seen that, in general, it has less cost not to make any corrections for the mono-authentication, while for the multi-authentication, it is better to perform between 1 or 2 error corrections at a maximum. Furthermore, it is optimal to check more errors because the amount of signatures performed in the multi-case is more considerable as discarding the key due to a large number of errors also discards the signatures performed in that round. From this result, we conclude that if the quantum channel is costly, we are more interested in checking more errors, whereas if the quantum channel is exceptionally cheap, discarding the keys and repeating the exchange is the optimal solution.

Finally, we studied the case of a realistic QKD setup with a continuous photon stream. The minimal time to perform the authentication and the post-processing steps that do not include the basis sifting can be considered constant and almost independent of the number of bits to be authenticated. The previous statement holds for all the algorithms but *CRYSTALS-DILITHIUM*, which decreases its performance significantly for bigger bins.

Finally, from the table with the recommendations for the different key rates, we conclude that *CRYSTALS-DILITHIUM* outperformed the other post-quantum schemes.

We cannot yet guarantee the superiority of the mono-signature proposal outside of the confines of our simulation because we have only evaluated the algorithms in selected cases. However, it appears promising, and it would be fascinating to test the simulations on a more powerful computer to see if the results hold for a realistic setup.

5 Conclusion and outlook

We numerically analyzed the performance of four post-quantum algorithms from the NIST standardisation process that are *CRYSTALS-DILITHIUM*, *SPHINCS+*, *FALCON* and *Rainbow*. We first conducted a study of those algorithms in both authentication cases depending on the number of bits we were transmitting during the transaction. In particular, we studied three different security levels, which correspond to an exchange of a key of 128, 192, or 256 bits. We concluded that the mono-authentication outperformed the authentication per step in all the cases. We also found that *CRYSTALS-DILITHIUM* was significantly faster in all the situations than the other quantum-resistant algorithms and significantly faster than the rest of the communication steps. We also concluded that noise played an important role in a realistic QKD and optimized the relation of corrected bits for each algorithm. In fact, as quantum channels improve, the tolerance for errors can be lower. Finally, despite not having had the opportunity to compare these results to the existing signature algorithms that we use daily nowadays, such as ECDSA or RSA, we have had the opportunity to test the upcoming generation of algorithms that will replace the currently existing ones.

Then, we took a new approach to improve the accuracy of our results to a realistic scenario, a continuous photon transmission in the QKD setup, after discussing the numerics of the post-quantum algorithms and seeing that these algorithms may not only be used for these specific blocks of bits. We found that except for *CRYSTALS-DILITHIUM*, the time required to accomplish authentication was nearly independent of the number of bits to be authenticated up to 2000 bits. Then, we provided recommendations aimed at QKD equipment administrators to adjust which are the best authentication mechanisms to use depending on the key rates. Indeed, this is the minimum quality of service that each configuration can give. From here, one possible line of investigation is to study how the signature algorithms perform with respect to distance. Any multi-user QKD network has different distances between the users, and the number of signatures we can perform per second also depends on that.

Finally, there are many ways to continue this research from this point. The next step could be to design the security proof of the mono-authentication method to prove mathematically that this method is still as secure as the mono-signature case. Another path would be to use $2-$universal hash functions instead of $SHA3_{512}$ for the hashing in the error correction, as was suggested in the original paper of the security proof of any QKD protocol [ECR04]. Nevertheless, the most essential and promising investigation path to be continued from this point is the use of KEMTLS instead of signatures [SSW21]. The implementation in QKD of this recently proposed authentication method may be beneficial as signature schemes generally have larger public key/signature sizes compared to the public key/ciphertext sizes we can find in KEMTLS.

Bibliography

[AA19] Frank Arute and et. al. Arya. Quantum Supremacy using a Programmable Superconducting Processor. *Nature*, 574:505–510, 2019.

[ABB+15] Jean-Philippe Aumasson, Daniel J. Bernstein, Ward Beullens, Christoph Dobraunig, Maria Eichlseder, Scott Fluhrer, Stefan-Lukas Gazdag, Andreas Hülsing, Panos Kampanakis, Stefan Kölbl, Tanja Lange, Martin M. Lauridsen, Florian Mendel, Ruben Niederhagen, Christian Rechberger, Joost Rijneveld, Peter Schwabe, and Bas Westerbaan. Sphincs: Practical stateless hash-based signatures. *Springer*, 9056, April 2015.

[BB84] Charles H Bennett and Gilles Brassard. Quantum cryptography: Public key distribution and coin tossing. In *Quantum cryptography: Public key distribution and coin tossing*, pages 475–480, Berlin, Heidelberg, August 1984. Springer-Verlag.

[Ber09] Daniel J Bernstein. Introduction to post-quantum cryptography. In *Post-quantum cryptography*, pages 1–14. Springer, 2009.

[BL17] Daniel J Bernstein and Tanja Lange. Post-quantum cryptography. *Nature*, 549(7671):188–194, 2017.

[CRE04] Matthias Christandl, Renato Renner, and Artur Ekert. A generic security proof for quantum key distribution, 2004.

[DKL+18] Léo Ducas, Eike Kiltz, Tancrède Lepoint, Vadim Lyubashevsky, Peter Schwabe, Gregor Seiler, and Damien Stehlé. Crystals-dilithium: A lattice-based digital signature scheme. *IACR Transactions on Cryptographic Hardware and Embedded Systems*, 2018(1):238–268, Feb. 2018.

[DLP14] Léo Ducs, Vadim Lyubashevsky, and Thomas Prest. E□cient identity-based encryption over ntru lattices. *Springer*, 8874, October 2014.

[DS05] Jintai Ding and Dieter Schmidt. Rainbow, a new multivariable polynomial signature scheme. *Springer*, 3531:164–175, June 2005.

[ECR04] Arthur Ekert, Matthias Christandl, and Renato Renner. A generic security proof for quantum key distribution. *Hardvard Edu*, March 2004.

[Eke91] Artur K. Ekert. Quantum cryptography based on bell's theorem. *Phys. Rev. Lett.*, 67:661–663, Aug 1991.

[FMC10] Chi-Hang Fred Fung, Xiongfeng Ma, and H. F. Chau. Practical issues in quantum-key-distribution post-processing. *Physical Review A*, 81(1):012318, January 2010. arXiv: 0910.0312.

[Gro96] Lov K. Grover. A fast quantum mechanical algorithm for database search. *arXiv:quant-ph/9605043*, November 1996. arXiv: quant-ph/9605043.

[KMG+20] E. O. Kiktenko, A. O. Malyshev, M. A. Gavreev, A. A. Bozhedarov, N. O. Pozhar, M. N. Anufriev, and A. K. Fedorov. Lightweight authentication for quantum key distribution. *IEEE Transactions on Information Theory*, 66(10):6354–6368, October 2020. arXiv: 1903.10237.

[Kob87] Neal Koblitz. Elliptic curve cryptosystems. *Mathematics of Computation*, 48(177):203–209, 1987.

[Llo96] Seth Lloyd. Universal Quantum Simulators. *Science New Series*, 273:1073–1078, August 1996.

[Mil86] Victor S. Miller. Use of elliptic curves in cryptography. In Hugh C. Williams, editor, *Advances in cryptology — CRYPTO '85 proceedings*, pages 417–426, Berlin, Heidelberg, 1986. Springer Berlin Heidelberg.

[MSU12] Michele Mosca, Douglas Stebila, and Berkant Ustaoglu. Quantum Key Distribution in the Classical Authenticated Key Exchange Framework. *arXiv:1206.6150 [quant-ph]*, June 2012. arXiv: 1206.6150.

[Pre12] John Preskill. Quantum computing and the entanglement frontier. *arXiv:1203.5813 [cond-mat, physics:quant-ph]*, November 2012. arXiv: 1203.5813.

[PZ04] John Proos and Christof Zalka. Shor's discrete logarithm quantum algorithm for elliptic curves, January 2004. Number: arXiv:quant-ph/0301141 arXiv:quant-ph/0301141.

[RSA78] R L Rivest, A Shamir, and L Adleman. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Massachusetts Institute of Technology*, page 15, 1978.

[Sha49] C. E. Shannon. Communication theory of secrecy systems. *The Bell System Technical Journal*, 28(4):656–715, 1949.

[Sho97] Peter W. Shor. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. *SIAM Journal on Computing*, 26(5):1484–1509, October 1997. arXiv: quant-ph/9508027.

[SSW21] Peter Schwabe, Douglas Stebila, and Thom Wiggers. More e□cient post-quantum KEMTLS with pre-distributed public keys. In Elisa Bertino and Haya Shulman, editors, *Proc. 26th European Symposium on Research in Computer Security (ESORICS) 2021*, volume 12972 of *LNCS*, pages 3–22. Springer, October 2021.

[WZ82] W. K. Wootters and W. H. Zurek. A single quantum cannot be cloned. *Nature*, 299(5886):802–803, October 1982. Number: 5886 Publisher: Nature Publishing Group.

[WZW+21] Liu-Jun Wang, Kai-Yi Zhang, Jia-Yong Wang, Jie Cheng, and et. al. Experimental authentication of quantum key distribution with post-quantum cryptography. *npj Quantum Information*, 7(1):1–7, May 2021.

A Quantum channels

A quantum channel is the most general quantum operation and consists of a completely-positive trace-preserving map, Λ, acting on a quantum state, ρ, such as

$$\begin{aligned} \Lambda : B(H_{in}) &\longrightarrow B(H_{out}) \\ \rho_{in} &\longrightarrow \Lambda(\rho_{in}) = \rho \,, \end{aligned} \tag{5}$$

where $B(H_i)$ is the set of bounded operators acting on H_i. Completely-positive means that the action of the map on a subsystem has to preserve the positivity of the original density matrix, that is, if $\rho_{in} = \rho_{AB}, (\mathrm{I}_A \quad \Lambda)(\rho_{AB}) > 0$. On the other hand, the trace preserving property claims that the action of the map should not modify the value of the trace of ρ after its application. This is mathematically denoted by $tr\left(\Lambda(\rho)\right) = tr(\rho)$.

B How the preparation of a state a□ects the measurement outcome

If we initially prepared the state $|\Psi\ $ in a basis $\{|\psi_m\ \}$, we will obtain the same output if we perform the measurement that projects into the same basis as the input of the measure will be an eigenstate.

$$p(m|\Psi) = |\ \ \psi_m|\Psi\ \ |^2 = \begin{cases} |\ \ \psi_m|\psi_i\ \ |^2 = \begin{cases} 1| \ _{i=m} \\ 0| \ _{i=m} \end{cases} & \text{if } |\Psi\ \text{ is eigenstate} \\ \sum_i |\alpha_i|^2 |\ \ \psi_m|\psi_i\ \ |^2 & \text{otherwise.} \end{cases} \tag{6}$$

However, if we choose a di□erent one, from this perspective, the state will be seen as a superposition that can project into more of the outputs of the measurement and not always into the same as before.

C Qubits

Qubits are the quantum mechanical units of information and are mathematically defined as the superposition of an orthonormal two-level basis. An example of a qubit is

$$|\psi\ = \alpha\,|0\ + \beta\,|1\ \ , \quad \rho_{qubit} = |\psi\ \quad \psi|\,, \tag{7}$$

where $|\alpha|^2 + |\beta|^2 = 1$ and α, β C. In this case we have represented the qubit in the computational basis, $\{|0\ \ , |1\ \}$. This is not the only basis choice to represent a qubit. In fact, there are infinite choices, however, in this work we prepared the qubits only in the computational and in the Hadamard basis, $\{|+\ \ , |-\ \}$, where $|+\ \ = \frac{1}{\sqrt{2}}(|0\ \ + |1\ \ )$ and $|-\ \ = \frac{1}{\sqrt{2}}(|0\ \ - |1\ \ )$. The Pauli-X and Pauli-Z basis are analogous to the Hadamard and computational basis, respectively. The Pauli operators X, Y, Z together with the identity I_2 read as

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Y = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \text{ and } \quad \mathrm{I}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix},$$

which generate the basis of the space of the operators, B, for dimension 2. In this work, we measure qubits with Z and X, where both have the potential measurement outcomes of 1 and -1.

D One Time Pad

The way of sending private information completely securely is by using a One-Time Pad (OTP). Two users, Alice and Bob, share a key of N perfectly correlated bits such that $N \geq M$, where M is the length of the message Alice wants to send. To send it securely, Alice performs a *XOR* operation between every bit of the secret key k_i and the message m_i such that $r_i = m_i \quad k_i$ obtaining a random bit-string $\vec{r}$ that is sent through the insecure classical channel but it can be shown it reveals no information about the original message. When Bob receives $\vec{r}$, he performs the XOR operation again obtaining the original message $r_i \quad k_i = (m_i \quad k_i) \quad k_i = m_i$. If the secret bits are never reused, i.e., the key is used only once, this message is indecipherable to an eavesdropper.

E No-cloning theorem proof

Proof. We seek to prove that it does not exist any unitary transformation U acting on the couple of states $|\psi \ |ref.$, where $|\psi$ is the state to be copied and $|ref.$ the reference state acting as a copy machine, such that $U\,|\psi \ |ref. = |\psi \ |\psi$.

We can begin by assuming that U *s.t.* $U\,|\psi_i \ |ref. = |\psi_i \ |\psi_i$, where $|\psi_i$ is an element of an arbitrary basis. Then, if $|\phi_j$ is one element of another basis,

$$U\,|\psi_i \ |ref. = |\psi_i \ |\psi_i$$

$$U\,|\phi_j \ |ref. = |\phi_j \ |\phi_j \ .$$

Hence,

$$\phi_j\phi_j|\psi_i\psi_i = \phi_j| \ ref.|U^{\dagger}U\,|\psi_i \ |ref. = \phi_j|\psi_i \tag{8}$$

from which we deduce that $\phi_j|\psi_i = 0$ or $\phi_j|\psi_i = 1$. This in general will not hold, proving that @U, because in general $|\psi_i$, $|\phi_j$ are not orthogonals nor proportional. □

F Security in QKD

We employ a variable called the QBER to ascertain whether or not an eavesdropper has intervened. We said that in the case of the $BB84$ protocol, the maximum allowed value for the QBER is 11%. This outcome is known as the Holevo bound, consisting of how much classical information someone can send through a quantum channel. If there is a third party connected in the quantum channel through an entangled state on its Hilbert space, meaning that when Bob measures, this action indirectly prepares the eavesdropper's state, leaking some information about the states Alice was sending. Mathematically the outcome of 11% represents the maximum allowed errors until the mutual information between Alice and Bob is less than the one between Alice and the eavesdropper, which is why we must remain under that value and discard the keys otherwise.

G Implementation details

To reach these results, we simulate the QKD protocol of BB84 as explained in Section. 2.3. We write the simulation in a Pyhton3 script and run it in a MacBook Pro (2021) with a memory of 16 GB and an Apple M1 Pro chip. The script of the simulation of the BB84 protocol, together with the authentication, error correction, and privacy amplification, can be found in https://github.com/sandbox-quantum/pqc-qkd.git. In this repository, we also find the data-processing to obtain the plots in this work.

H Key-pair and signature size for each post-quantum algorithm

In Table 6, we have collected the signatures and key pair sizes for each post-quantum algorithm. Some essential features of the studied algorithms are contained in the following table. We observe some di□erences, such that the signature size from *Rainbow* is compared to the other algorithms, and the same happens with the key-pair in *SPHINCS+*.

Post-Quantum algorithm	(bytes)	DILITHIUM	SPHINCS+	FALCON	Rainbow
128 bits	sk	2544	64	1281	626048
	pk	1312	32	897	882080
	sig	2420	16976	659	164
192 bits	sk	4016	96	-	1408736
	pk	1952	48	-	1930600
	sig	3293	35664	-	212
256 bits	sk	4880	128	2305	-
	pk	2592	64	1793	-
	sig	4595	49216	1276	-

Table 6: In this table we show the sizes in bytes of the secret keys, *sk*, public keys, *pk*, and signatures, sig. These sizes are given for three di□erent security bits: $128, 192$, and 256, for each post-quantum algorithm studied in this work. All the values were obtained from https://bench.cr.yp.to/results-sign.html.

I Table with error correction times

In this section it is calculated the failure time for the error correction. In Table 7 we find the maximum (averaged) time the script losses in doing brute force search for the error.

Max corrected bits	0	1	2	3
128 bits	$\approx 0.004s(1h)$	$\approx 0.021s(129h)$	$\approx 0.9s(8257h)$	$\approx 38s(691000h)$
192 bits	≈ 0.005 s $(1h)$	$\approx 0.035s(193h)$	$\approx 3s(18529h)$	$\approx 187s(2341089h)$
256 bits	$\approx 0.006s(1h)$	$\approx 0.06s(257h)$	$\approx 7s(32897h)$	$\approx 580s(5559937h)$

Table 7: Maximum time (s) and number of hashes performed in parenthesis, in three di□erent security levels, 128, 192 and 256, for di□erent maximum numbers of corrected bits.

The number of hashes performed that appears in every cell come from the combinatorial formula that is

$$ {}_nC_r = \frac{n!}{r!(n-r)!}, \tag{9} $$

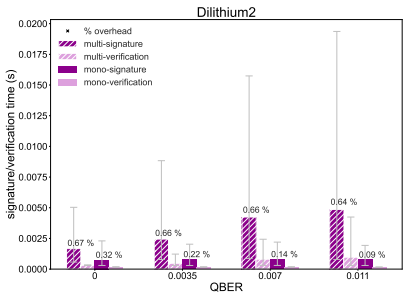
where ${}_nC_r$ are the total number of hashes performed, n is the number of bits we study (128, 192 or 256) and r is the maximum number of corrected bits.

In this case, we are also appending the hashes from the previous round. For example, in a key of length 128, in a brute force search of *up to* 1 error, we perform first the hash of the original string (1 hash), and then the possible 128 combinations corresponding of flipping only one bit (128 hashes). In total, we have 129 hashes. The same applies to the other cases.

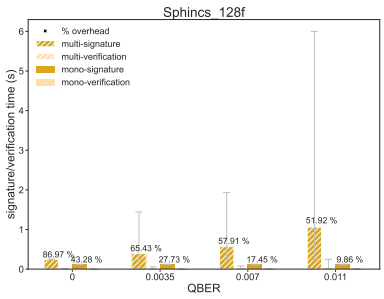
J Plots in linear scale

In this section, we show the plots we have seen in Section 4, but in linear scale. The results are the same as those in the indicated section but shown in seconds, not log(s).

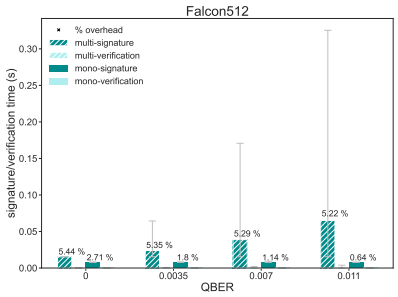
J.1 Signature and verification time vs. QBER



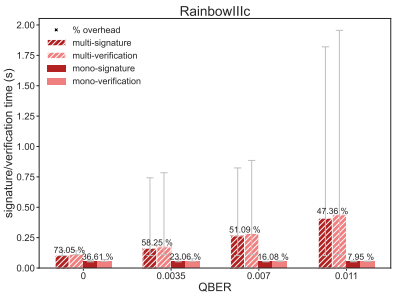
(a) *CRYSTALS-DILITHIUM*



(b) *SPHINCS+*

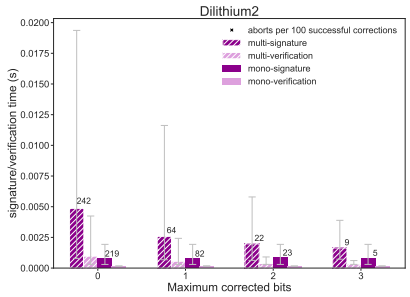


(c) *FALCON*

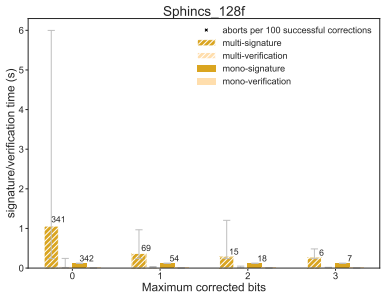


(d) *Rainbow*

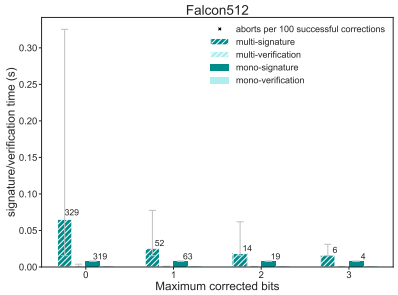
J.2 Signature and verification time vs. number of maximum corrected bits



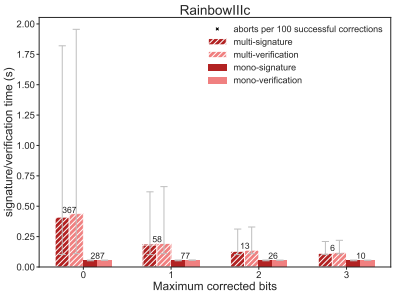
(a) *CRYSTALS-DILITHIUM*



(b) *SPHINCS+*

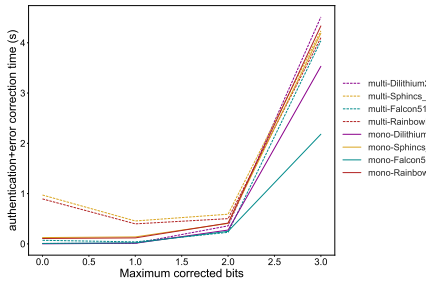


(c) *FALCON*

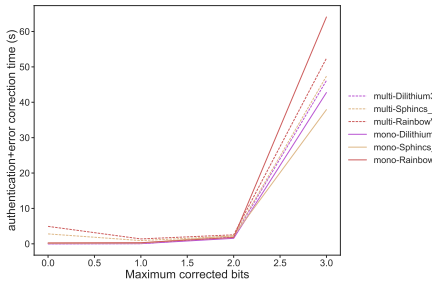


(d) *Rainbow*

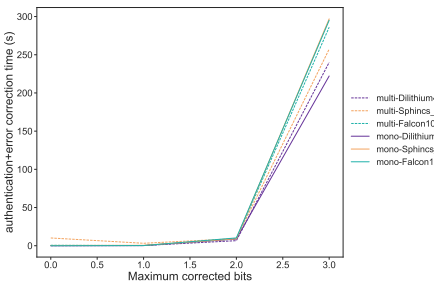
J.3 Authentication + error correction time vs. maximum corrected bits



(a) 128 bits

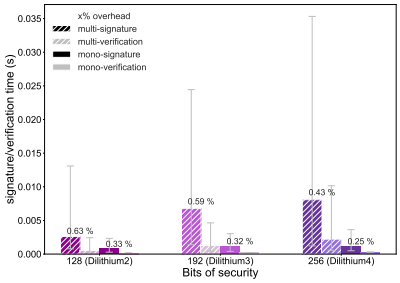


(b) 192 bits

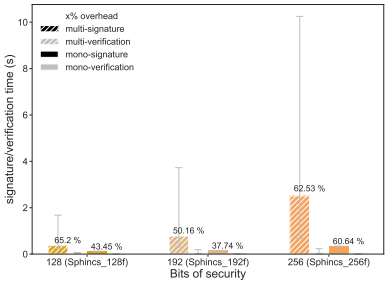


(c) 256 bits

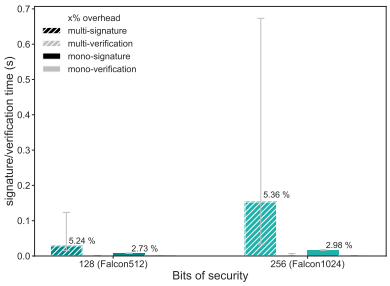
J.4 Signature and verification time vs. security level



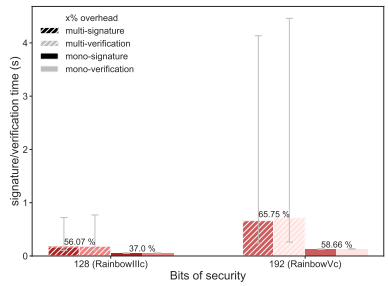
(a) *CRYSTALS-DILITHIUM*



(b) *SPHINCS+*

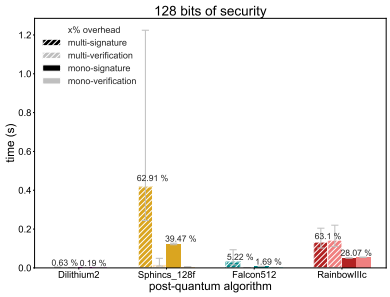


(c) *FALCON*

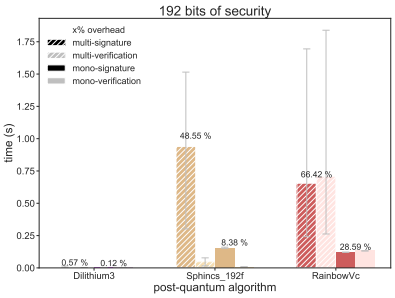


(d) *Rainbow*

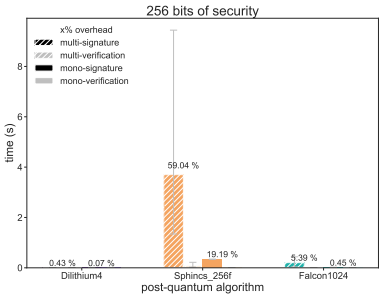
J.5 Signature and verification time vs. post-quantum agorithms



(a) 128 bits



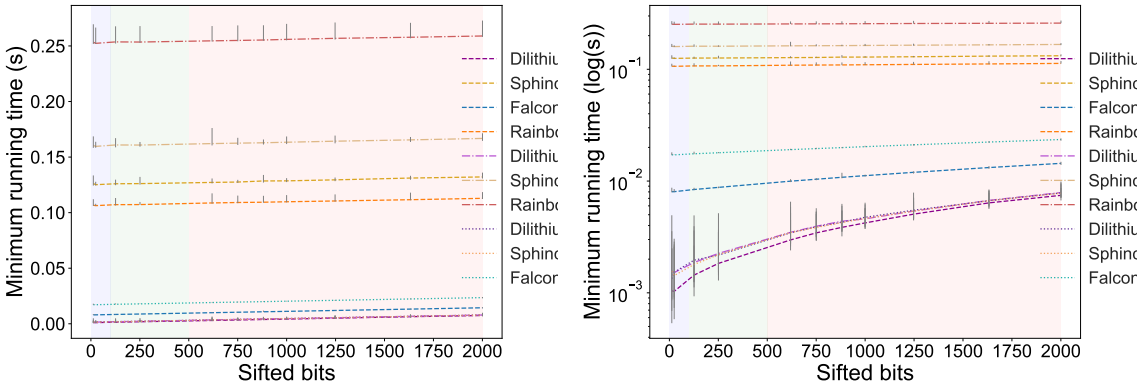
(b) 192 bits



(c) 256 bits

K Minimum times to perform the signatures for key sizes from 0 to 2000

We show in Fig. 14 the minimum period to authenticate with every algorithm and the respective security level. We observe the results both in a linear scale in Fig. 14a and logarithmic scale in Fig. 14b.



(a) Linear scale

(b) Logarithmic scale

Figure 14: Several lengths of bit-strings named sifted bits are authenticated and it determines the time each algorithm takes to perform them. In grey bars, it is indicated the uncertainty of the outcome as they are the output of an average. The plot is colored in blue for small bins (0 to 100 bits), green for medium (100 to 500), and red for large (500 to 2000).

L Table with error probabilities

In this section the probabilities of having n errors in a bit-string of k_l bits. It is calculated using Eq. 4.

Number of errors	0	1	2	3
$k_l = 128bits$	≈ 0.276	≈ 0.357	≈ 0.229	≈ 0.194
$k_l = 192bits$	≈ 0.145	≈ 0.280	≈ 0.272	≈ 0.174
$k_l = 256bits$	≈ 0.076	≈ 0.197	≈ 0.254	≈ 0.217

Table 8: Probabilities of having the indicated number of errors in a key of length k_l.

Comparing the numbers on the table, one can easily see that, for example, having 0 errors with a 1% of QBER is highly unusual for a key length of 256 while having 3 errors happens with a probability of more than 40%. Correcting no errors leads to discarding many times, but when there is a correction of three bits, the abortion rate drops.