cryptography

IMPACT FACTOR 2.1

CITESCORE 5.0

Article

Design and Implementation of a Quantum-Assisted Digital Signature

Marta Irene García-Cid, Rodrigo Martín, David Domingo, Vicente Martín and Laura Ortiz

MDPI

https://doi.org/10.3390/cryptography9010011

cryptography

MDPI

Article

Design and Implementation of a Quantum-Assisted Digital Signature

Marta Irene García-Cid [1,2,*], Rodrigo Martín [3], David Domingo [3], Vicente Martín [2] and Laura Ortiz [2,4]

[1] Indra Sistemas S.A., 28850 Madrid, Spain
[2] Center for Computational Simulation, Universidad Politécnica de Madrid, 28660 Madrid, Spain
[3] Indra Sistemas de Comunicaciones Seguras, 28108 Madrid, Spain
[4] Departamento de Arquitectura y Tecnología de Sistemas Informáticos, ETSIInf, Universidad Politécnica de Madrid, 28660 Madrid, Spain
* Correspondence: migarcia@indra.es

Abstract: We propose a new quantum-assisted digital signature (Q-DS) protocol based on the composite of truly random symmetric keys generated by quantum key distribution with secure standardized hash functions, which allows for high parameterization to provide different security levels. The protocol is demonstrated to be secure, it is implemented, and its performance is tested for several system configurations. A comparative evaluation of the results obtained for Q-DS is carried out with 6 pre-quantum and 12 post-quantum digital signature algorithms. The results show that the Q-DS overperforms during the signature generation and verification processes, while its performance is affected by the key generation process. However, using more efficient QKD devices, this process can be highly improved, making the Q-DS protocol comparable to the most efficient post-quantum solution, i.e., *CRYSTALS-Dilithium*.

Keywords: quantum cryptography; digital signature

Academic Editor: Josef Pieprzyk

Received: 6 December 2024
Revised: 14 January 2025
Accepted: 27 January 2025
Published: 31 January 2025

Citation: García-Cid, M.I.; Martín, R.; Domingo, D.; Martín, V.; Ortiz, L. Design and Implementation of a Quantum-Assisted Digital Signature. *Cryptography* **2025**, *9*, 11. https://doi.org/10.3390/cryptography9010011

1. Introduction

Digital signatures aim to guarantee the identity of the sender of a message, as well as the integrity and non-repudiation of the information sent. The former means that the content of the message has not been modified after signing, and the latter implies that the signer cannot deny its authorship. Over the years, a large number of digital signature algorithms (DSAs) have been proposed, analyzed, and standardized. Today, the most widely used DSAs are RSA and signatures based on elliptic curves, among others. Both are built from pre-quantum asymmetric paradigms in which the security is based on one-way functions, i.e., mathematical problems that are easy to calculate in one direction but extremely hard to calculate in the reverse direction, such as the factorization of large prime numbers.

The arrival of the quantum computer has altered the security panorama. In particular, the ability to implement Shor's [1] algorithm in a Cryptographically Relevant Quantum Computer (CRQC) will make the pre-quantum DSA vulnerable. To solve this security challenge, two new cryptographic paradigms have been explored in recent years: quantum cryptography [2] and post-quantum cryptography (PQC) [3]. PQC algorithms are based on new mathematical problems, such as lattices or codes, among others. Currently, two PQC-DSA have been standardized by the National Institute of Standards and Technology (NIST) [3] (CRYSTALS-DILITHIUM [4] and SPHINCS+ [5]), while a third one is on the process of standardization (FALCON). However, it is important to note that these new

algorithms are demonstrated to be resistant only against a quantum computer running Shor's algorithm, although it is still unknown what a CRQC might be capable of.

On the other hand, quantum cryptography has its foundations in quantum mechanical properties instead of a mathematical problem. Consequently, it is resistant to computational attacks even if the attacker has access to a CRQC. The most widely known quantum cryptographic primitive is the quantum key distribution (QKD), which allows the generation of random symmetric keys between two users. Currently, commercial devices are already available to execute QKD protocols, which have been implemented in long-range networks and whose QKD-generated keys have been employed to encrypt transmitted information [6]. As a drawback, the hardware utilized for quantum cryptography requires greater technological maturity to improve its performance and range, and several challenges related to implementation attacks must be overcome [7].

Due to the aforementioned panorama of cryptography, it is advised for certain scenarios to hybridize or combine cryptographic paradigms (i.e., pre-quantum and post-quantum cryptography) [8] to provide solutions against the unknown real impact that quantum computers could have in pre-quantum cryptography, the youth of PQC and the low Technology Readiness Level and lack of certification of QKD devices. To this end, we focus our research on the development of composite cryptographic systems. Specifically, in this paper, we present a quantum-assisted digital signature (Q-DS) based on the composite of symmetric keys generated by QKD with NIST-recommended and secure cryptographic hash functions [9].

In the last two decades, several quantum digital signature (QDS) protocols have been proposed. The first pure QDS was proposed in 2001 [10] and was based on Lamport's one-way function scheme. This signature scheme has been demonstrated to be Information-Theoretic Secure (ITS) but, despite being a robust QDS, its implementation is currently not possible since it requires the use of long-term quantum memories. In 2014, a QDS scheme was proposed that did not require quantum memories for its implementation since the measurement of quantum states is performed upon receiving them [11]. In this case, the main disadvantages are the vulnerability to coherent forging attacks. In addition, the presence of eavesdroppers during the protocol is not considered in the security analysis, which means that the use of secure quantum channels must be guaranteed. Two years later, a new scheme was published whose security no longer depends on the availability of secure quantum channels [12], but on secure classic channels in order to carry out a symmetrization step. In this case, the signer receives and measures the quantum states of the receivers, without carrying out error correction and privacy amplification mechanisms corresponding to the key distillation processes, which are used both for the signature and for the securitization of the classic channels. These types of protocols have certain limitations regarding the maximum length of the message that can be signed, i.e., 1 bit. More recently, in 2021, a scheme was proposed where the symmetrization step [13] was removed. With this improvement, the scheme achieves greater execution efficiency, but the limitation regarding the length of the signed message remains the same.

In this paper, we propose a new Q-DS protocol, which simplifies the previous schemes and allows us to sign messages with an arbitrary length and for different levels of security. Using QKD-generated keys as the basis of the scheme, the protocol is built employing known hash functions, with current NIST recommendations, thus providing a highly parametrizable composite system that integrates quantum and symmetric cryptography. The security of the protocol is analyzed to demonstrate its robustness against integrity, authenticity and repudiation attacks, considering the security strength of the pre-quantum functions. It should be taken into account that even though the full protocol is not ITS, because of the composite of hash functions with QKD, the pre-quantum mechanisms

used are considered quantum-resistant and allow us to remove the need for vulnerable asymmetric algorithms. For evaluation purposes, we implement the protocol and execute different experiments modifying the system parameters and measuring its efficiency during the key generation, signature generation and signature verification processes. Finally, we compare our results with the ones obtained from the execution of 6 pre-quantum DSA and 12 PQC DSA algorithms in order to perform a comparative evaluation for different security levels.

2. Protocol Design

The general structure of the proposed Q-DS protocol is divided into a distribution phase and a messaging phase.

The protocol begins with the distribution phase, in which Alice and Bob carry out a QKD process, for example the well-known decoyed-states BB84 protocol [2]. This protocol is executed and the result is a secret symmetric key k_1 of length l_k, shared by Alice and Bob. These same steps are carried out between Alice and Charlie (a second possible receiver of the message), obtaining, at the end of the process, a secret symmetric key k_2 of length l_k. The keys k_1 and k_2 are divided into n blocks of a certain length and stored. Each block has a fixed labeling $B = \{B_1, B_2, ..., B_n\}$ which is known by all the users. Once the generation of symmetric keys between pairs is finished, Bob and Charlie carry out a process of exchanging random blocks of keys through an encrypted and authenticated classic channel, for example, with other QKD-generated keys or hybridizing QKD with *CRYSTALS-KYBER* through a Key Derivation Function (KDF). For this, given Z_n as the group of permutations of n elements so that $|Z_n| = n!$, we denote the random permutation of n elements $\gamma_j \in Z_n$, with $j = \{b, c\}$, as

$$\gamma_j = \begin{pmatrix} 1 & 2 & \cdots & n \\ a_1 & a_2 & \cdots & a_n \end{pmatrix}, \gamma_j(i) = a_i, a_i \in \{1, 2, ..., n\} \quad (1)$$

Bob sends to Charlie the subset of elements k_1' corresponding to the first $n/2$ indices of B associated with k_1 after applying γ_b. Next, Charlie sends to Bob the subset of elements k_2' corresponding to the first $n/2$ indices of B associated with k_2 after applying γ_c. Alice does not know which blocks Bob and Charlie have exchanged. This is a necessary condition so that Alice cannot repudiate the authorship of the message, as we will see later.

The protocol continues with a messaging phase in which Alice generates the digital signature for a given message and sends it to the recipients, who verify the validity of both the message and the signature, as shown in Figure 1. Alice, the signer, wants to send Bob, one of the possible verifiers, a message (m) of any length and its associated signature (S_a). To do so, Alice generates a combined key k_a of length $2l_k$ by concatenating the secret symmetric key shared with Bob to the secret symmetric key shared with Charlie ($k_a = k_1||k_2$). Alice calculates the hash of the message as $h_a = h(m)$. The use of a hash function allows us to take as input text of any size and returns a fixed-size digest equal to δ_m. Then, Alice encrypts h_a with the composed secret key k_a using a One-Time Pad (OTP) encryption function, whereby an element-by-element XOR is made between h_a and k_a, obtaining $c_a = ENC_{k_a}(h_a)$. In order to carry out this step, h_a and k_a must have the same length, so $2l_k = \delta_m$. At this point, it is important to remember that k_1 and k_2 were divided into n blocks each, so k_a has $2n$ blocks. And, in turn, c_a will also be made up of $2n$ blocks. As a last step, Alice generates individual hashes for each of the blocks of c_a, each with length δ_B, thus obtaining the signature S_a of length $2n \cdot \delta_B$, associated with her message. Alice then sends to Bob the (m, S_a) tuple. Upon reception, Bob starts the signature verification process, which consists of performing the same steps as those performed by

Alice for the generation of S_a. Thus, Bob makes the concatenation of the secret symmetric key shared with Alice and the known blocks of Charlie's key ($k_b = k_1||k'_2$). By using the same hash function as Alice, Bob calculates the hash of the message as $h_b = h(m)$ and encrypts h_b with k_b using an OTP cipher, $c_b = ENC_{k_b}(h_b)$. Finally, Bob generates the hash of each of the blocks of c_b, taking into account that he does not know half of the blocks in c_b. Upon performing this action, he obtains S_b and compares it with the received S_a in such a way that, if all the known elements match, with the number of coincidences being above a predefined verification threshold T_V (see Section 3), Bob accepts the message and the signature as genuine. However, if the number of coincidences is below T_V, Bob rejects the validity of the message and the signature. To complete the sending chain, Bob forwards the tuple (m, S_a) to Charlie, who performs the same signature verification procedure as Bob. The steps of the full protocol are shown in Figure 2.

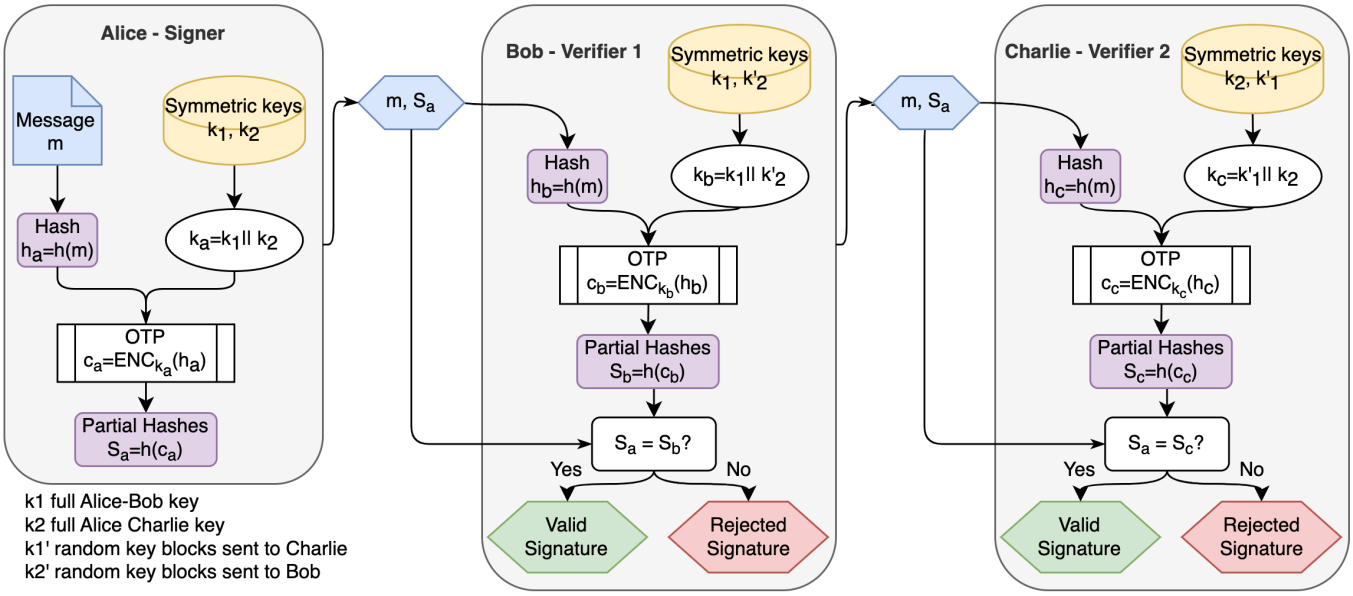

Figure 1. Messaging phase of the proposed Q-DS scheme. Alice generates the signature S_a associated with a message m and sends both to Bob, who verifies their validity. Then, the message and the signature are forwarded to Charlie, who also verifies them. OTP = One-Time Pad.

It is worth highlighting that in the case of multiple receivers (i.e., more than two possible verifiers in Q-DS), the increase in the number of possible verifiers implies an increase in the number of quantum communication links to perform QKD. Thus, complex scenarios like today's Internet are not supported. However, the proposed Q-DS might be considered for more constrained environments where high-security mechanisms are needed and the number of users involved is restricted, as may occur in military contexts.

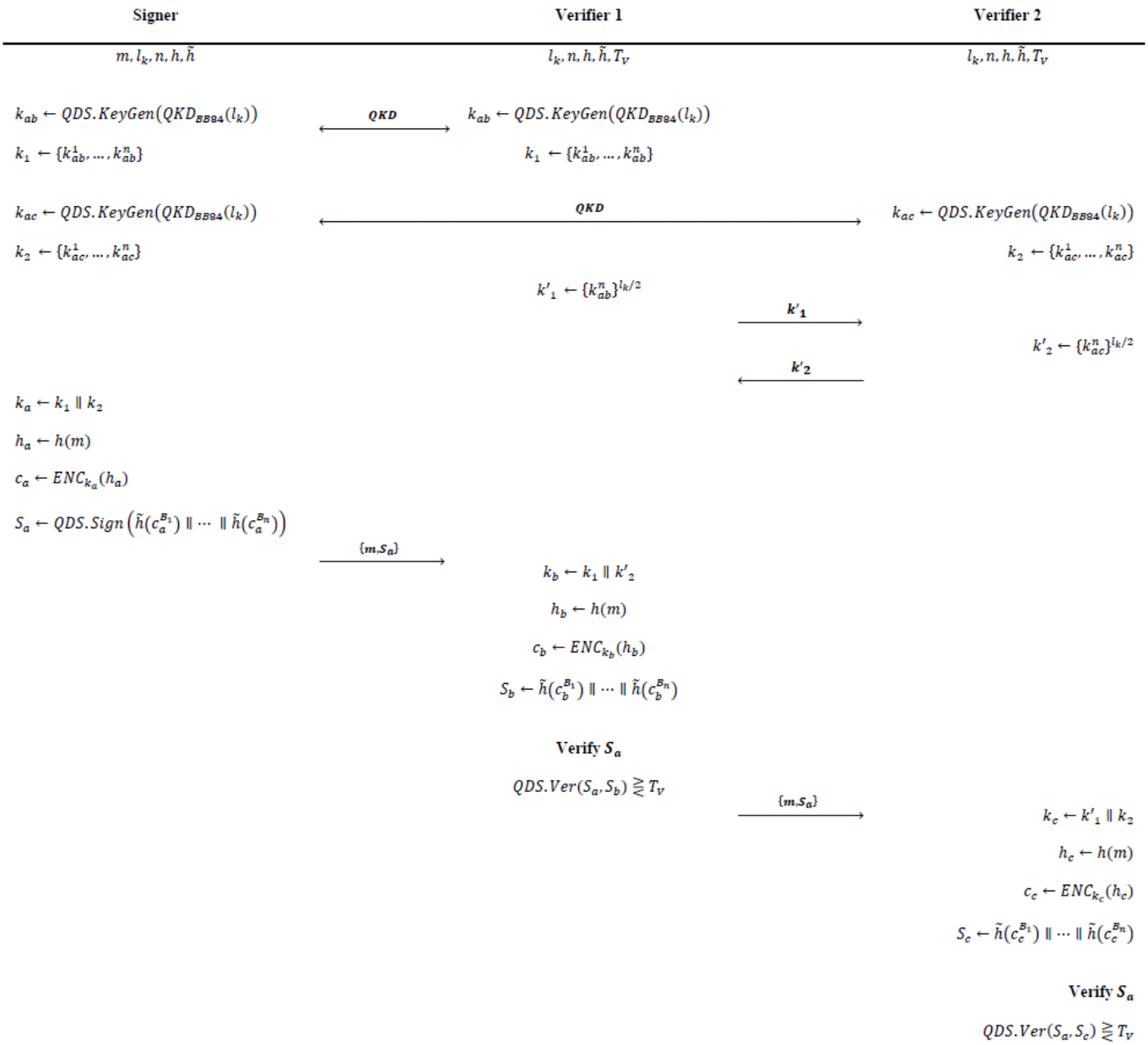

Figure 2. Steps of the Q-DS protocol for both the distribution and the messaging phases.

3. Security Analysis

The security of the proposed Q-DS protocol is analyzed from different perspectives. First, we assess the robustness of the Q-DS protocol against integrity attacks on the message, meaning that if m is modified along the way, it is detected and the protocol aborted. To do so, we assume that Bob is a malicious player. Thus, Alice generates the Q-DS as specified in Section 2. Bob receives (m, S_a) from Alice, and verifies and accepts the signature. After that, he modifies the message $m \rightarrow M$, but keeping the signature generated by Alice S_a. Bob sends to Charlie the tuple (M, S_a). Upon receiving it, Charlie performs the verification process, and the detection of the attack attempt is straightforward, since $S_c \neq S_a$. Modifying the message causes the first alteration in the calculation of the message hash. Since the calculation of S_a and S_c depends on the value of c_a and c_c, respectively, and these depend on the value of h_a and h_c, the alteration produced by the change in message is propagated, triggering an error in Charlie's verification test. Note that in this case, Charlie cannot distinguish whether the reason for the negative verification test was an attack on the integrity of the message or an attempt to forge the signature. In order to succeed in an

integrity attack, Bob would have to be able to find a collision or a second preimage of the hash value. A collision attack means finding any two messages, m and M, with $M \neq m$, such that $h(M) = h(m)$. A preimage attack means that, from the given message digest of m, $h(m)$, finding a different input M that provides the same hash value $h(M) = h(m)$. The difference between collision and a second preimage is that in the later, the malicious user can be unaware of the content of m if, for example, it is encrypted. These attacks can be prevented, as we will see at the end of this section.

Second, we assess the robustness of the Q-DS protocol against a signature forgery attack perpetrated by Bob. In this case, Bob generates a fake digital signature S_f associated with the tampered message M. The generation process of S_f is the same as Alice's, except that Bob does not know around half of Charlie's key k_2, so the key $K \neq k_2$ used to generate k_f will produce an alteration that will propagate through all the steps of the Charlie verification test and will result in a signature rejection. In order to successfully forge the signature, Bob would have to be able to guess the unknown elements of k_2. To do this, Bob could carry out different strategies. (1) Guess the unknown bits of k_2. Due to the randomness of the QKD-generated key, he has a 50% chance of matching the value of each bit. If he has to guess $0.5l_k$ bits, where l_k is the length of k_2, the probability that all his guesses are correct is $P_{guess} = 2^{-l_k/2}$. Therefore, the larger the length of the key, the smaller P_{guess}, as plotted in Figure 3 (left). (2) Extract the unknown elements of k_2 from the signature S_a. Block hashes of a certain size prevent Bob from accessing the full value of c_a and, therefore, the keys from being extracted. (3) Find a preimage, that is, given a hash value of H, identifying the input message m that provides $h(m) = H$. These kind of attacks are mainly performed by brute force but can be prevented by taking into account a series of security parameters and requirements in the choice of hash functions and the input length, as is explained at the end of this section.

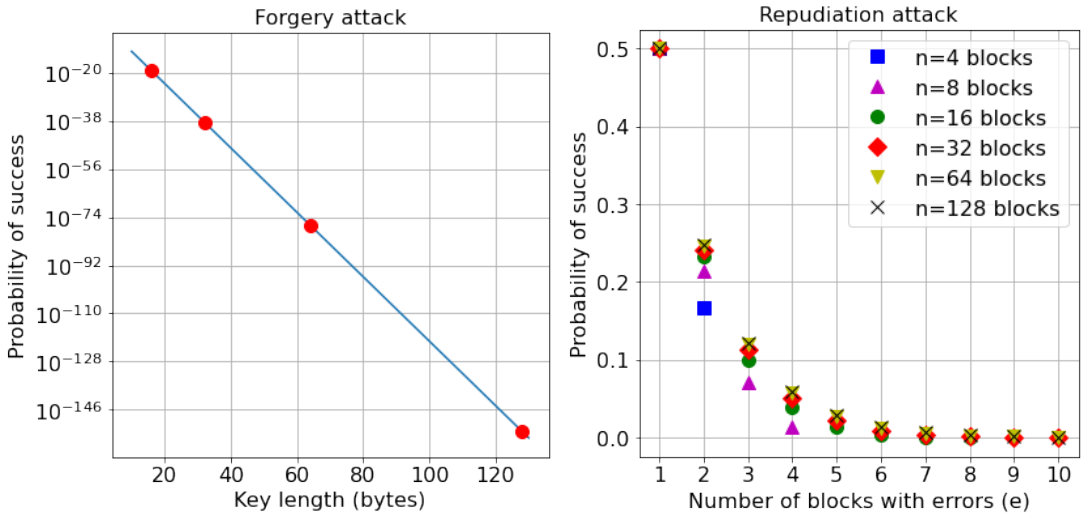

Figure 3. (**Left**) Probability of Bob succeeding in a forgery attack trying to guess the unknown bits of k_2 as a function of the key length, l_k. (**Right**) Probability of Alice succeeding in a repudiation attack as a function of the number of blocks with errors, e, and the configuration of the number of blocks, n.

Finally, we assess the robustness of the Q-DS protocol against a repudiation attack, where we assume that Alice is now the malicious player, whose goal is to generate a signature that Bob accepts but Charlie rejects, thus denying authorship of her message. Alice is aware that Bob only knows half of Charlie's key k_2, but she does not know which elements Bob knows. For Bob to accept the signature and Charlie to reject it, Alice will introduce some errors in random bits of k_2, so that $k_2 \rightarrow K$. For the strategy to be successful, Alice has to ensure that 100% of the errors introduced are in the elements of k_2 that Bob does not know about. Otherwise, Bob will detect

the attempt and reject the signature. The probability of Alice introducing all the errors, e, in the unknown blocks by Bob is given by

$$P_{rep} = \prod_{i=0}^{e-1} \frac{n-2i}{2(n-i)} \tag{2}$$

Then, the more blocks with errors Alice introduces, the lower the probability of success, as plotted in Figure 3 (right). Based on this, Bob and Charlie are able to define thresholds of permissible errors given by $T_V = P_{rep}$, which will determine the maximum admissible number of blocks with errors e for a given n, forcing Alice to introduce a greater number of error and, thus, decreasing her probability of success.

In this final part, we analyze the security strength of the scheme in terms of its cryptographic functions. We emphasize that the protocol is as secure as the most vulnerable of its elements. Both the QKD-generated keys and the OTP encryption are classified, under certain conditions, as ITS [14] and, therefore, conform to perfect secrecy as long as each key is only used once. Therefore, the most vulnerable elements are the hash functions. A priori, the hash functions applied to the message and to the blocks are considered secure as long as one of the recommended hash functions defined in the NIST FIPS 180-4 [15] or NIST FIPS 202 [9] standards is used. The security strength of a hash function is determined by the Collision Resistance (CR), Preimage Resistance (PR) and Second Preimage Resistance (2PR) strengths.

The probability of finding a collision is given by Equation (3) [16], where Δ is the length in bits of the input message, and δ is the length of the message digest. As an example, the amount of work needed to find a collision for SHA2-256 with $CR = 128$ is 2^{128}.

$$P_{col} = 1 - exp\left[-\frac{(2^{\Delta}+1)^2}{2(2^{\delta}+1-2^{\Delta})}\right] \tag{3}$$

As we have seen before, for Bob to successfully carry out a message integrity attack, he has to be able to find a collision or a second preimage of $h(m)$. The amount of work required to find a second preimage is 2^{2PR}.

The most favorable alternative for Bob to forge the signature is to try to find the preimage of each of the unknown key blocks. These types of attacks can be carried out by brute force or using more efficient methods such as rainbow tables. For this reason, we have to find the optimal balance between having as many blocks as possible to reduce P_{rep} and having the size of each block large enough to be secure against a dictionary attack. In this sense, the use of the extendable output function (XOF) SHAKE from the SHA-3 family is proposed. A XOF allows us to securely extend the length of the input up to a predefined δ value in bits. The XOF can be applied to the message for a specific δ_m, as long as $\delta_m = 2 \cdot \delta_k$. If Alice, Bob and Charlie carry out this operation in k_1 and k_2, prior to the block division, they can increase $l_k \rightarrow \delta_k$. As a consequence, the security against dictionary attacks is improved, the efficiency of the protocol is increased (as explained in Section 5) and, furthermore, Bob and Charlie do not exchange plane key blocks but the blocks resulting from the XOF, which increases the privacy of k_1 and k_2.

To establish security levels for the proposed Q-DS scheme comparable with the pre-quantum and PQC algorithms, we employ the measurable computational resources needed to perform the integrity and forgery attacks. In particular, we consider the security in bits of our implementation as the minimum of the security between these two attacks. It is worth noting that we do not directly include repudiation attacks as part of the security level, as it is measured in a probabilistic way. Nevertheless, it is part of the overall security considerations of each parametrization by defining a T_V.

For SHAKE-256, the security strength for CR, PR and 2PR takes the values $CR = min(\delta/2, 256)$, $PR \geq min(\delta, 256)$ and $2PR = min(\delta, 256)$ in bits [9]. From this analysis an example can be derived where SHAKE-256$(m, \delta_m = 2048$ B$)$ is implemented to generate the message hash in the Q-DS protocol, providing a security level against an integrity attack of $CR = min(\delta_m/2, 256) = 256$ bits. We suppose that k_1 and k_2 are $l_k = 1024$ B. For $n = 16$ blocks, the size of each block is 64 B. Finally, SHAKE-256 is also applied to each block on c_a, with $\delta_B = 64$ B, providing a final signature S_a of length 2048 B, and a security level against forgery attack of $PR \geq min(\delta_B = 512$ bits$, 256) = 256$ bits. According to these values, if Bob sets a verification threshold of $T_V = 5\%$ (see Figure 3 (right)), the maximum admissible number of blocks with errors will be $e = 4$, and the overall security level of the Q-DS in this setup will be 256 bits.

4. Implementation

The Q-DS protocol was implemented using *C++* language integration OpenSSL libraries. This study was carried out under the Digest and Sign paradigm, whereby before signing a message, it is hashed, and the message digest is signed afterwards. This is a standard technique used for singing procedures on most applications. The system was built and run in an Oracle VM VirtualBox Ubuntu 64 bit operating system with an 11th Gen Intel(R) Core(TM) i7-1185G7 processor. For the key generation, a pair of commercial discrete variable QKD devices was used in a back-to-back setup, with a security parameter of $\epsilon = 10^{-9}$. The QKD protocol used was a decoy state BB84 composed of one signal, μ_0, and two decoy states, $\{\mu_1, \mu_2\}$, with intensities of $\{0.45, 0.225, 0\}$ and probabilities of $\{0.2, 0.6, 0.2\}$, respectively. The parameters that can be modified in the Q-DS protocol are the message length l_m, hash function applied to the message, key length l_k, number of blocks n, hash function applied to the blocks, and allowed number of blocks with error e. Depending on the parameters chosen, the system will provide a specific security level and the probability of a malicious player succeeding in a repudiation attack, as specified in Section 3.

In the first part of this research, we analyzed the efficiency of the Q-DS protocol during the key generation (*KeyGen*) and signature generation and verification (*SigGen* and *SigVer*) when increasing l_m, l_k and n. The hash function applied to the message and the blocks was *SHAKE256* in all cases.

Subsequently, we carried out a comparative evaluation of the performance of the Q-DS with the most relevant pre-quantum and PQC DSAs. The pre-quantum DSAs that were selected were those most widely implemented in current systems, that is, elliptic curves and Rivest–Shamir–Adleman (RSA). For a security level of 128 bits, the following algorithms were implemented: ECDSA—*BrainpoolP256r1*, *Secp256r1* and EDDSA—*ED25519*. For a security level of 256 bits, the following algorithms were implemented: ECDSA—*BrainpoolP512r1*, *Secp521r1* and *RSA15360*. The PQC digital signature algorithms corresponded to those that are standardized by NIST, which are CRYSTALS-Dilithium under the FIPS 204 standard [4], SPHINCS+ under the FIPS 205 standard [5], and FALCON, whose standard will be published at a later date. The specific algorithms implemented were *Dilithium2*, *Falcon512*, *sphincssha2128{f,s}simple* and *sphincsshake128{f,s}simple*, for a security level of 128 bits. In addition to, *Dilithium5*, *Falcon1024*, *sphincssha2256{f,s}simple* and *sphincsshake256{f,s}simple*, for a security level of 256 bits. The pre-quantum and post-quantum algorithms were benchmarked using the OpenSSL generic interface, and the post-quantum implementation was provided by the Open Quantum Safe (OQS) project through the open-source C library *liboqs* [17]. To compare the different solutions, under an equivalent level of security, common parameters were set in all of them, such as the length of the message ($l_m = 10$ kB), the hash function to calculate the message digest ($SHAKE256(m, \delta_m)$) and the output length of the message

digest ($\delta_m = 512$ B), except for *RSA*, where $\delta_m = 256$ B. The security levels of pre-quantum and post-quantum algorithms were established in terms of the computational resources needed to break the underlying schemes. These levels were detailed by NIST [18,19].

5. Results

To analyze the Q-DS protocol performance, we carried out three experiments. First, the size of the message was increased from $l_m = 1$ B to 1 MB, and each of them was executed 100.000 times. The rest of the parameters were set as follows: output message and block digest of $\delta_m = 512$ B and $\delta_B = 64$ B, respectively; $n = 16$ blocks; and a key length of $l_k = 256$ B. The resulting signature lengths for these setups are always $L_S = 2048$ B. The results obtained are shown in Figure 4 (left). The protocol performance during the *SigGen* and *SigVer* processes remains constant, taking an average value of 23 µs up to message sizes of 1 kB. Above this value, the average *SigGen* and *SigVer* times increase exponentially, although the values reached are in the order of 2 ms when $l_m = 1$ MB. These setups provide a security level of 128 bits.

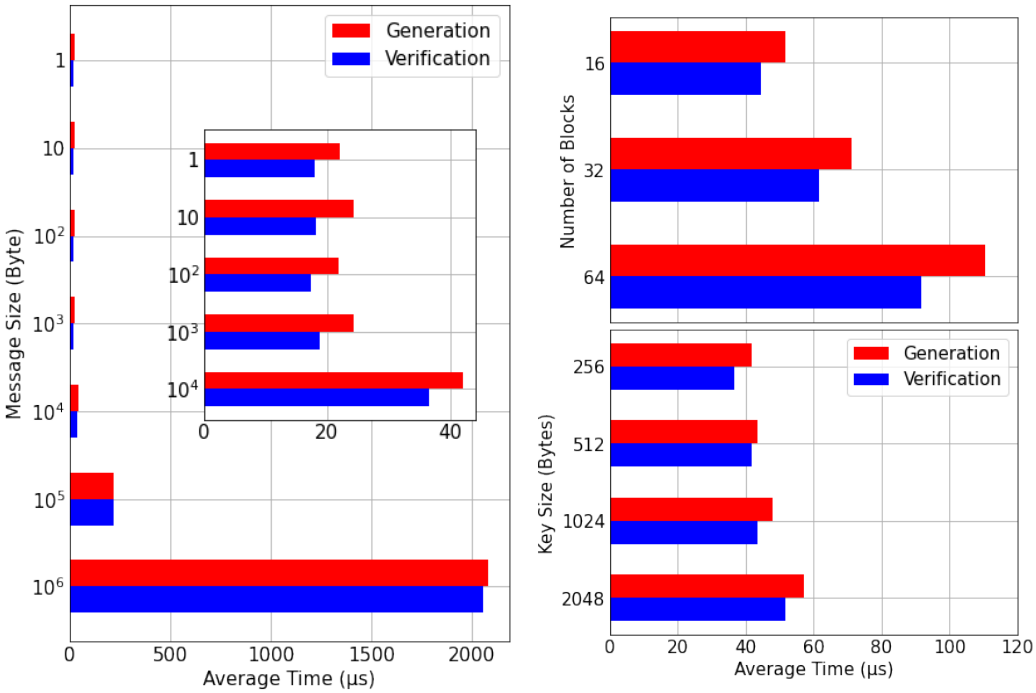

Figure 4. (**Left**) The average generation and verification times in µs with an increase in the message size in Bytes, (**top right**) the number of blocks and (**bottom right**) the key size in Bytes.

A second analysis was conducted that consisted of analyzing the performance when increasing n from 16 to 64 for a given message of $l_m = 10$ kB, an output message digest of $\delta_m = 2048$ B, a block digest of $\delta_B = 64$ B and a key length of $l_k = 1024$ B. In this case, these setups provide a security level of 256 bits for $n = \{16, 32\}$, and 128 bits for $n = 64$. An increase in n leads to a greater number of hashes/XOF calculations, giving rise to a linear increase in the execution time, as shown in Figure 4 (right/top), and in the final signature length. For a division of 16 blocks, the average *SigGen* (*SigVer*) time is 51.7 µs (44.5 µs), while for $n = 64$, the average *SigGen* (*SigVer*) time increases to 110.6 µs (91.6 µs).

Thirdly, we analyzed the performance as a function of the key length and the related length of the message digest. Here, the message was fixed to $l_m = 10$ kB, $n = 16$, and $\delta_B = 64$ B. The key length was increased from $l_k = 256$ B (security level 128 bits) to 2048 B (security level 256 bits). The resulting signature lengths for these setups were always $L_S = 2048$ B. As shown in Figure 4 (right/bottom), increasing the size of the keys does

not have a significant impact on the signature generation and verification times, yielding 41.8 µs when $l_k = 256$ B and 57.3 µs when $l_k = 2048$ B. This behavior is expected because during the generation and verification of the signature, the key size increase only impacts the OTP calculation, which is a simple, low-consumption operation. However, the increase in l_k will have a greater impact on the *KeyGen* process. A key generation rate of 9.3 kbps was obtained from the QKD devices in a back-to-back setup, taking 221.7 ms to generate 257 B of key. This rate decreases as the distance between the sender and the receiver increases, due to the losses present in quantum channels.

Finally, to carry out a comparative evaluation of the Q-DS, and the pre-quantum and PQC DSAs, the Q-DS protocol was set at $\{l_k = 256\text{ B}, n = 16, \delta_B = 64\text{ B}, \delta_m = 512\text{ B}\}$ for a security level of 128 bits and at $\{l_k = 512\text{ B}, n = 16, \delta_B = 64\text{ B}, \delta_m = 1024\text{ B}\}$ for a security level of 256 bits. All the results obtained are shown in Figure 5.

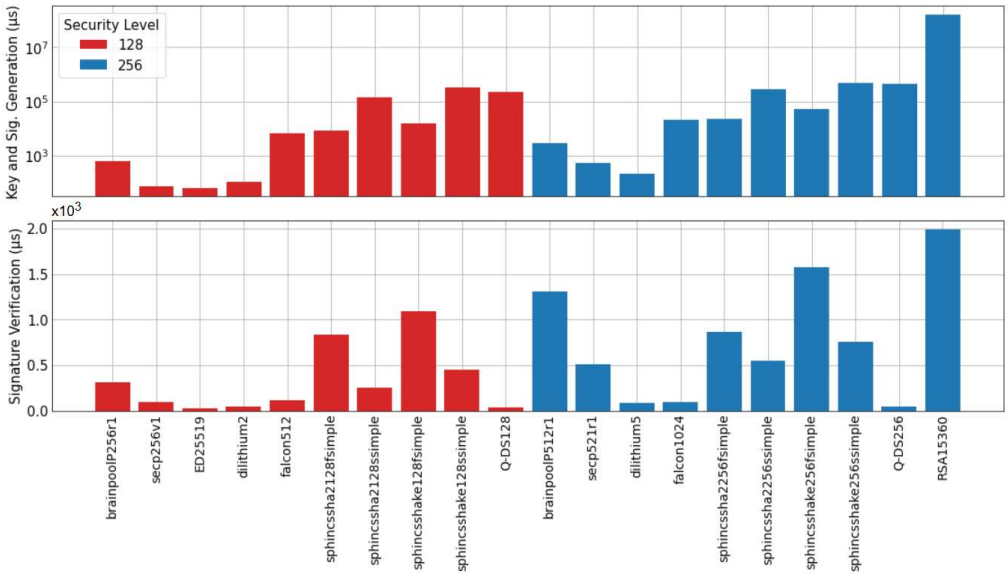

Figure 5. The average time in µs for the (**top**) key and signature generation and (**bottom**) signature verification of all pre-quantum, PQC and Q-DS digital signatures. All the solutions are grouped by their security levels, 128 bits (red) and 256 bits (blue).

The top graph presents the average time in µs that it takes for the key generation plus signature generation processes to complete for each algorithm on logarithmic scale. For a security level of 256 bits, RSA is the algorithm that shows the worst performance, i.e., $1.6 \cdot 10^8$ µs, despite its favorable δ_m configuration. Within the pre-quantum algorithms, the most efficient is *secp521r1*, with an average generation time of 522.9 µs. In general, the out-performer is the PQC DSA *Dilithium5* with an average time of 211.5 µs, while FALCON and SPHINCS+ present comparable performance. The Q-DS average generation time is $2.2 \cdot 10^5$µs, similar to the "small" variants of SPHINCS+ *sphincssha256ssimple* and *sphincsshake256ssimple*. This decrease in the performance of Q-DS in comparison with the results shown in Figure 4 is due to the cost of the QKD process. The efficiency of the *KeyGen* process can be drastically increased with other QKD solutions such as Twin-Field QKD [20], or by generating keys with $l_k = 256$ bits and expanding their size by means of an XOF. For a security level of 128 bits, the order of the results is similar to the previous case, as shown in Figure 5. Q-DS and PQC FALCON and SPHINCS+ present lower efficiency, while PQC DSA *Dilithium2* stands out. In this case, the pre-quantum algorithms *secp256r1* and *ED25519* show generation times of less than 100µs, comparable to *Dilithium2*.

The signature verification process is, in general, the most efficient process for all algorithms, as can be seen in the bottom graph of Figure 5. For a security level of 256 bits,

in the case of the Q-DS protocol, the average time for the signature verification process is 41.7 µs, which is almost equivalent to the signature generation due to the symmetry in the steps that are executed in both processes. In this case, Q-DS is the out-performer solution. For PQC algorithms, it can be seen that the "fast" variants *sphincssha256fsimple* and *sphincsshake256fsimple* of *SPHINCS+* present less efficiency than the other variants of this family, with average times of 862.5 and 1517.4 µs, respectively. The PQC out-performer algorithms in this case are *dilithium5* and *falcon1024*, providing verification times of 86.7 µs and 95.7 µs, respectively. Within the pre-quantum solutions, the most efficient algorithm is *secp521r1*, i.e., 507.5 µs which is a similar execution time to that of *sphincssha256ssimple*. As in the previous case, *RSA15360* is the less efficient algorithm. The results of this analysis are similar for those algorithms with a security level of 128 bits.

6. Conclusions

The quantum-assisted digital signature protocol proposed in this article presents an alternate to the use of currently employed pre-quantum asymmetric cryptosystems (RSA, ECDSA, etc.), which are considered vulnerable against a CRQC implementing Shor. The Q-DS protocol replaces the key exchange with ITS QKD-generated keys for which commercial devices are available. Unlike the QDS protocols published to date, our proposal provides a composite approach that allows us to sign messages with an arbitrary length with high efficiency. This is accomplished by taking into account NIST-recommended hash functions from the SHA-2 and SHA-3 families, and combining them with QKD-generated keys.

The protocol is demonstrated to be secure against attacks on the integrity of the message, forgery of the signature and attempts at repudiation by the signer. In addition, it is highly parametrizable in terms of the message length, hash function applied to the message, key length, number of blocks and hash function applied to the blocks, allowing us to configure digital signatures with different levels of security and with minimal impact on the efficiency of the signature generation and verification.

The implementation of the proposed Q-DS together with 6 pre-quantum DSA and 12 PQC DSA, for two different levels of security (128 and 256 bits) allowed us to conduct a comparative evaluation in terms of the efficiency of the different solutions during the key generation, signature generation and verification. It is observed that, for the same level of security, the Q-DS protocol is the most efficient solution during the generation and verification of the signature, only surpassed by the pre-quantum *ED25519* for a security of 128 bits.

The greatest impact on the efficiency of the protocol is observed in the generation of QKD keys, which results in a key generation rate of 9.3 kbps in a back-to-back configuration. However, other commercial devices and QKD protocols available can further improve the *KeyGen* process of the Q-DS, making it comparable to the most efficient PQC DSA *DILITHIUM*.

Finally, it should be noted that this type of analysis aims to facilitate the selection of possible candidates during the migration processes from current cryptographic systems to quantum-resistant ones, given the needs of a specific environment. Regarding future directions for this research, three main ones have been identified. First, we propose an analysis of the protocol for multiple receivers (i.e., more than two possible verifiers in Q-DS), as an increase in the number of possible verifiers leads to an increase in the number of quantum communication links to perform QKD and has an impact on the security proof due to an increase in the attack surface or the possibility of collective attacks. Second, we propose testing the protocol in a more realistic environment supported by deployed quantum communication infrastructures covering long distances. Third, we

suggest improving the performance during the key generation process by analyzing other approaches to performing QKD.

Author Contributions: Conceptualization, M.I.G.-C. and D.D.; software, R.M.; validation, M.I.G.-C. and L.O.; formal analysis, M.I.G.-C.; investigation, M.I.G.-C.; writing—original draft preparation, M.I.G.-C. and L.O.; writing—review and editing, M.I.G.-C., V.M. and L.O.; supervision, V.M. and L.O.; project administration, D.D. and L.O. All authors have read and agreed to the published version of the manuscript.

Funding: This project received funding from EIT Digital, co-funded by the European Institute of Innovation and Technology (EIT), a body of the European Union, and was performed in the scope of the PQ-REACT European Research Project, supported by the Commission of the European Communities HORIZON, Grant Agreement No. 101119547.

Data Availability Statement: All data generated and analyzed are presented in the main text. No additional data were generated or analyzed in the presented research.

Conflicts of Interest: Authors Marta Irene García-Cid, Rodrigo Martín and David Domingo were employed by the companies Indra Sistemas S.A. and Indra Sistemas de Comunicaciones Seguras. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Shor, P.W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.* **1997**, *26*, 1484–1509. [CrossRef]
2. Bennett, C.H.; Brassard, G. Public key distribution and coin tossing. In Proceedings of the IEEE International Conference on Computers, Systems and Signal Processing, Bangalore, India, 10–12 December 1984; pp. 175–179.
3. National Institute of Standards and Technology. Post-Quantum Cryptography. Available online: https://csrc.nist.gov/projects/post-quantum-cryptography (accessed on 20 January 2025).
4. *FIPS 204 Standard*; Module-Lattice-Based Digital Signature Standard. NIST: Gaithersburg, MD, USA, 2023. Available online: https://csrc.nist.gov/pubs/fips/204/final (accessed on 20 January 2025).
5. *FIPS 205 Standard*; StatelesshHash-Based Digital Signature Standardd. NIST: Gaithersburg, MD, USA, 2023. Available online: https://csrc.nist.gov/pubs/fips/205/final (accessed on 20 January 2025).
6. Martin, V.; Brito, J.P.; Ortiz, L.; Mendez, R.B.; Buruaga, J.S.; Vicente, R.J.; Sebastián-Lombraña, A.; Rincon, D.; Perez, F.; Sanchez, C.; et al. MadQCI: A heterogeneous and scalable SDN QKD network deployed in production facilities. *Npj Quantum Inf.* **2024**, *10*, 80. [CrossRef]
7. Federal Office for Information Security (BSI). *A Study on Implementation Attacks Against QKD Systems*; Federal Office for Information Security (BSI): Bonn, Germany, 2023; Project number: 575.
8. *NIST SP 1800-38C*; Migration to Post-Quantum Cryptography Quantum Readiness: Testing Draft Standards. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2023.
9. *FIPS PUB 202*; SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions. National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015.
10. Gottesman, D.; Chuang, I. Quantum digital signatures. *arXiv* **2001**, arXiv:quant-ph/0105032.
11. Vedran, D.; Wallden, P.; Andersson, E. Quantum digital signatures without quantum memory. *Phys. Rev. Lett.* **2014**, *112*, 040502.
12. Amiri, H.-R.; Wallden, P.; Kent, A.; Andersson, E. Secure quantum signatures using insecure quantum channels. *Phys. Rev. A* **2016**, *93*, 032325. [CrossRef]
13. Lu, Y.S.; Cao, X.Y.; Weng, C.X.; Gu, J.; Xie, Y.M.; Zhou, M.G.; Yin, H.L.; Chen, Z.B. Efficient quantum digital signatures without symmetrization step. *Opt. Express* **2021**, *29*, 10162–10171. [CrossRef] [PubMed]
14. Shor, P.W.; Preskill, J. Simple proof of security of the BB84 quantum key distribution protocol. *Phys. Rev. Lett.* **2000**, *85*, 441. [CrossRef] [PubMed]
15. *FIPS PUB 180-4*; Secure Hash Standard (SHS). National Institute of Standards and Technology: Gaithersburg, MD, USA, 2015.
16. Peyravian, M.; Roginsky, A.; Kshemkalyani, A. On probabilities of hash value matches. *Comput. Secur.* **1998**, *17*, 171–176. [CrossRef]
17. OQS Project. Available online: https://openquantumsafe.org/liboqs/ (accessed on 5 June 2024).

18. NIST. Submission Requirements and Evaluation Criteria for the Post-Quantum Cryptography Standardization Process. Available online: https://csrc.nist.gov/CSRC/media/Projects/Post-Quantum-Cryptography/documents/call-for-proposals-final-dec-2016.pdf (accessed on 5 June 2024).
19. *FIPS 186-5*; Digital Signature Standard (DSS). NIST: Gaithersburg, MD, USA, 2023.
20. Wang, S.; Yin, Z.-Q.; He, D.-Y.; Chen, W.; Wang, R.-Q.; Ye, P.; Zhou, Y.; Fan-Yuan, G.-J.; Wang, F.-X.; Chen, W.; et al. Twin-field quantum key distribution over 830-km fibre. *Nat. Photonics* **2022**, *16*, 154–161. [CrossRef]