Computer Communications 213 (2024) 345–358

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Quantum-resistant Transport Layer Security

Carlos Rubio García [a,*], Simon Rommel [a], Sofiane Takarabt [b], Juan Jose Vegas Olmos [c], Sylvain Guilley [b], Philippe Nguyen [b], Idelfonso Tafur Monroy [a]

[a] Department of Electrical Engineering, Eindhoven University of Technology, Eindhoven, Netherlands
[b] Cybersecurity Solutions, Secure-IC, Cesson-Sévigné, France
[c] Software Architecture, NVIDIA Corporation, Yokneam, Israel

ARTICLE INFO

Keywords:
Quantum-resistant cryptography
Quantum key distribution
Post-quantum cryptography
Transport layer security
Cybersecurity

ABSTRACT

The reliance on asymmetric public key cryptography (PKC) and symmetric encryption for cyber-security in current telecommunication networks is threatened by the emergence of powerful quantum computing technology. This is due to the ability of quantum computers to efficiently solve problems such as factorization or discrete logarithms, which are the basis for classical PKC schemes. Thus, the assumption that communications networks are secure no longer holds true. Quantum Key Distribution (QKD) and post-quantum cryptography (PQC) are the first cyber-security technologies that allow communications to resist the attacks of a quantum computer. To achieve quantum-resistant communications, the aforementioned technologies need to be incorporated into a network security protocol such as Transport Layer Security (TLS). In this paper, we describe and implement two novel, hybrid solutions in which QKD and PQC are combined inside TLS for achieving quantum-resistant authenticated key exchange: Concatenation and Exclusively-OR (XOR). We present the results, in terms of complexity and security enhancement, of integrating state-of-the-art QKD and PQC technologies into a practical, industry-ready TLS implementation. Our findings demonstrate that the adoption of a PQC-only approach enhances the TLS handshake performance by approximately 9 % compared to classical methods. Furthermore, our hybrid PQC-QKD quantum-resistant TLS comes at a performance cost of approximately 117 % during the key establishment process. In return, we substantially augment the security of the handshake, paving the road for the development of future-proof quantum-resistant communication systems based on QKD and PQC.

1. Introduction

Quantum-resistant cryptography and communications have emerged as the response to withstand the potential security attacks by an adversary using quantum computers. Although the time-frame for such a threat to materialize is still unknown, there are already major concerns in regard to cybercrime and privacy because, for most Internet users, information is transported over public infrastructures handled by third parties. Hence, extremely sensitive data is often exposed to vulnerable communication channels.

Currently, the security of critical information is provided primarily by a mix of asymmetric public key cryptography (PKC) and symmetric encryption schemes. PKC schemes include Rivest–Shamir–Adleman (RSA), Elliptic-curve Diffie–Hellman (ECDH) or the Digital Signature Algorithm (DSA), while a predominantly employed symmetric encryption algorithm is the Advanced Encryption Standard (AES) [1–5]. In essence, PKC schemes agree upon and share a session key, which is later on used for symmetric encryption, a technique still considered to be quantum-resistant [6].

The security guarantees of PKC rely on the computational difficulty associated with solving specific mathematical problems (e.g., factorization, discrete logarithms) within a reasonable amount of time. However, since the publication of Shor's algorithm [7,8], it is well known that the development of powerful-enough quantum computers can compromise classical PKC schemes, making current data communications vulnerable. In addition, the development of such systems might not be very distant: massive investments have been launched across the world and scenarios where modern encryption systems are affected by quantum attacks are relatively likely to occur within the next decades. Indeed, the National Institute of Standards and Technology (NIST) estimates, supported by [9], that there is a 50% chance that a quantum computer capable of breaking RSA-2048 will be built by 2031 [10].

Therefore, the vulnerabilities of classical cryptography in regards to quantum attacks require immediate attention. With strategies such as

* Corresponding author.
E-mail address: c.rubio.garcia@tue.nl (C. Rubio García).

https://doi.org/10.1016/j.comcom.2023.11.010
Received 17 July 2023; Received in revised form 26 October 2023; Accepted 10 November 2023
Available online 11 November 2023

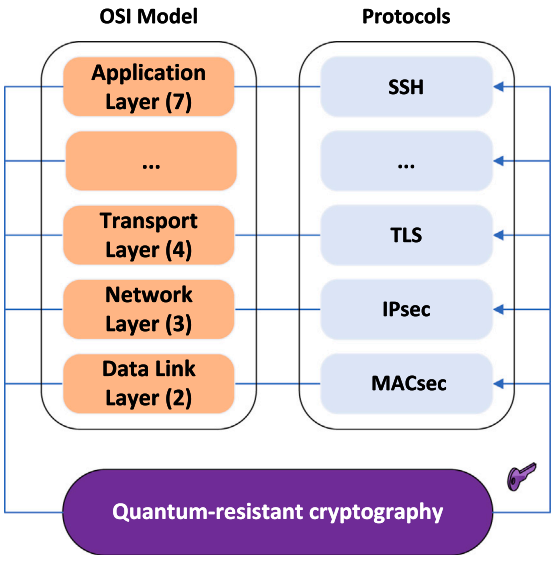

Fig. 1. Integration of quantum-resistant cryptography at different network layers of the open systems interconnection (OSI) model [12].

the "harvest now, decrypt later attacks", adversaries can already collect encrypted versions of long-lived data that will still be significant in the future. This information can then be stored until a quantum computer capable of breaking its encryption is available [11]. In such a scenario, an adversary can decrypt the cipher text and gain access to the private data. This results in a critical need for implementing quantum-resistant communications urgently.

Quantum key distribution (QKD) and post-quantum cryptography (PQC) are the first quantum-resistant technologies capable of withstanding the attacks of a quantum computer. By modifying the foundation of the underlying cryptographic operations, QKD and PQC mitigate the computational advantages conferred by Shor's algorithm.

For the implementation of quantum-resistant communications, both QKD and PQC need to be integrated into a network security protocol of choice. Fig. 1 illustrates the different layers existing in current networks as per the OSI model, as well as the associated mechanisms to ensure confidential communications, all of them based on classical cryptography [12]. As depicted in Fig. 1, transport layer security (TLS) is one such network security protocols whose security can be augmented via quantum-resistant technologies [6,13]. It is used in the transport layer, serves as the standard client–server protocol for securing communications over the Internet, and it is being proposed to secure the control and management communication for emerging software-defined networking (SDN) in 5G and 6G networks [14–17].

Nonetheless, none of the works seen so far combine PQC and QKD at different stages of the TLS protocol (authentication and key exchange) for enhancing the security of TLS, nor provide practical measurements of the performance constraints and communication cost of jointly integrating QKD and PQC in TLS.

In this paper, we present our hybrid quantum-resistant TLS protocol enabled by both QKD and PQC. We explore a quantum-resistant TLS protocol in which QKD and PQC can complement each other for achieving enhanced security. We also aim to analyze the performance implications of employing the aforesaid solution in an experimental networking scenario. Our contributions can be summarized as follows:

- To the best of the author's knowledge, we present the first experimental demonstration in which fully commercially available QKD devices are integrated together with PQC algorithms selected by NIST (CRYSTALS-Dilithium and CRYSTALS-Kyber) into an industry-ready TLS implementation (Mbed TLS).
- We propose and analyze two novel mechanisms in which PQC-based and QKD-based secret keys can be combined for augmenting the security of the TLS handshake.
- We evaluate, in terms of communication cost and security enhancement, the practicality of integrating our proposed hybrid quantum-resistant TLS protocol based on QKD and PQC in an experimental network scenario.
- Finally, we conclude that the adoption of our proposed quantum-resistant TLS protocol comes at a considerable communication cost. In exchange, we significantly augment the security of the communications, now protected by two different quantum-resistant cryptographic assumptions.

The remainder of the paper is structured as follows. Section 2 gives an overview of QKD and PQC and related work on quantum-resistant communications. Section 3 addresses the transition from classical to quantum-resistant TLS via the aforementioned technologies. Section 4 introduces our proposed quantum-resistant TLS architecture and illustrates two ways in which QKD and PQC can be combined. Section 5 presents the obtained results and the corresponding analysis. Finally, Section 6 summarizes and concludes the article.

2. Background and state of the art

The quantum computing threat has recently raised the need for current networks to transition towards implementing quantum-resistant communications. In this regard, there are industry, academic and governmental projects that are pushing towards the adoption of both QKD and PQC within network security protocols such as TLS.

2.1. Quantum Key Distribution

QKD is a cryptographic technique that aims at the secure distillation of a secret key as a result of successfully transmitting quantum signals among authenticated partners [18]. In any given QKD scenario, there are at least two authorized partners (transmitter and receiver), connected via two channels. First, a quantum channel that is used to transmit the quantum signals that are later on used to distill a secret key. In addition, an authenticated service channel is used for both key distillation and post-processing. The authenticated channel allows both the transmitter and receiver to perform tasks such as error correction and privacy amplification on the key. The calculated quantum bit error rate (QBER) can be used to accurately estimate the amount of information leaked during the exchange, and allows transmitter and receiver to limit this leakage as much as desired [19].

While classical and PQC cryptography algorithms rely on the mathematical complexity of solving a given problem (e.g., factorization, discrete logarithms, shortest vector problem, etc.), the security of QKD relies on two theorems of quantum physics. The no-cloning theorem [20] and Heisenberg's uncertainty principle [21]. The first states that it is impossible to obtain an identical copy of an arbitrary unknown quantum state. The second explains that the measurement of a quantum signal causes a perturbation in its state that cannot be recovered. Hence, any attempt to eavesdrop in the quantum channel will inevitably alter the state of the quantum signals exchanged, alerting the involved parties of the presence of an eavesdropper.

After the transmission of the quantum signals has happened, both the transmitter (Alice) and receiver (Bob) can execute a series of post-processing operations in which it is estimated how much information about the quantum signals has been leaked to an outside party. If the post-processing operations reveal the presence of an eavesdropper on the quantum channel during the transmission of quantum signals, Alice and Bob abort the protocol. Otherwise, Alice and Bob can distill a key whose security is guaranteed by the laws of quantum physics [18].

Upon completing the process, the resulting shared keys are stored symmetrically in both the transmitter and receiver's key management

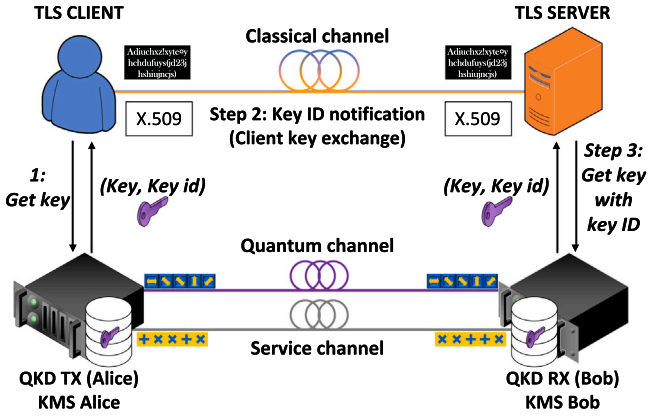

Fig. 2. Integration of QKD into classical network security protocols (e.g., TLS): Architecture overview.

systems (KMS), from which both client and server applications can retrieve the same QKD-based key via the application programming interface (API) defined in [22]. Guaranteeing the security of this interface is essential for the correct implementation of a QKD system. In this regard, the secure application entities (SAE) requesting keys must be located within the same secure point-of-presence as its corresponding QKD nodes, thus providing the key delivery API with an extra intrinsic security [22]. The QKD key retrieval API is used to retrieve both a base64 encoded key and its corresponding key identifier (key ID). The key ID is a 36-bytes universally unique identifier (UUID) that must be transmitted through a classical communication channel.

Fig. 2 presents a simplified architectural overview highlighting the essential elements required for integrating QKD into a network security protocol. Such a system encompasses at least a pair of QKD modules, key management servers, and secure application entities (e.g., TLS client and TLS server). Fig. 2 also highlights three main steps that allow the same quantum key to be obtained at both endpoints of the communication: (1) key retrieval on client's side; (2) key id notification; and (3) key retrieval on server's side. These steps are common to any network security protocol, including TLS, prospect to use QKD-based keys. Upon the successful completion of the process, the resultant QKD-based key must be used as a common, shared secret that allows it to perform symmetric encryption. If implemented properly, QKD is an information-theoretic secure (ITS) primitive independent of any future algorithmic advance, including the development of machines with unbounded computing power [23].

It must be noted that, in order to take full advantage of the ITS primitive that QKD offers, it must be used in combination with an ITS symmetric encryption primitive, such as the one-time pad (OTP) [24]. While OTP encryption is theoretically secure, it has practical limitations for use in current communications. OTP encryption requires a truly random secret key that must be at least as long as the message itself. In addition, to provide statistical secrecy, session keys can be used only once for encryption. The key generation process required by OTP introduces significant time overhead before a message can be encrypted. Therefore, OTP is suitable for scenarios where short messages with high security requirements are encrypted, but is less practical than other symmetric encryption algorithms when encrypting long messages that would require equally long keys. Nonetheless, although the combination of QKD with a quantum-resistant authenticated encryption with associated data (AEAD) encryption algorithm (e.g., AES-256 GCM) does not provide ITS security, it is still resistant to quantum computers [25].

The main challenge for QKD lies in the existing trade-off between secret key rate (SKR) generation and distance among QKD nodes. As the distance and corresponding optical fiber channel loss increase, the SKR decays exponentially [26]. In this regard, significant progress is being made towards overcoming the SKR-distance trade-off limitations of QKD [26,27]. Another significant challenge for QKD is the implementation of a remote key delivery API. Although remote key delivery is a desirable feature that would largely improve the usability of QKD keys, it would require the secure delivery of the key between the SAE and QKD node to be done via classical or post-quantum cryptography. However, this would undermine the security of the system as the key will already be exposed via a 'weaker' cryptographic algorithm. Therefore, the current standard [22] determines that both the QKD node and requesting SAE must be in the same point-of-presence.

2.2. Post-quantum cryptography

PQC is a set of novel cryptographic algorithms that remains secure under the assumption that an attacker has access to a fault-tolerant quantum computer. Unlike QKD, PQC maintains the same functional principles seen in classical asymmetric cryptography, such as key-pair generation, digital signatures, key-establishment or key encapsulation/decapsulation [28]. While the mathematical problems that are the base for classical cryptography (e.g., large factorization and discrete logarithms) have been proven insecure against quantum computers [8], PQC has identified and based its security on mathematical operations such as lattice-based or hash-based cryptography, for which quantum algorithms offer little to no advantage in speed [29].

NIST has recently conducted a contest for quantum-resistant alternatives to current pre-quantum cryptography. The algorithms that entered the competition were designed for two main tasks for which PKC is normally used. General public-key encryption, used to protect information that is exchanged through a public network, and digital signatures, used for identity authentication and verification of message authenticity among other purposes. As a result, in July 2022, NIST announced the winning algorithms of the competition. For both public-key encryption and digital signatures, NIST has selected algorithms of the 'Cryptographic Suite for Algebraic Lattices' (CRYSTALS) family. CRYSTALS security is based on hard problems over module lattices and is designed to withstand attacks by large quantum computers. Regarding digital signatures, NIST has selected three different algorithms, CRYSTALS-Dilithium, FALCON and SPHINCS+ (read as 'Sphincs plus') and recommends using CRYSTALS-Dilithium as the primary digital signature algorithm. For public-key encryption, NIST has selected CRYSTALS-Kyber,a key-encapsulation mechanism (KEM) that is secure against indistinguishably under adaptive chosen-ciphertext attacks (IND-CCA2). This selection was due to its optimal encryption key size and the efficiency of its operations, crucial features for ensuring secure and fast encryption [30–32].

The central challenge for some post-quantum algorithms is to meet the demands for usability and scalability of systems while remaining safe against quantum computers. Longer key and signature required by different PQC algorithms might give rise to performance and scalability issues that might demand modifications to protocols and infrastructures [28,33]. Given that PQC represents a novel area in cryptography with ongoing research efforts, there is a risk that algorithms thought to be quantum-resistant might become vulnerable to future algorithmic breakthroughs [34].

As an example, in February 2022, Ward et al. [35] introduced a key recovery attack against Rainbow [36], one of the three third round finalists of the NIST PQC standardization project for digital signatures. The authors claim that breaking Rainbow takes approximately 53 h with a laptop. Moreover, regarding others PQC-based algorithms involved in the NIST standardization project, Castryck et al. [37] presented an efficient key recovery attack on the Supersingular Isogeny Diffie–Hellman protocol (SIDH). These findings make evident the increasing risks associated with only implementing PQC-based algorithms whose security might be challenged by emerging cryptanalytic attacks.

Table 1
Related work for quantum-resistant TLS.

Reference	Year	Methodology		Experimental integration in TLS	Results analysis and evaluation
		Authentication	AKE		
Paquin et al. [38]	2020	PQC	PQC	Yes	Yes
Sikeridis et al. [39]	2020	PQC	PQC	Yes	Yes
Tasopoulos et al. [40]	2022	PQC	PQC	Yes	Yes
Mink et al. [41]	2010	–	QKD	No	No
Buruaga et al. [42]	2023	–	QKD	Yes	No
Huberman et al. [43]	2020	–	QKD	Yes	No
Dowling et al. [44]	2020	QKD (pre-shared keys)	PQC	No	No
Huang et al. [45]	2020	PQC	Hybrid PQC-QKD	No	No
Shim et al. [46]	2022	–	Combined PQC-QKD	No	No
Giron et al. [47]	2021	–	Hybrid PQC-QKD	No	No
Our work	2023	**PQC**	**Hybrid PQC-QKD**	**Yes**	**Yes**

2.3. Related work

In the context of quantum-resistant communications based on QKD and PQC, numerous studies have emerged for finding resilient alternatives to classical authentication and key exchange mechanisms inside TLS. These studies are summarized in Table 1.

Paquin et al. [38] presented a hybrid classical-PQ solution for securing TLS connections within an emulated network scenario developed via the Linux kernel. The analysis presented determined that the TLS handshake completion time was mainly dominated by the cost of both classical and PQ PKC. It was also mentioned that unreliable networks are affected by maximum transmission unit (MTU) size, normally set at 1500 bytes, which is an issue to take into account when implementing PQ primitives with large message sizes. This translated into an increase in the number of network packets exchanged per handshake.

In the same line, Sikeridis et al. [39] assessed the overhead of PQ cryptography on both TLS and the secure shell (SSH) protocols. Dilithium and SPHINCS+ were tested for digital signatures, and Kyber and NewHope were tested for the PQ key exchange. RSA 2048 and ECDH with curve SECP256 were used as a control group for benchmarking PQ-based solutions against classical TLS. The experiment concluded that a hybrid PQ TLS handshake combining SECP256 ECDH with AVX2-optimized kyber512 for the key exchange had a minimal impact on the handshake duration, with approximately ≈1 ms of added latency. Finally, the impact of TCP initial congestion window on the overall handshake duration is explained.

The work presented by Tasopoulos et al. [40] explored the implementation of PQ cryptography on resource-constrained devices. The implementation was based on WolfSSL and evaluated the adoption of PQ cryptography in terms of execution speed, memory requirements and size of communication packets. In the study, it is acknowledged that although the performance of certain PQ algorithms might be better than others, bandwidth increases considerably with stronger algorithms.

Mink et al. [41] presented a study on both QKD and the network security protocols in which QKD can be integrated, such as TLS or Internet protocol security (IPsec). The work stated that quantum/resistant public key cryptography (commonly known as PQC) are based on security assumptions of unproven computational complexity, and if those assumptions are shown to be false, the algorithms would be insecure. Thus, incorporating QKD into such protocols is the solution that provides more confidence for the future. To conclude the study, the authors theoretically explained how the QKD-based secret can be integrated into TLS via the pre-master or master secrets, and what changes would be necessary to make.

Buruaga et al. [42] recently described a mechanism to integrate QKD with open-source interfaces, so that the integration of QKD in a network environment will no longer rely on vendor specific implementations. To this end, the authors in [42] investigated the integration of QKD-derived keys into virtual private networks (VPN) technologies via the OpenVPN library, which uses TLS as a mechanism to encrypt information. A methodology for integrating QKD-derived keys into TLS and OpenVPN. However, no results or practical implementation are shown.

Hubermann et al. [43] also presented a mechanism to integrate QKD into TLS as an alternative for adopting quantum-resistant cryptography, thus allowing quantum-secure internet services. The strategy shown aimed to map the QKD key material as the pre-shared key (PSK) for creating a TLS session. It was also claimed in the paper that through the ETSI QKD key retrieval API, the available on-demand QKD keys do not incur in additional latency for performing a TLS handshake. While Hubermann et al. used the same equipment as we do, our practical experiments presented in Section 5 show otherwise. Finally, a proof-of-concept was shown in this work, in which a QKD-TLS tunnel is implemented via NodeJS v13.6 and ID Quantique Clavis 3 QKD equipment. Nonetheless, no experimental results, analysis or evaluation of the impact of integrating QKD in TLS was shown.

Regarding implementations that take into account the combination of PQC and QKD for hybrid key exchange, Dowling et al. [44] suggested a mechanism for integrating various cryptographic methods – including classical, post-quantum (PQ) cryptography, and QKD – to to achieve quantum-secure hybrid key exchanges. The presented work remarks the importance of building quantum-resistant protocols that can protect both against catastrophic failures and future cryptanalytic attacks, as NIST has already ruled out 13 of the original submissions due to lack of confidence. Furthermore, the authors explained that QKD is an alternate solution that offers unconditional security and because of that, should be taken into account, especially in point-to-point applications such as data center-to-data center communications or metropolitan networks. Different from what we propose, the experiments were performed with simulated QKD equipment, and have not integrated the authenticated key exchange (AKE) proposal into a network protocol such as TLS. Moreover, QKD is used as a source of pre-shared keys that can authenticate the post-quantum key exchange.

Huang et al. [45] described a theoretical mechanism for integrating classical, PQ cryptography and QKD into a network protocol, thus achieving hybrid 'triple-security'. The study proposes this method to be implemented in solutions such as IPsec or TLS for achieving quantum-resistant communications. Nonetheless, no results or analysis are presented on handshake performance or communication size, nor any practical implementation of the aforementioned protocols.

Shim et al. [46] proposed a system in which PQC and QKD are combined for augmented security. However, this was not done in a hybrid fashion, since the proposed implementation suggests to employ QKD secure communications at the backbone of the network, while PQ cryptography is used to secure communications end-to-end.

Finally, the work presented by Giron et al. [47] provided an analysis of the challenges related with designing a post-quantum hybrid key exchange, and propose to establish a "transitional security", where the need for establishing a quantum-resistant key exchange is more

urgent than authentication. The hybrid approach including both PQC and QKD as an alternative for achieving quantum-resistant TLS was also considered in the paper, but no further explanations are given on what would be the procedure, or the impact of such a solution. Nonetheless, the need to study different combination approaches for the key agreement, like the concatenation or Exclusively-OR (XOR) approaches, was also mentioned.

Given the aforementioned issues of both PQC and QKD, developing quantum-resistant TLS protocols that bring together the strengths of both cryptographic methods is essential. Such protocols should aim to add minimal overhead to the overall key agreement time. However, and to the best of the author's knowledge, none of the previously presented work managed to present a practical scenario in which QKD and PQC are combined for achieving an enhanced security during the key exchange, nor evaluated or analyzed the impact of such implementation. Thus, there is a research gap that needs to be filled.

In our work, we have developed a practical network scenario in which commercially available QKD equipment is integrated together with the reference implementations of CRYSTALS-Dilithium [48] and CRYSTALS-Kyber [49] PQC algorithms. Moreover, both of them have been incorporated into a well-known, industry-ready TLS library for secure communications: Mbed-TLS [50]. As a result, and to the best of the author's knowledge, we believe we are the first to evaluate the advantages and drawbacks, in terms of added communication cost and security enhancement, of integrating the previously described solution into an experimental network scenario that involves interaction with real equipment.

3. Network security: From classical to quantum-resistant communications

Network security is an increasingly critical issue in the digital age. The advent of quantum computing challenges the effectiveness of network security protocols that are based on classical cryptography. Therefore, these protocols must be adapted to become quantum-resistant and to maintain effectiveness in protecting data [6]. As depicted in Fig. 3, TLS is one example of a network security protocol widely used over the internet and whose security must be enhanced. In this regard, this section presents the concept of TLS, explores the structure of the protocol, and explains what needs to be improved in order to achieve quantum-resistant TLS communications.

3.1. Transport Layer Security

TLS is a network security protocol designed to provide safe communication channels over the internet between two communicating applications. TLS can be added on top of any system, providing a layer of security that allows client/server applications to communicate in a way that prevents eavesdropping, tampering or message forgery, thus ensuring privacy and data integrity between the two applications. At its core, TLS is built on top of a reliable transport layer protocol such as the transmission control protocol (TCP) [51]. The secure channel established by TLS should provide [52]:

- **Authentication**: Communicating peers' identity needs to be verified before communication can be established. The server side of the channel must always be authenticated. Client side authentication is optional.
- **Confidentiality**: The data sent over the secure channel travels encrypted, and must only be visible to the rightful endpoints of the communication.
- **Integrity**: Data sent over the channel, even if encrypted, cannot be tampered or modified by an external attacker without both endpoints noticing, ensuring that the data remains accurate, complete, and trustworthy during transmission.

The TLS protocol consists of two distinct layers: TLS Handshake and TLS Record protocols:

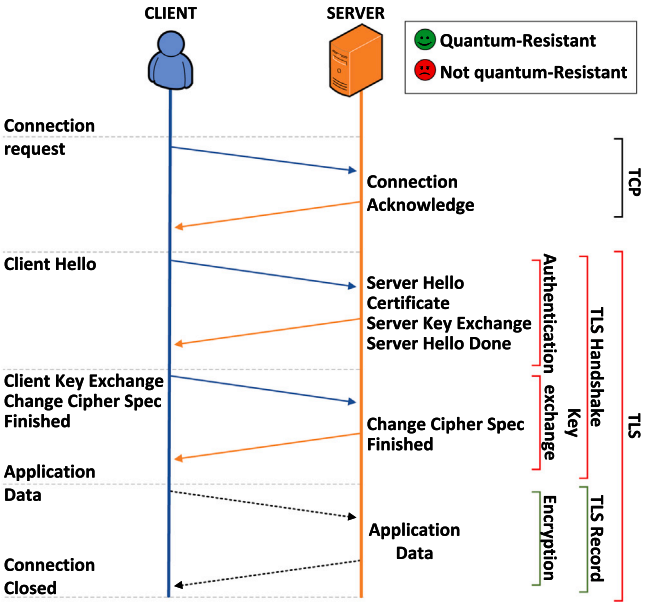

Fig. 3. Messages exchanged during authentication and session key negotiation in TLS 1.2 protocol [13].

3.1.1. TLS handshake

As depicted in Fig. 3, the TLS handshake allows communicating parties (client and server) to securely exchange a series of messages for identity verification and negotiation of the cryptographic modes, security parameters and shared key material that will be used for encrypting upcoming communication. The handshake phase must ensure that an attacker cannot interfere during the negotiation of the cryptographic parameters nor have access to the negotiated key material used for encryption. During the handshake, a series of messages are exchanged for completing both authentication and AKE phases. Messages highlighted in red use classical PKC algorithms not resistant to quantum attacks, and thus, need to be replaced by quantum-resistant alternatives.

Authentication: In current digital communications, authentication is the process in which two communicating parties can identify each other as the rightful endpoint of the communications. This process is usually done by means of PKC, as it is the only way to engage in communications with an unknown peer, and it is something ubiquitous in almost all existing network security protocols.

TLS, for example, provides authentication via X.509 public key infrastructure (PKI) certificates [53]. Such certificates are issued by trusted third parties, commonly known as certificate authorities (CA), and are used to verify the peer's identity and obtain the public key inside the certificate by leveraging a chain of certificates that goes all the way to a trusted root CA. TLS supports three different authentication modes: authentication of both peers, server-only authentication, and total anonymity.

In the most common TLS authentication scenario, server-only authentication, the server possesses a key-pair (private and public keys) to authenticate itself as a trusted endpoint. The server's public key is used by other peers (clients) to encrypt communications. This ensures that only the server, who owns the private key, can decrypt the messages that have been encrypted with the matching public key. It must be noted that, by nature, public keys are not designed to be confidential, and can be easily found either in public key servers or embedded within certificates.

Table 2 displays an estimate of the sizes of public keys and signed certificates to be exchanged during the authentication process for the most commonly used algorithms, as recommended by [54]. Longer key sizes can achieve higher security levels, but at the cost of directly

Table 2
PKC algorithms for digital signatures and signature verification: Our results. Size is presented in bytes. CRYSTALS-Dilithium1 and Dilithium3 are used for signature and signature verification and benchmarked against other classical algorithms. **Note that the sizes differ in the latest implementations of CRYSTALS-Dilithium [31].

Algorithm	Private key	Public key	Digital signature	Certificate (approx.)
RSA 1024	128	128	128	523
RSA 2048	256	256	256	784
RSA 4096	512	512	512	1296
SECP384r1	48	96	103	454
SECP521r1	65	132	139	529
PQC - Dilithium1	2096	896	1387	2570
PQC - Dilithium3	3504	1472	2701	4460

impacting message sizes and processing time, thus adding extra latency to the handshake (shown in Section 5). Therefore, it is crucial to find a balance between the achieved security level and key sizes. In a common authentication scenario, where certificates contain both public key and signature, the presented parameters can be used to calculate the approximate size of the certificate message exchanged during the authentication phase of TLS. The 'certificate' field shown in Table 2 assumes certificates containing only one signature and one public key.

The authentication phase in TLS involves the exchange of several messages between the client and server. As shown in Fig. 3, the messages exchanged during this phase include the client hello, server hello, certificate, server key exchange (for certain ciphersuites), and server hello done messages.

During the exchange of the client hello and server hello messages, the client and server agree on, among other things, the protocol version, selected ciphersuite, and random values to be used later in the protocol. The server's certificate is then sent to the client to allow the client to verify the server's identity using the certificate's signature. The server key exchange message is used in both authentication and key exchange. Within the authentication phase, the client can verify that the public key received belongs to the server whose identity has been previously verified. The server hello done message is sent by the server to indicate the completion of the authentication phase of the handshake. This message signals to the client that it can proceed with the next phase of the key exchange [13].

While the difficulty of deriving the relationship between private keys, public keys and digital signatures has been secure enough until now, the emergence of quantum computers makes these relations 'trivial' to solve, as Shor's algorithm has polynomial time complexity. Thus, potentially allowing attackers to use Shor's algorithm as a way to obtain a private key and impersonate the endpoint of the communications, which can lead users to suffer attacks such as man-in-the-middle (MITM), in which an attacker can insert itself into the communication channel among two endpoints [55]. Then he can establish TLS connections with each victim, relaying the messages among them without either party being aware of the attacker's presence. This setup allows the attacker to gain access to all the plain text exchanged among parties, and even selectively modify the transmitted data. Since both the client and server consider the channel to be safe, all incoming data is considered trustworthy. As a result, an attacker could severely alter the behavior of networks and users, with catastrophic consequences [55].

Authenticated Key Exchange (AKE): Process in which both participating parties generate the cryptographic material needed to establish a secure, shared secret, also known as the master secret. As explained in [13], the negotiation of a shared secret must generate a byte array that is completely unavailable to any eavesdropper. Several asymmetric cryptography algorithms such as RSA and Elliptic-curve Diffie–Hellman ephemeral (ECDHE) are currently used for establishing shared secret key material between authenticated parties. Similar to the behavior of digital signature algorithms represented in Table 2, Table 3 shows that the choice of stronger security levels for key exchange increases the size of the exchanged messages, potentially introducing extra processing and transmission overhead to the handshake when agreeing on a shared secret.

Table 3
PKC algorithms for key exchange/establishment: Our results for server key exchange and client key exchange. Size is presented in bytes. **Note that the size of ciphertext for Kyber512 has been augmented from 736 to 768 in the latest implementations of CRYSTALS-Kyber [32]. QKD key ID size is explained in Section 2.1.

Algorithm	Private key	Public key	Ciphertext/Key ID
ECDHE – x25519	32	32	–
ECDHE – SECP256r1	32	64	–
ECDHE – SECP384r1	48	96	–
ECDHE – SECP521r1	65	132	–
PQC - Kyber512	1632	800	736
PQC - Kyber768	2400	1184	1088
QKD	–	–	36

Tables 2 and 3 can be used to estimate the size of the server key exchange and client key exchange messages exchanged during the AKE phase. Message size corresponds both to the algorithm chosen to perform the AKE, as well as the digital signature algorithm employed by the server to sign the server key exchange message.

The AKE phase includes the server key exchange (for certain ciphersuites), client key exchange, client's change cipher spec, client's finished, server's change cipher spec and server's finished messages. The server key exchange message takes care of delivering the public key. It also includes a digital signature, so that the client can verify the trustworthy origin of the message. Then, the client key exchange message provides the server with the remaining information needed for the AKE. Both client and server's change cipher spec messages indicate that the delivering peer has derived the necessary keys for encryption and decryption, and will switch to the negotiated and newly established security parameters for all future communications. Finally, the finished message is always sent immediately after the change cipher spec to verify that both authentication and key exchange processes were successful. After client and server have received and validated the finished message from its peer, both parties may begin sending and receiving encrypted application data [13].

As it happened with authentication, the algorithms currently used by TLS for the key exchange/negotiation process are based on classical cryptography. If a quantum attacker gains access to the shared-secret, any effort to implement a secure symmetric encryption algorithm becomes useless.

3.1.2. TLS record

The TLS record protocol is responsible for the secure transmission of data between communicating peers. The protocol operates by collecting the data to be transmitted, fragmenting it into manageable blocks, compressing it if necessary, applying a message authentication code (MAC) to ensure the integrity of the data, encrypting with the already derived session keys and transmitting the resulting data to the other peer. The master secret negotiated during the handshake, as well as other security parameters, are used to generate the session keys that can be used within an AEAD symmetric encryption algorithm such as AES-256 Galois-counter mode (GCM) for the exchange of encrypted application data between client and server. The security parameters and secret values of a TLS connection are set via:

- **Connection end**: Indicates whether the communicating entity is considered the 'client' or the 'server'.
- **PRF**: A pseudorandom function (PRF) algorithm used to generate session keys from the master secret that is negotiated during the TLS handshake.
- **Symmetric encryption algorithm**: An algorithm used to perform symmetric encryption. This should include the key size needed by the algorithm, whether it is a block, stream or AEAD kind of algorithm, the block size of the cipher (if applicable) and the lengths of the initialization vectors (IVs) needed by the encryption algorithms.
- **MAC**: Algorithm used for message authentication. This includes the size of the value returned by the MAC.
- **Master secret**: A 48-byte shared secret negotiated among the two peers during the TLS handshake.
- **Client random**: A 32-byte random value provided by the client.
- **Server random**: A 32-byte random value provided by the server.

The record layer uses the aforementioned security parameters and secret values to generate the session keys via a PRF. Within the PRF, the master secret is expanded into a sequence of secure bytes, which is then split into the needed session keys. For the implementation of an AEAD algorithm that is quantum-resistant, such as AES-256 GCM [6], a total of 160 bytes of session key material are needed: 32 bytes of client write MAC key, 32 bytes of server write MAC key, 32 bytes of client write encryption key, 32 bytes of server write encryption key, 16 bytes of client write IV and 16 bytes for server write IV.

3.2. Combining PQC and QKD for quantum-resistant TLS communications

As it can be seen in Fig. 3, the TLS protocol combines both PKC and symmetric encryption in two sub-protocols (TLS handshake and TLS record) to securely exchange encrypted messages over insecure channels. During the handshake, the cryptographic keys needed to encrypt communications are negotiated. The operations involved during this phase for authentication and key exchange (e.g., digital signature, digital signature verification, ephemeral key-pair generation, key encapsulation, key exchange, key decapsulation, etc.) are currently PKC based and, therefore, face the threat of quantum computing.

In contrast, the TLS Record sub-protocol makes use of the previously negotiated cryptographic keys to implement symmetric encryption (e.g., AES-256). Although AES encryption algorithm might also be questioned by Grover's search algorithm [56], its efficiency can be mitigated via key expansion, thus AES encryption (with sufficiently long keys) can be considered quantum-resistant [6]. Given that TLS is practically ubiquitous in today's internet and infrastructure and its underlying symmetric encryption using AES is considered quantum-safe, it is highly desirable to adapt the TLS handshake so the processes of authentication and key exchange also become quantum-resistant.

Most of the studies done in the literature focus on PQC-based solutions for developing quantum-resistant communications in TLS. Specifically, Dilithium and Kyber are commonly used, as they have been selected as the proposed standards by NIST for implementing quantum-resistant authentication and key exchange [30]. However, in this work, we identify QKD as a key technology for enhancing the security of the key exchange process, as shown in Fig. 4. Although QKD does not provide a natural way of authenticating client and server like classical cryptography and PQC do (with digital signatures), thanks to the properties given by the quantum channel, having a pre-established QKD connection among client and server allows for the negotiation of shared, quantum-resistant secret keys, which then can be integrated into the protocol and combined with PQC cryptography for augmented quantum-resistant communications.

In such a system depicted in Fig. 4, authentication can be done via PQC, with the identity verification and cryptographic mode negotiation processes relying on the successful exchange of server hello, certificate, server key exchange and server hello messages. The aforementioned messages can be adapted to negotiate the use of PQC-based algorithms at the cost of increased message size. On the other hand, the key exchange can be enabled by means of PQC, QKD, or both technologies combined. Similar to the authentication process, PQC-based key exchange can be performed at the cost of increased client key exchange message size. In the case of combining QKD and PQC, an additional round trip is required, allowing both client and server to communicate with their respective QKD nodes via the key retrieval API [22] to obtain the same QKD-based key.

3.3. Security assumptions of quantum-resistant TLS

A detailed security analysis of both the CRYSTALS-Kyber and CRYSTALS-Dilithium algorithms has been presented in the literature [57,58]. Moreover, a security model for QKD is described in [59]. This security model presents an intricate examination of the assumptions, strengths, and potential vulnerabilities of QKD, suggesting an integration of QKD with classical cryptography to enhance the ultimate security of the entire system. In our case, we go a step further and propose the combination of QKD with PQC.

In the context of authentication and key exchange, PQC is specifically designed to offer robust security against the potential threats posed by quantum computers, thus ensuring security within the random oracle model [60]. Particularly, in the context of key exchange, QKD provides an inherently secure environment resistant to any exhaustive key search or cryptanalytic attacks, even those originating from quantum computers.

By combining QKD and PQC in the key exchange, we can ensure that the key remains safe as long as one of the algorithms is secure, thus addressing the problem of the key exchange. Quantum-resistant key exchange is a more urgent issue than quantum-resistant authentication; Impersonation cannot happen retroactively and, therefore, it is not a major threat until a sufficiently powerful quantum computer exists. However, in the context of key exchange, "harvest now, decrypt later attacks" attacks [11] can already take place, thereby resulting in a critical need for implementing quantum-resistant key exchange now [39].

We envision that the combination of both technologies allows, for example, to alleviate the existing fear towards adopting PQC-based quantum-resistant communications for AKE in the advent of possible algorithmic breakthroughs, to enable crypto-agile communication systems, and to considerably augment the security of the TLS protocol via multiple quantum-resistant cryptographic assumptions. Furthermore, our approach immediately confronts and mitigates the risks associated with potential "harvest now, decrypt later attacks" attacks, addressing an already significant security vulnerability in the current landscape of quantum computing.

4. Quantum-resistant TLS architecture

QKD and PQC are cryptographic techniques that address different challenges. While PQC targets both authentication and key exchange by means of different quantum-resistant algorithms, QKD is only suitable for enabling quantum resistant key exchange. Our work, shown in Fig. 4, proposes the following methodology: To use PQC for authentication (CRYSTALS-Dilithium), and to use both PQC (CRYSTALS-Kyber) and QKD for the AKE. To further strengthen the protocol against "harvest now, decrypt later attacks", directly related with the key exchange, we propose and analyze two novel ways in which QKD and PQC can be combined in the TLS AKE phase. The concatenation approach at the premaster secret level, and the XOR approach at the master secret level.

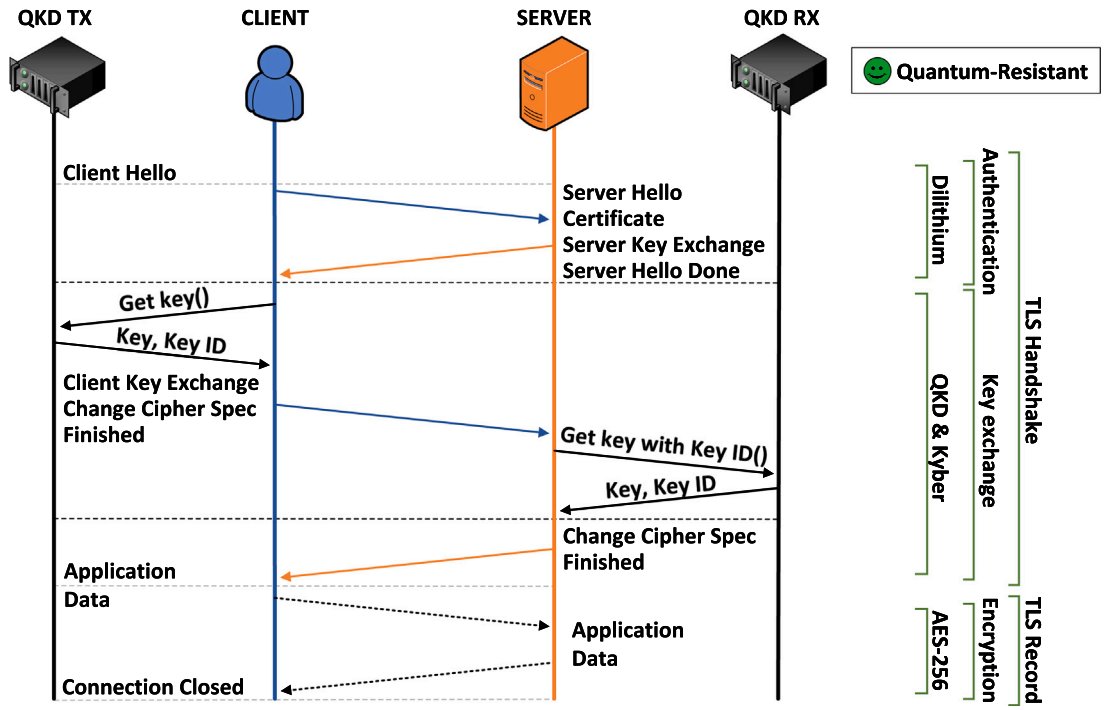

Fig. 4. Hybrid quantum-resistant TLS 1.2 protocol: Quantum handshake steps.

4.1. Quantum-resistant authentication

PQC is the first quantum-resistant authentication technology mature enough to be considered as an alternative for implementing authentication within the TLS handshake. In our implementation, and following the recommendations from NIST [30], CRYSTALS-Dilithium is used for digital signatures and signature verification. Table 2 shows a comparison of the Dilithium1 and Dilithium3 algorithms used in our experiments for digital signatures and digital signature verification, as well as the practical message sizes in comparison with some of the most commonly used classical algorithms. Within the TLS handshake, Dilithium is used by the server to sign the server's ephemeral public key. On the client side, Dilithium is used for both verifying the signature on the server's certificate, as well as to verify the signature of the server's ephemeral public key.

4.2. Hybrid quantum-resistant AKE

Regarding the AKE, we explain the two main mechanisms in which we believe that QKD and CRYSTALS-Kyber (or other PQC algorithm) could be combined together for augmented quantum security, and we explain how we could further integrate any other cryptographic algorithm for key exchange into our system. It must be noted that, as discussed in Section 2.1, both requesting SAE and corresponding QKD node must be located within the same point-of-presence.

Once both parties have successfully completed the QKD key retrieval process and generated a shared secret based on PQC, both client and server will have two secrets generated via different cryptographic assumptions. In our proposed hybrid quantum-resistant TLS architecture, the quantum-based secret keys can be used in several ways that are explained in the following subsections:

4.2.1. QKD-PQC AKE: premaster secret

In the first scenario that we propose, we explore the 'concatenation-based' approach. This line of work aligns with what is presented by [61] and depicted in Fig. 5. Here, each hybrid key exchange combination is considered a single, new key exchange method. In this context, we refer to the hybrid key exchange as the use of two (or more) key exchange algorithms that are based on different cryptographic assumptions, such as the combination of QKD and CRYSTALS-Kyber. The main motivation for using hybrid key exchange algorithms is allowing early adopters of either QKD or PQC security to include these quantum-resistant mechanisms into communication systems, guaranteeing that systems remain secure as long as at least one of the cryptographic algorithms remains unbroken.

The 'concatenation-based' approach employed in our protocol is summarized in Fig. 5. In our proposal, after the transmission of the necessary messages, both Kyber-based and QKD-based shared secrets are derived. Those two byte arrays are concatenated together into a single, cryptographically secure 'quantum premaster secret', which serves as the input key material (IKM) to the key schedule process. The 'quantum premaster secret' is fed into a PRF, as described in Section 3.1 to generate a 48 byte master secret, known as the 'quantum master secret'. The 'quantum master secret' is finally introduced into another PRF, which derives the necessary output key material (OKM) to be used in an AEAD symmetric encryption as session keys. This scenario also guarantees the security of the session keys used for the AEAD as long as one of the cryptographic algorithms used to generate the quantum-hybrid master secret remains unbroken.

To achieve an hybrid key exchange, it is recommended that PQC-based systems should combine next-generation (quantum-resistant) algorithms with pre-quantum cryptography, due to the fact that new PQC-based quantum-resistant algorithms are very recent and have not been subject to the same depth of study as other pre-quantum algorithms (e.g., RSA and ECC). Consequently, any potential algorithmic breakthrough that compromises the security of PQC algorithms may not entirely undermine the system's security but would rather only affect the security assumptions provided by PQC. Our current implementation rather includes QKD in combination with PQC for the authenticated key exchange, thus attaining hybrid quantum-resistance by means of two different quantum-resistant cryptographic assumptions.

In the proposed scenario, once the client has verified the identity of the server, the client triggers the communication with its corresponding QKD node by requesting a key. Once the client has obtained both a QKD-based key and a PQC-based key, the two byte arrays are concatenated to form the quantum-hybrid premaster secret. The

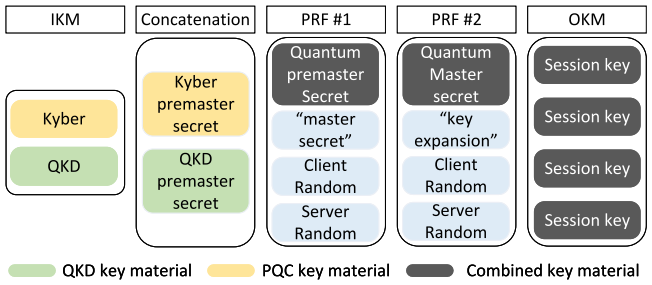

Fig. 5. Process of generating a quantum-hybrid premaster secret via concatenation of cryptographically secure byte arrays.

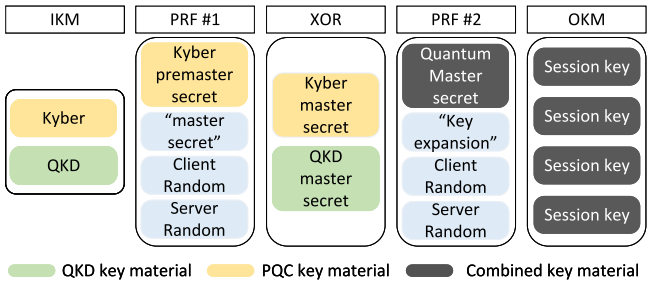

Fig. 6. Process of generating a quantum-hybrid master secret by XORing byte arrays of the same size.

resulting client key exchange message must contain both the ciphertext generated by CRYSTALS-Kyber and the key ID required by QKD. On the server side, this message can be used to obtain both the same QKD-based key and the PQC-based key, and thus to derive the same quantum-hybrid premaster secret. This can be used to derive the same quantum-resistant master secret at both sides, and distill the necessary session key material. In this way we can achieve a mechanism whose AKE is protected by two different quantum-resistant cryptographic assumptions. An estimated size of the messages exchanged during each version of our hybrid quantum-resistant TLS handshake can be calculated via Tables 2 and 3.

4.2.2. QKD-PQC AKE: master secret

In the scenario presented in Fig. 6, we explore the XOR approach, in which the QKD-based key is introduced inside the protocol at the master secret level, just after the first pseudo random function. This is a different version of the hybrid key exchange combination, in which the main motivation is to generate a key whose security is guaranteed by two different quantum-resistant cryptographic assumptions, but introducing them at different stages of the protocol.

As it happened with the aforementioned key schedule, the first step is the transmission of the necessary messages to derive the Kyber and QKD-based shared secrets, which will serve as IKM needed to derive the session keys. In this method, the Kyber-based shared secret is employed as the premaster secret. As shown in Fig. 6, the Kyber-based premaster secret is fed into a PRF, from which a 'post-quantum master secret' is derived. This secret is then XORed with the previously derived QKD-based master secret, generating what is known as the 'quantum master secret'. The quantum-hybrid master secret is then used in a PRF as described in Section 3.1.2 to derive the necessary OKM. As it happened in the previously described case, the TLS record protocol continues by performing symmetric encryption, and thus again guaranteeing that two quantum-resistant cryptographic assumptions need to be broken before the master secret becomes vulnerable.

5. Results and discussion on experimental quantum-resistant TLS architecture

The presented statistics and performance analysis are directly collected from experiments done in the Eindhoven QKD testbed [62,63]. As illustrated in Fig. 7, our experimental setup integrates two Raspberry Pi 4 Model B devices, each equipped with a 64-bit quad-core ARM Cortex-A72 processor running at 1.5 GHz, two IDQuantique Clavis3 QKD nodes and a switch in a restricted networking scenario. That means, all the equipment has been located in a single Virtual local area network (VLAN), creating an isolated logical network. This approach ensures that only authorized devices can access the VLAN, which helps to maintain the accuracy and reliability of the experiment results. The network time measurements have been collected via Wireshark [64], a widely known free and open-source packet tool for capturing and analyzing network traffic.

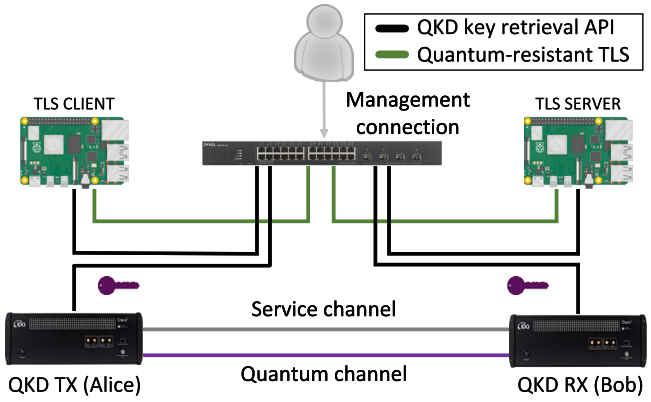

Fig. 7. Experimental quantum-resistant TLS network environment.

In this configuration, the communications on both quantum and service channels take place over individual optical fibers, in a direct point-to-point link between Alice and Bob. The classical communications among client, server, QKD Alice and QKD Bob happen through a gigabit Ethernet network switch. Within our experimental scenario, the reference implementations of CRYSTALS-Dilithium [48] and CRYSTALS-Kyber [49] PQC algorithms have been incorporated into a well known, industry-ready TLS library: Mbed-TLS [50]. The Raspberry Pi devices utilize the customized library to function as both a TLS client and a TLS server, enabling quantum-resistant communication within the testbed.

The remainder of this section is devoted to discussing the performance of classical cryptography and PQC within TLS 1.2, as well as the performance and added value of our hybrid QKD-PQC quantum resistant TLS. Additionally, this section introduces existing challenges and research gaps associated with our work.

5.1. Performance analysis of classical and PQC cryptographic alternatives for TLS

While the adoption of quantum-resistant cryptography alternatives for TLS is the main motivation for our work, performance is a critical factor in real-world network applications, thus it is imperative to evaluate the effectiveness of incorporating quantum-resistant cryptography alternatives into TLS from a performance perspective. For that, a performance evaluation of QKD and PQC in regards to classical cryptography has been conducted. Fig. 8 and Table 4 displays the mean network time measurements for each step of the TLS 1.2 handshake process of both classical and PQC cryptography, as depicted in Fig. 3. For the classical control groups, we selected several versions of both RSA and EC listed in Table 4, and compare those against quantum-resistant ciphersuites that use a combination of Dilithium1, Dilithium3, Kyber512, Kyber768 and QKD.

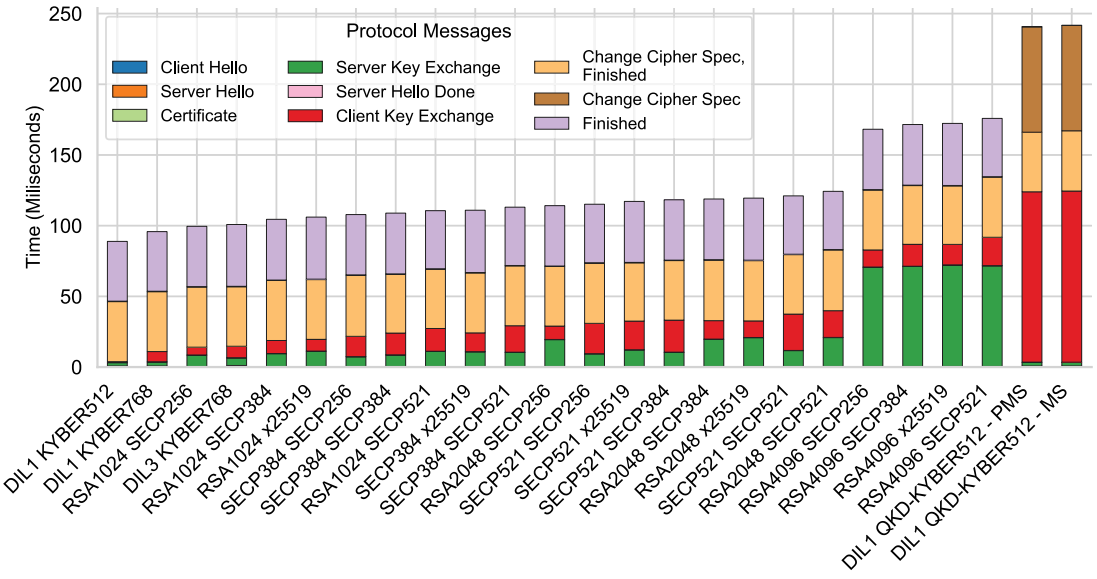

Fig. 8. Network time comparison of different classical and quantum-resistant cryptography algorithms inside TLS 1.2 (mean time per handshake step). Wireshark has been used for measuring the TLS 1.2 protocol step times over the described network scenario.

Within our PQC-based solution, and before the server key exchange is delivered, CRYSTALS-Kyber is used for generating an ephemeral key pair – private key and public key – which is then signed by the server's static CRYSTALS-Dilithium private key and sent to the client. On the client's side, signature verification first and key encapsulation thereafter are the most time-consuming steps before the client key exchange message is successfully delivered. As it can be seen in Fig. 8 and Table 4, the use of PQC-based cryptography not only provides the TLS protocols with quantum-resilience, but also improves the performance of the handshake. In fact, the comparison between PQC group Dilithium3 + Kyber768 and the classical control group SECP384 + x25519 determines that the PQC-based ciphersuite is 10 ms faster, which represents approximately a 9 % improvement. The PQC control group also outperforms other classical ciphersuites that are widely used to protect network communications, such as RSA2048 + x25519, RSA2048 + SECP256, SECP384 + SECP256 or SECP384 + x25519 by 20 %, 16 %, 11 % and 13 % respectively.

In general, as seen in Table 4, all the PQC-only based alternatives presented, with different levels of strength and key size, outperform most of the presented classical alternatives. In both PQC and classical cases, the main difference among ciphersuites becomes evident during the processes of both generating the server key exchange and the client key exchange messages. In our experiments, the verification of the server's certificate signature by the client does not affect the total time of the handshake, as it happens right after the certificate is exchanged, and in parallel to the server's ephemeral key-pair generation and signature processes.

However, the verification of the server's public key does affect the total time of the handshake, as it needs to happen before the client sends the encapsulated premaster secret. For example, in the previously described comparison, certificate exchange takes more time when using Dilithium. This is normal as the Dilithium certificates are considerably bigger than either RSA or EC-based certificates. Nonetheless, it can be observed that the time taken by the PQC-based ciphersuites to generate and send both client and server key exchange messages is considerably faster – 5 ms and 5 ms respectively – compared to the classical ciphersuites. This is attributed to the Dilithium and Kyber cryptographic operations being more efficient than their classical counterparts. It is in this context where the major performance differences among classical and post-quantum ciphersuites can be appreciated. The cryptographic operations referred above include, among others, public key-pair generation, key encapsulation, key decapsulation, digital signature, and digital signature verification.

After the master secret has been generated, the processes of creating the session keys and verifying the success of the handshake – change cipher spec and finished messages respectively – have approximately the same duration for each ciphersuite. This is due to the fact that all tested ciphersuites use the same processes outlined in the TLS record protocol for generating the session keys and encrypting messages, as the finished message needs to be sent encrypted.

Overall, only the classical ciphersuite which combines RSA-1024 for digital signatures and ECDHE-SECP256 for key establishment beats our proposed Dilithium3 and Kyber768-based post-quantum ciphersuite. In this case, the difference is minimal, with the classical group taking approximately 99 ms, while the post-quantum ciphersuite takes 101 ms. It must be noted that the aforementioned classical ciphersuite is not considered secure, and thus, it is not used in practice. Even in that scenario, the lighter version of our post-quantum solution (Dilithium1 + Kyber512) remains as the fastest ciphersuite for generating a client key exchange and server key exchange in the same restricted conditions, with a difference of 11 ms over its classical equivalent (RSA1024 + SECP256).

5.2. Performance analysis of hybrid QKD-PQC quantum-resistant TLS

As mentioned in Section 3, there is still some mistrust regarding the potential of quantum computing. To address this concern, it is relevant to analyze the overhead added by combining two quantum-resistant key exchange methods within the handshake. This approach not only provides augmented security, but also ensures fallback security in case one of the implemented cryptographic assumptions is compromised. To address this issue, we present two different versions of a hybrid quantum-resistant AKE, in which both QKD and Kyber are combined in the process of generating a master secret. QKD-PQC AKE: master secret and QKD-PQC AKE: premaster secret.

For the aforementioned versions of the protocol, Fig. 8 and Table 4 show a noticeable difference in delivering two specific messages during the handshake when QKD is used: Client key exchange and

Table 4
TLS 1.3 times per handshake step with different cryptographic algorithms – selected control groups and **quantum-resistant variants** – shown in Fig. 8. Results are in ms.

Algorithm	CH	SH	Cert.	SKE	SHD	CKE	CCCS Fin.	SCCS	Fin.	Total time
DIL1 KYBER512	0.0	0.31	0.48	2.30	0.21	0.52	42.47	0.31	42.27	88.88
DIL1 KYBER768	0.0	0.32	0.48	2.76	0.21	7.21	42.35	0.31	42.16	95.80
RSA1024 SECP256	0.0	0.29	0.15	7.97	0.14	5.51	42.56	0.30	42.62	99.55
DIL3 KYBER768	0.0	0.33	0.79	5.32	0.19	8.09	42.08	0.31	43.72	100.83
RSA1024 x25519	0.0	0.30	0.16	10.72	0.13	8.29	42.27	0.31	43.88	106.05
SECP384 SECP256	0.0	0.29	0.15	6.81	0.13	14.34	43.16	0.31	42.65	107.85
SECP384 x25519	0.0	0.29	0.15	10.33	0.14	13.26	42.40	0.30	44.05	110.91
RSA2048 SECP256	0.0	0.29	0.16	18.95	0.14	9.44	42.17	0.31	42.70	114.15
SECP521 SECP256	0.0	0.29	0.15	8.90	0.14	21.43	42.58	0.31	41.36	115.15
SECP521 x25519	0.0	0.29	0.15	11.70	0.11	20.25	41.21	0.30	43.13	117.14
SECP521 SECP384	0.0	0.29	0.15	9.98	0.13	22.63	42.14	0.30	42.68	118.29
RSA2048 SECP384	0.0	0.29	0.15	19.20	0.14	13.05	42.83	0.30	42.90	118.86
RSA2048 x25519	0.0	0.30	0.17	20.31	0.14	11.71	42.65	0.31	43.89	119.48
SECP521 SECP521	0.0	0.29	0.16	11.19	0.12	25.69	42.08	0.28	41.27	121.07
RSA2048 SECP521	0.0	0.29	0.16	20.39	0.14	18.89	42.97	0.28	41.16	124.27
RSA4096 x25519	0.0	0.30	0.17	71.63	0.18	14.48	41.37	0.30	43.90	172.34
RSA4096 SECP521	0.0	0.29	0.17	71.12	0.17	20.04	42.60	0.27	41.21	175.88
DIL1 QKD-KYBER512 Premaster secret	0.0	0.30	0.48	2.52	0.19	120.47	42.14	74.51	0.17	240.78
DIL1 QKD-KYBER512 Master secret	0.0	0.32	0.49	2.44	0.19	121.04	42.63	74.46	0.17	241.74

server's change cipher spec messages. This is due to the fact that the QKD key retrieval API implementation exposed by the QKD nodes to retrieve keys is used before these two steps are completed [22]. Given the present equipment, proposed networking environment and the obtained results, we can affirm that the QKD key retrieval API performs slow in comparison to the other aspects of the handshake, adding an average of ≈100 ms per key retrieval call.

From our analysis, we can conclude that the integration of QKD in combination with PQC generates an added overhead of approximately 117 % in regards to the classical control group used for comparisons (SECP384 + x25519). Since both Kyber512 and Kyber768 (PQC) perform fairly well, we concluded that the overall time of the handshake depends mainly on the overhead added by QKD. Also, the two methods in which we combine QKD and PQC for the key exchange, either by concatenating both premaster secrets or by XORing two master secrets, have negligible differences in terms of performance (less than 1 ms).

The strength of our quantum-resistant TLS protocol is that multiple quantum-resistant cryptography assumptions are used to calculate the master secret. That provides TLS with assurance that the protocol remains secure in the face of future quantum computing advances, since two quantum-resistant cryptographic assumptions must be broken before the protocol becomes vulnerable. Moreover, the scenario described for combining QKD and PQC is flexible enough to also introduce classical cryptography algorithms for the key exchange. In the same fashion that QKD and PQC keys for the key exchange were concatenated or XORed, a third byte array, generated by a pre-quantum key exchange algorithm could be employed to generate the master secret, whose security would then be guaranteed by three different cryptographic assumptions.

As a limitation, the performance of the QKD key retrieval API performs slower in relation to the rest of the handshake. However, we believe that this key retrieval process could be reduced to a latency of around ≈10 ms, as it is a purely local exchange within a secured node, opening the door for more performant quantum-resistant TLS implementations.

5.3. Challenges and research gaps

It must be noted that our proposed solution has been developed on top of TLS 1.2. The reasons for this choice are compatibility, availability and simplicity, since this version of the protocol is widely deployed within current networking infrastructure. Moreover, the architecture of TLS 1.2 is currently more suitable than TLS 1.3 for adopting QKD as a method for the key exchange.

5.3.1. Key exhaustion attacks on QKD

As explained in Section 2.1, it is crucial for any QKD-based implementation to maintain the use of QKD keys as efficiently as possible. Due to the distance and SKR generation limitations of current QKD equipment, the number of QKD-based available is limited. When including QKD into TLS (or any other network security protocol), it becomes critical that the client is fully aware of the other peer's identity before requesting a key from its corresponding QKD node. This is usually done via authentication between client and server. By having confirmation of the truthful identity of the communicating peer before triggering the QKD key retrieval process, both parties can ensure that any key requested from the QKD nodes will be efficiently used and not wasted, preventing denial of service (DoS) key exhaustion attacks [65]. This issue needs to be taken into consideration when migrating the proposed architectures to TLS 1.3.

High-speed TLS connections represent another challenge for using QKD as a method for key exchange. The latest standard of TLS establishes a limit on the volume of plaintext that can be securely encrypted using a specific set of session keys. For a quantum-resistant algorithm such as AES-GCM 256, it is possible to encrypt up to approximately 379 GB of data using keys derived from the same secret. After that, keys need to be updated [52].

To illustrate this in the context of high-speed TLS connections, assuming a data rate of 100 Gb/s, and key updates of 256 bits each, along with our QKD equipment average SKR of 2.5 kbps, we can use Eq. (1) to calculate the key renewal interval for high-speed AES-GCM. With this information, Eq. (2) can be applied to estimate the capacity for simultaneous user connections (UC) of our quantum-hybrid TLS protocol.

$$\text{Renewal Interval (s)} = \frac{\text{Key usage limit (GB)}}{\text{High-speed connection (Gb/s)}}. \quad (1)$$

$$\text{UC} = \frac{\text{QKD SKR (bits/s)} \times \text{Renewal Interval (s)}}{\text{Key Size (bits)}}. \quad (2)$$

According to our calculations, our protocol can sustain up to 292 simultaneous high-speed UC effectively while avoiding the issue of key exhaustion on QKD.

In any other case where the SKR is not sufficiently high to support TLS connections, QKD-enabled systems should employ a fallback mechanism for the AKE. This entails either using PQC or a combination of classical cryptography and PQC, until the SKR returns to an acceptable value.

5.3.2. TLS 1.2 vs TLS 1.3 for integrating QKD

The TLS 1.2 protocol currently allows the client to have confirmation of the authentic identity of the server before starting the QKD

protocol by verifying the signature on the server key exchange message, thus preventing any possible key exhaustion attacks from an unknown malicious party. On the other hand, the newer TLS 1.3 architecture does not currently allow either the client or server to verify the identity of the other communicating party (without major modifications) before triggering the QKD key retrieval process, thus the process becomes susceptible to key exhaustion attacks. In this regard, the integration of QKD and PQC inside TLS 1.2 allows for a smoother, faster transition, allowing every device to immediately switch to quantum-resistant communications without waiting for TLS 1.3 to be widely adopted. In the meantime, new approaches can be studied towards moving this quantum-resistant solution to the latest version of the protocol.

5.3.3. Suitability of QKD only key exchange: QKD AKE

As explained above, it is crucial to maintain the use of QKD keys as efficiently as possible due to the limitations of current equipment. To avoid possible QKD key exhaustion attacks, the client (or the entity requesting keys) should only retrieve QKD-based keys once it has verified the identity of the communicating peer via authentication. In a possible QKD-only AKE scenario, the client should be able to authenticate one of the messages sent by the server to establish the key material. By using the previously exchanged certificate and PQC-based (CRYSTALS-Dilithium) public key, the client could verify that the signature present on the certificate corresponds to the same entity that is signing the dynamically generated message, thus gaining confirmation that the server is trusted and making sure that the QKD-based keys will be efficiently used.

With classical cryptography (e.g., ECDHE) and PQC (e.g., CRYSTALS-Kyber), this can be done by signing the server key exchange message, which contains the key material needed to derive the premaster and master secrets. In the hybrid implementations where QKD and PQC are combined for the key exchange, this is done equally. However, if using QKD by itself as a mechanism for the key exchange in a server-only authentication scenario, and by doing a 'natural' mapping of QKD to both classical and PQC cryptography, the only element that could be signed and that would link the QKD key material to the ongoing TLS session would be the key ID. This imposes two main limitations:

- The server could be the one who starts the QKD key retrieval process, including the key ID inside the signed server key exchange message. However, the server would retrieve a QKD-based key each time a client 'claims' to have a QKD connection available with the server. Thus, a malicious client attacker could potentially do a DoS attack in the form of QKD key exhaustion.
- If the client is the peer who starts the QKD key retrieval process, there is no natural way without modifying the TLS handshake in which the client has the guarantee of the identity of the server (it cannot verify the other communicating entity other than after spending one QKD key, in which the result might be that the server is who he claims to be, or not) before triggering the QKD key retrieval API. Additionally, to link the QKD key material retrieved to the current session, the key ID should be signed dynamically before it is sent to the server. However, in a server-only authentication scenario (most commonly-used in current networks), the client does not have the capability to sign the key ID, nor the server to verify the signature provided by the client.

As a result, for the current TLS 1.2 version of the handshake and if no major modifications should be made to the protocol, QKD keys should always be used in a hybrid fashion in combination with another PQC based key exchange algorithm.

6. Conclusions and future works

In this work, we introduce a novel scheme for integrating PQC and QKD into TLS. Our proposed architecture allows the combination of QKD and PQC technologies to not only augment the security of TLS against quantum computing threats but also to maintain high performance in session key generation. We observe that, in restricted network conditions, PQC itself outperforms classical cryptographic algorithms by approximately 10 % of the overall handshake time. Regarding the authenticated key exchange process, we conclude that the use of QKD-based keys on top of PQC adds a considerable overhead of approximately 117 %. Our analysis indicates that such limitation is due to the performance of the key retrieval API that is exposed by our QKD equipment. However, for the presented scenario, the added overhead is well justified by the fact that the resultant master secret's security is now guaranteed by two different quantum-resistant cryptographic assumptions.

Both presented approaches for integrating QKD-based keys in the handshake perform similarly in terms of performance, with negligible differences (less than 1 %) among concatenating two byte arrays at the premaster secret level or XORing two byte arrays at the master secret level.

Furthermore, our work introduces a quantum-resistant TLS architecture that can combine QKD-based keys on both client and server sides with any other classical or PQC cryptographic algorithm of choice. This approach paves the way towards combining multiple cryptographic algorithms to achieve maximum security, not only against already well-known classical and quantum attacks, but also against other future not-known-yet attacks.

For the presented experimental solution to become widely used in real communication networks, there are still open research questions and challenges that require further investigation. First, the integration of the proposed solution into TLS 1.3 must be studied. Furthermore, due to the dynamic nature of QKD, it is expected that future implementations can communicate with network controllers in real-time, so that optical quantum links can be optimally deployed between client and servers according to changing network conditions. This requires the integration of quantum-resistant solutions like the one presented in this paper with SDN networks. Finally, as an engineering research question, our experiments concluded that the key retrieval API imposes the biggest performance limitations. Ultra-fast key retrieval APIs must be developed to reduce the performance cost of QKD as much as possible, fostering a new era of robust, fast and adaptable quantum-resistant communications.

CRediT authorship contribution statement

Carlos Rubio García: Conceptualization, Methodology, Software, Investigation, Writing – original draft, Validation. **Simon Rommel:** Conceptualization, Methodology, Writing – review & editing, Supervision. **Sofiane Takarabt:** Conceptualization, Software, Writing – review & editing. **Juan Jose Vegas Olmos:** Writing – review & editing, Supervision. **Sylvain Guilley:** Writing – conceptualization, Writing – review & editing. **Philippe Nguyen:** Writing – conceptualization, Writing – review & editing. **Idelfonso Tafur Monroy:** Conceptualization, Writing – review & editing, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

This work was supported by the EC H2020 MSCA ITN-ETN IoTalentum (grant no. 953442) and ECSEL JU project BRAINE (grant no. 876967) projects and the Dutch Ministry of Economic Affairs and Climate Policy (EZK), as part of the Quantum Delta NL programme. C. Rubio Garcia thanks Dr. Sebastian Verschoor from Eindhoven University of Technology for fruitful discussions regarding TLS and the work in this article.

References

[1] R.L. Rivest, A. Shamir, L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM 21 (2) (1978) 120–126, http://dx.doi.org/10.1145/359340.359342.

[2] V.S. Miller, Use of elliptic curves in cryptography, in: H.C. Williams (Ed.), Advances in Cryptology, Springer, Berlin, Heidelberg, 1986, pp. 417–426, http://dx.doi.org/10.1007/3-540-39799-X_31.

[3] N. Koblitz, Elliptic curve cryptosystems, Math. Comp. 48 (177) (1987) 203–209, http://dx.doi.org/10.1090/S0025-5718-1987-0866109-5.

[4] W. Diffie, M. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory. 22 (6) (1976) 644–654, http://dx.doi.org/10.1109/TIT.1976.1055638.

[5] M.J. Dworkin, E.B. Barker, J.R. Nechvatal, J. Foti, L.E. Bassham, E. Roback, J.F.D. Jr., Advanced encryption standard (AES), Federal Inf. Process. Stds. (197) (2001) http://dx.doi.org/10.6028/NIST.FIPS.197.

[6] M. Campagna, L. Chen, Ö. Dagdelen, J. Ding, J.K. Fernick, N. Gisin, D. Hayford, T. Jennewein, N. Lütkenhaus, M. Mosca, B. Neill, M. Pecen, R. Perlner, G. Ribordy, J.M. Schanck, D. Stebila, N. Walenta, W. Whyte, Z. Zhang, ETSI white paper no. 8 - quantum safe cryptography and security an introduction, benefits, enablers and challenges, Tech. Rep., (8) European Telecommunications Standards Institute, 2015.

[7] P.W. Shor, Algorithms for quantum computation: Discrete logarithms and factoring, in: Proceedings 35th Annual Symposium on Foundations of Computer Science, IEEE, 1994, pp. 124–134, http://dx.doi.org/10.1109/SFCS.1994.365700.

[8] P.W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM Rev. 41 (2) (1999) 303–332, http://dx.doi.org/10.1137/S0036144598347011.

[9] M. Mosca, Cybersecurity in an era with quantum computers: Will we be ready? IEEE Secur. Priv. 16 (5) (2018) 38–41, http://dx.doi.org/10.1109/MSP.2018.3761723.

[10] L. Chen, S. Jordan, Y.-K. Liu, D. Moody, R. Peralta, R. Perlner, D. Smith-Tone, Report on post-quantum cryptography, Tech. Rep. NIST IR 8105, National Institute of Standards and Technology, 2016, http://dx.doi.org/10.6028/NIST.IR.8105.

[11] N.A. Hassan, R. Hijazi, What's next? in: Digital Privacy and Security using Windows a Practical Guide, A Press, Berkeley, CA, 2017, pp. 273–278, http://dx.doi.org/10.1007/978-1-4842-2799-2_6.

[12] ITU, X.200: Information technology – open systems interconnection – basic reference model: The basic model, Tech. Rep., International Telecommunication Union, 1994.

[13] T. Dierks, E. Rescorla, RFC 5246 - the transport layer security (TLS) protocol version 1.2, Tech. Rep., Internet Engineering Task Force, 2008, http://dx.doi.org/10.17487/RFC5246.

[14] V.-L. Nguyen, P.-C. Lin, B.-C. Cheng, R.-H. Hwang, Y.-D. Lin, Security and privacy for 6G: A survey on prospective technologies and challenges, IEEE Commun. Surv. Tutor. 23 (4) (2021) 2384–2428, http://dx.doi.org/10.1109/COMST.2021.3108618.

[15] M. Sjoholmsierchio, B. Hale, D. Lukaszewski, G. Xie, Strengthening SDN security: Protocol dialecting and downgrade attacks, in: 2021 IEEE 7th International Conference on Network Softwarization (NetSoft), 2021, pp. 321–329, http://dx.doi.org/10.1109/NetSoft51509.2021.9492614.

[16] C. Rubio Garcia, O. Bouchmal, C. Stan, P. Giannakopoulos, B. Cimoli, J.J. Vegas Olmos, S. Rommel, I. Tafur Monroy, Secure and agile 6G networking – quantum and AI enabling technologies, in: 2023 23rd International Conference on Transparent Optical Networks (ICTON), 2023, pp. 1–4, http://dx.doi.org/10.1109/ICTON59386.2023.10207418.

[17] C. Rubio Garcia, S. Rommel, J.J. Vegas Olmos, I. Tafur Monroy, Enhancing the security of software defined networks via quantum key distribution and post-quantum cryptography, in: 20th International Conference on Distributed Computing and Artificial Intelligence (DCAI), in: LNNS, Vol. 741, Springer, 2023, pp. 428–437, http://dx.doi.org/10.1007/978-3-031-38318-2_42.

[18] V. Scarani, H. Bechmann-Pasquinucci, N.J. Cerf, M. Dušek, N. Lütkenhaus, M. Peev, The security of practical quantum key distribution, Rev. Modern Phys. 81 (3) (2009) 1301–1350, http://dx.doi.org/10.1103/RevModPhys.81.1301.

[19] A. Aguado, V. Lopez, D. Lopez, M. Peev, A. Poppe, A. Pastor, J. Folgueira, V. Martin, The engineering of software-defined quantum key distribution networks, IEEE Commun. Mag. 57 (7) (2019) 20–26, http://dx.doi.org/10.1109/MCOM.2019.1800763.

[20] W.K. Wootters, W.H. Zurek, A single quantum cannot be cloned, Nature 299 (5886) (1982) 802–803, http://dx.doi.org/10.1038/299802a0.

[21] P. Busch, T. Heinonen, P. Lahti, Heisenberg's uncertainty principle, Phys. Rep. 452 (6) (2007) 155–176, http://dx.doi.org/10.1016/j.physrep.2007.05.006.

[22] ETSI, Quantum key distribution (QKD); protocol and data format of REST-based key delivery API, Tech. rep., European Telecommunications Standards Institute, 2019.

[23] C.E. Shannon, Communication theory of secrecy systems, BSTJ 28 (4) (1949) 656–715, http://dx.doi.org/10.1002/j.1538-7305.1949.tb00928.x.

[24] G.S. Vernam, Cipher printing telegraph systems: For secret wire and radio telegraphic communications, Trans. AIEE 45 (2) (1926) 109–115, http://dx.doi.org/10.1109/JAIEE.1926.6534724.

[25] M. Lucamarini, A. Shields, R. Alléaume, C. Chunnilall, I.P. Degiovanni, M. Gramegna, A. Hasekioglu, B. Huttner, R. Kumar, A. Lord, N. Lütkenhaus, V. Makarov, V. Martin, A. Mink, M. Peev, M. Sasaki, A. Sinclair, T. Spiller, M. Ward, C. White, Z. Yuan, ETSI white paper no. 27 - implementation security of quantum cryptography introduction, challenges, solutions, Tech. Rep., (27) European Telecommunications Standards Institute, 2018.

[26] M. Takeoka, S. Guha, M.M. Wilde, Fundamental rate-loss tradeoff for optical quantum key distribution, Nat. Commun. 5 (1) (2014) 5235, http://dx.doi.org/10.1038/ncomms6235.

[27] Y. Cao, Y. Zhao, Q. Wang, J. Zhang, S.X. Ng, L. Hanzo, The evolution of quantum key distribution networks: On the road to the qinternet, IEEE Commun. Surv. Tutor. 24 (2) (2022) 839–894, http://dx.doi.org/10.1109/COMST.2022.3144219.

[28] D.J. Bernstein, T. Lange, Post-quantum cryptography, Nature 549 (7671) (2017) 188–194, http://dx.doi.org/10.1038/nature23461.

[29] D.J. Bernstein, Introduction to Post-Quantum Cryptography, Springer, 2009, http://dx.doi.org/10.1007/978-3-540-88702-7_1.

[30] G. Alagic, D. Apon, D. Cooper, Q. Dang, T. Dang, J. Kelsey, J. Lichtinger, Y.-K. Liu, C.M.D. Moody, R. Peralta, R. Perlner, A. Robinson, D. Smith-Tone, Status report on the third round of the NIST post-quantum cryptography standardization process, Tech. Rep. NIST IR 8413, National Institute of Standards and Technology, 2022, http://dx.doi.org/10.6028/NIST.IR.8413.

[31] S. Bai, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehlé, Crystals-Dilithium, 2023, https://pq-crystals.org/dilithium/index.shtml.

[32] R. Avanzi, J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J.M. Schanck, P. Schwabe, G. Seiler, D. Stehlé, Crystals-Kyber, 2023, https://pq-crystals.org/kyber/index.shtml.

[33] W. Barker, M. Souppaya, Getting ready for post-quantum cryptography: Exploring challenges associated with adopting and using post-quantum cryptographic algorithms, Tech. rep., National Institute of Standards and Technology, 2021, http://dx.doi.org/10.6028/NIST.CSWP.04282021.

[34] L. Chen, Cryptography standards in quantum time: New wine in an old wineskin? IEEE Secur. Priv. 15 (4) (2017) 51–57, http://dx.doi.org/10.1109/MSP.2017.3151339.

[35] W. Beullens, Breaking rainbow takes a weekend on a laptop, in: Advances in Cryptology, in: LNCS, Springer, Cham, 2022, pp. 464–479, http://dx.doi.org/10.1007/978-3-031-15979-4_16.

[36] M.-S. Chen, J. Ding, M. Kannwischer, J. Patarin, A. Petzoldt, D. Schmidt, B.-Y. Yang, Rainbow signature, 2023, https://www.pqcrainbow.org/.

[37] W. Castryck, T. Decru, An efficient key recovery attack on SIDH, in: Advances in Cryptology EUROCRYPT 2023, in: LNCS, 14008, Springer, 2023, pp. 423–447, http://dx.doi.org/10.1007/978-3-031-30589-4_15.

[38] C. Paquin, D. Stebila, G. Tamvada, Benchmarking post-quantum cryptography in TLS, in: J. Ding, J.-P. Tillich (Eds.), Post-Quantum Cryptography, Springer International Publishing, Cham, 2020, pp. 72–91, http://dx.doi.org/10.1007/978-3-030-44223-1_5.

[39] D. Sikeridis, P. Kampanakis, M. Devetsikiotis, Assessing the overhead of post-quantum cryptography in TLS 1.3 and SSH, in: Proceedings of the 16th International Conference on Emerging Networking EXperiments and Technologies, Association for Computing Machinery, New York, NY, USA, 2020, pp. 149–156, http://dx.doi.org/10.1145/3386367.3431305.

[40] G. Tasopoulos, J. Li, A.P. Fournaris, R.K. Zhao, A. Sakzad, R. Steinfeld, Performance evaluation of post-quantum TLS 1.3 on resource-constrained embedded systems, in: C. Su, D. Gritzalis, V. Piuri (Eds.), Information Security Practice and Experience, 2022, pp. 432–451, http://dx.doi.org/10.1007/978-3-031-21280-2_24.

[41] A. Mink, S. Frankel, R. Perlner, Quantum key distribution (QKD) and commodity security protocols: Introduction and integration, IJNSA 1 (2) (2009) http://dx.doi.org/10.48550/arXiv.1004.0605.

[42] J.S. Buruaga, H.H. Brunner, F. Fung, M. Peev, A. Pastor, D.R. López, L. Ortiz, V. Martin, J.P. Brito, VPN protection with QKD-derived keys using standard interfaces, in: 2023 23rd International Conference on Transparent Optical Networks (ICTON), 2023, pp. 1–4, http://dx.doi.org/10.1109/ICTON59386.2023.10207212.

[43] B. Hubermann, B. Lund, J. Wang, Quantum secured internet transport, Inf. Syst. Front. 22 (2020) 1561–1567, http://dx.doi.org/10.1007/s10796-020-10086-5.

[44] B. Dowling, T.B. Hansen, K.G. Paterson, Many A Mickle Makes A Muckle: A framework for provably quantum-secure hybrid key exchange, in: Post-Quantum Cryptography: 11th International Conference, PQCrypto 2020, Paris, France, April 15–17, 2020, Proceedings, Springer, 2020, pp. 483–502, http://dx.doi.org/10.1007/978-3-030-44223-1_26.

[45] L. Huang, K. Feng, C. Xie, A practical hybrid quantum-safe cryptographic scheme between data centers, in: Emerging Imaging and Sensing Technologies for Security and Defence V; and Advanced Manufacturing Technologies for Micro- and Nanosystems in Security and Defence III, Vol. 11540, SPIE, 2020, pp. 30–35, http://dx.doi.org/10.1117/12.2573558.

[46] K.-S. Shim, Y.-H. Kim, W. Lee, A design of secure communication architecture applying quantum cryptography, J. Inf. Sci. Theory Pract. 10 (spc) (2022) 123–134, http://dx.doi.org/10.1633/JISTaP.2022.10.S.12.

[47] A. Giron, Encouraging the adoption of post-quantum hybrid key exchange in network security, in: Security and Privacy in Communication Networks, Vol. 399, Springer, Cham, 2021, pp. 363–371, http://dx.doi.org/10.1007/978-3-030-90022-9_18.

[48] Crystals-dilithium reference implementation, 2023, https://github.com/pq-crystals/dilithium.

[49] Crystals-kyber reference implementation, 2023, https://github.com/pq-crystals/kyber.

[50] Mbed-TLS, 2023, https://github.com/Mbed-TLS/mbedtls.

[51] W. Eddy, RFC 9293 - transmission control protocol (TCP), Tech. rep., Internet Engineering Task Force, 2022, http://dx.doi.org/10.17487/RFC9293.

[52] E. Rescorla, The transport layer security (TLS) protocol version 1.3, Tech. rep., Internet Engineering Task Force, 2018, http://dx.doi.org/10.17487/RFC8446.

[53] R. Housley, Spyrus, W. Ford, VeriSign, W.P. NIST, D. Solo, Citicorp, Internet X.509 public key infrastructure certificate and CRL profile, Tech. rep., Internet Engineering Task Force, 1999, http://dx.doi.org/10.17487/RFC5280.

[54] G.M. Raimondo, L.E. Locascio, Digital signature standard (DSS), Tech. rep., (FIPS 186-5) National Institute of Standards and Technology, 2023, http://dx.doi.org/10.6028/NIST.FIPS.186-5.

[55] M. Conti, N. Dragoni, V. Lesyk, A survey of man in the middle attacks, IEEE Commun. Surv. Tutor. 18 (3) (2016) 2027–2051, http://dx.doi.org/10.1109/COMST.2016.2548426.

[56] L.K. Grover, A fast quantum mechanical algorithm for database search, in: Proceedings of the Twenty-Eighth Annual ACM Symposium on Theory of Computing, ACM, 1996, pp. 212–219, http://dx.doi.org/10.1145/237814.237866.

[57] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J.M. Schanck, P. Schwabe, G. Seiler, D. Stehle, Crystals - kyber: A CCA-secure module-lattice-based KEM, in: 2018 IEEE European Symposium on Security and Privacy (EuroS&P), 2018, pp. 353–367, http://dx.doi.org/10.1109/EuroSP.2018.00032.

[58] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, D. Stehlé, Crystals-dilithium: A lattice-based digital signature scheme, IACR Trans. Cryptogr. Hardw. Embed. Syst. 2018 (1) (2018) 238–268, http://dx.doi.org/10.13154/tches.v2018.i1.238-268.

[59] H.-K. Lo, M. Curty, K. Tamaki, Secure quantum key distribution, Nat. Photon. 8 (2014) 595–604, http://dx.doi.org/10.1038/nphoton.2014.149.

[60] N. Koblitz, A. Menezes, The random oracle model: A twenty-year retrospective, Des. Codes Cryptogr. 77 (2015) 587–610, http://dx.doi.org/10.1007/s10623-015-0094-2.

[61] D. Stebila, S. Fluhre, S. Gueron, Hybrid key exchange in TLS 1.3, Tech. rep., Internet Engineering Task Force, 2023, https://datatracker.ietf.org/doc/draft-ietf-tls-hybrid-design/.

[62] T.R. Raddo, S. Rommel, V. Land, C. Okonkwo, I.T. Monroy, Quantum data encryption as a service on demand: Eindhoven QKD network testbed, in: 2019 21st International Conference on Transparent Optical Networks (ICTON), 2019, pp. 1–5, http://dx.doi.org/10.1109/ICTON.2019.8840238.

[63] C. Stan, C.R. Garcia, B. Cimoli, J.J.V. Olmos, I.T. Monroy, S. Rommel, Securing communication with quantum key distribution: Implications and impact on network performance, in: Optica Advanced Photonics Congress 2022, Optica Publishing Group, 2022, p. SpW2J.2, http://dx.doi.org/10.1364/SPPCOM.2022.SpW2J.2.

[64] Wireshark, Wireshark: The World's Foremost and Widely-Used Network protocol analyzer, 2023, https://www.wireshark.org/.

[65] M. Mehic, S. Rass, E. Dervisevic, M. Voznak, Tackling denial of service attacks on key management in software-defined quantum key distribution networks, IEEE Access 10 (2022) 110512–110520, http://dx.doi.org/10.1109/ACCESS.2022.3214511.