Received: 15 December 2020 | Accepted: 5 February 2021

DOI: 10.1049/qtc2.12006

IET Quantum Communication

The Institution of Engineering and Technology WILEY

ORIGINAL RESEARCH PAPER

Using QKD in MACsec for secure Ethernet networks

Joo Yeon Cho[1] | Andrew Sergeev[2]

[1]ADVA Optical Networking SE, Fraunhoferstrasse, Martinsried, Germany

[2]ADVA Optical Networking Israel Ltd., Hatidhar Street, Ra'anana, Israel

Correspondence

Joo Yeon Cho, ADVA Optical Networking SE, Fraunhoferstrasse, Martinsried, Germany.
Email: jcho@adva.com

Funding information

H2020 Industrial Leadership, Grant/Award Number: 857156

Abstract

Media access control security (MACsec) is an IEEE 802.1AE standard for secure communication on Ethernet links. MACsec ensures the confidentiality, integrity and origin authenticity of Ethernet frames. The secrecy of MACsec stems from a root key that is either configured as a pre-shared key or derived from a mutual authentication protocol. However, both methods are not ideal because such a root key may be disclosed due to human errors or broken by quantum attacks. Here, the authors investigate the quantum key distribution (QKD) as an alternative source of trust for MACsec. QKD can be used as either a root key provider or a session key generator. The authors develop a new key exchange protocol based on QKD for Ethernet networks. Furthermore, it is verified by the experiment that QKD could be well integrated into MACsec without performance degradation.

1 | INTRODUCTION

Ethernet is widely used for connecting remote sites. Metro and carrier Ethernet are capable of transporting a large volume of data over a link layer for metropolitan and wide area networks. MACsec (media access control security) is a standard security protocol which can be used for establishing a secure channel over Ethernet networks [1]. It ensures data confidentiality, integrity and authenticity of Ethernet frames in transit. MACsec requires only 24 to 32 bytes of overhead for security. Hence, MACsec is an attractive solution for applications which require strong security as well as high speed and low latency.

The MACsec standard specifies ciphering suites that can be used for payload encryption and integrity check. Nevertheless, MACsec itself does not define any key exchange protocol nor a mutual authentication protocol. Those protocols are provided by the companion protocol called a MACsec key agreement (MKA) protocol which is defined in IEEE 802.1X [2]. MKA provides a mechanism of authenticating and authorising devices attached to networks and establishing a session key between peers. Each peer needs to hold a connectivity association key (CAK) that is a root key of the MKA key hierarchy.

There are two ways to establish CAK; one is to configure it as a pre-shared key (PSK) and the other is to execute an Extensible Authentication Protocol (EAP), from which a master session key (MSK) and a CAK are derived. PSK requires a trusted communication channel for the key distribution. Though CAK is not changed frequently, it may be disclosed by a human error or by a man-in-the-middle attack if the trusted channel is compromised or mishandled. On the other hand, an EAP method (e.g. EAP-transport layer security [TLS]) does not require an extra trusted channel; authentication between a server and a supplicant is based on the public-key cryptosystem. The authenticity of public-keys can be verified since they are signed by a certificate authority (CA) under the public-key infrastructure (PKI).

Recently, quantum attacks have drawn lots of attention for the network security. Most popular public-key cryptosystems, such as Rivest–Shamir–Adleman (RSA), elliptic-curve cryptography and Diffie-Hellman could be broken by Shor's algorithm in a polynomial time when large scale quantum computers are available [3]. Since the EAP methods in the MKA protocol rely on the security of the classical public-key cryptography, they could be vulnerable to quantum attacks. The MACsec protocol itself can be quantum-resistant by enforcing the usage of a 256-bit encryption key since MACsec defines only symmetric-key crypto algorithms for payload encryption and integrity check. On the contrary, the MKA protocol allows to use various classical public-key cryptosystems, leaving the MACsec protocol not immune to quantum attacks.

This is an open access article under the terms of the Creative Commons Attribution-NonCommercial License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited and is not used for commercial purposes.

© 2021 The Authors. *IET Quantum Communication* published by John Wiley & Sons Ltd on behalf of The Institution of Engineering and Technology.

1.1 | Our contribution

Here, we propose a quantum key distribution (QKD)-based MACsec key management framework for secure Ethernet networks. We assume that QKD has been already deployed and is available for MACsec key rollover. In a nutshell, two scenarios are taken into account. Firstly, QKD acts as a source of trust for the MACsec key hierarchy structure; QKD is used to generate a MSK, from which a root key of the connectivity association and other subsequent keys are derived in a hierarchical way. This scenario applies to the use-case where a standard MACsec key structure needs to be unchanged. Secondly, QKD is used to provide a SAK (secure association key) to peers without hierarchical key derivation. A SAK is a session key that is used to encrypt the payload of an Ethernet frame with a symmetric-key encryption algorithm. This scenario applies to the use-case where the standard MKA is not required and a key rate of QKD is sufficiently high so that a QKD key can be served as a session key to MACsec directly. In both scenarios, the key synchronisation is an important task to ensure that no packet loss occurs. We develop a QKD-based key exchange protocol to resolve this issue. We extend our framework to a trusted node network which consists of multiple QKD links. We implemented a QKD-based MACsec protocol on a commercial product and tested the secure packet transmission in a peer-to-peer Ethernet link.

The rest of this article is structured as follows: first, we briefly describe the background on MACsec and QKD. Then, we propose a framework of MACsec in a peer-to-peer QKD and in a trusted node network. Next, we describe our test platform and experimental results. Finally, we conclude the paper.

2 | BACKGROUND

Here, MACsec and MKA are briefly described in terms of encryption, authentication and a key management framework. Then, QKD deployment is reviewed. In addition, an European Telecommunications Standards Institute (ETSI) standard key interface for QKD is described.

2.1 | MACsec

As mentioned earlier, MACsec is a link layer security solution that is standardised in IEEE 802.1AE. A MACsec packet is constructed by adding SecTAG (security tag) and ICV (integrity check value) to an Ethernet frame. SecTAG contains information that identifies the protocol, the cipher suites, as well as a packet number for replay protection. ICV is calculated by GMAC (Galois message authentication code) to verify the integrity of the MAC destination address, the MAC source address, SecTAG, and the encrypted payload. A payload is encrypted by AES (advance encryption standard) cipher suite using either a 128-bit or a 256-bit key, depending on the configuration. Note that a payload encryption is optional. If a packet-authentication-only mode is configured, MACsec applies only ICV without the payload encryption. Figure 1 depicts the packet structure of a MACsec frame.

The cipher suites defined in MACsec are used for packet encryption, decryption and authentication. The standard defines the AES-GCM-128 with a 128-bit key as a default cipher suite. In the version of IEEE 802.1AEbn-2011, GCM-AES-256 is added as an optional cipher suite to allow a 256-bit key [4]. Later, GCM-AES-XPN-128 and GCM-AES-XPN-256 are added to IEEE 802.1AEbw-2013 for further optional cipher suites [5]. It makes use of a 64-bit PN (packet number) to allow more than 2^{32} MACsec frames to encrypt with a single SAK. MACsec is now part of the Linux kernel from the version 4.6 [6].

Although MACsec was developed for the local area network (LAN) where a hop-by-hop encryption is sufficient for the secure communication, MACsec can be extensively used for the WAN (wide area network) or the MAN (metropolitan area network) security by manipulating virtual LAN (VLAN) tags defined in IEEE 802.1Q [7]. An overview of the MACsec frame with VLAN tags is drawn in Figure 1. A MACsec frame can transverse across multiple hops supporting the VLAN tags. Hence, MACsec can be used for the security of large networks over carrier Ethernet.

2.2 | MACsec key agreement

MKA is a standard protocol defined in IEEE 802.1X-2010 that provides methods of the cryptographic key establishment for MACsec [2]. MKA is based on a hierarchical key derivation structure and a root key is configured as a pre-shared key or it is derived from a master key which is an outcome of an EAP method, for example, the EAP-TLS. A root of the key hierarchy is called a secure CAK. The possession of a CAK is a prerequisite for the MACsec membership and all potential members possess the same CAK. Each CAK is identified by a secure connectivity association key name (CKN). MKA does not use a CAK directly but derives two further keys from a CAK using the AES cipher in the CMAC mode: an ICV Key (ICK) and a key encrypting key (KEK) [8]. ICK is used to verify the integrity and authenticity of MKA packets, and KEK is used by a key server to transport a succession of SAKs, for use by MACsec, to the other members.

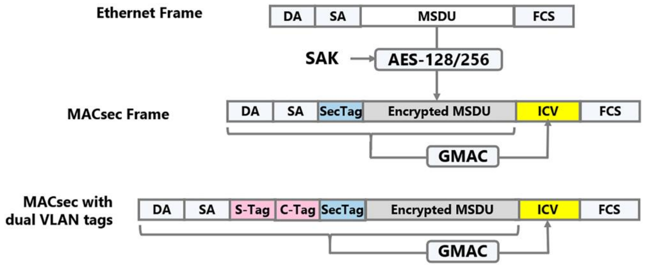

FIGURE 1 IEEE 802.1AE MACsec confidentiality and integrity check

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Oth Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

2.3 | Quantum key distribution

QKD is a method for secure key establishment based on fundamental laws of quantum physics. Since the first quantum key distribution protocol was developed by Bennett and Brassard in 1984 [9], there have been several different protocols for implementing quantum key distribution, all of which require both a quantum channel (to send quantum states of light), and an insecure classical channel which is bidirectional. The BB84 protocol remains one of the most widely implemented protocols in laboratories and commercial products.

The BB84 protocol consists of two parts: quantum transmission and measurement over a quantum channel and key distillation over a classical (non-quantum) channel. In QKD, Alice and Bob first exchange quantum bits over the quantum channel to generate a raw key. Then, they agree on a shared secret key from the raw key by performing a joint key distillation of the raw key by communicating over the classical channel. Note that a classical channel is not necessarily established over an optical network; it could be done over any communication channel available.

QKD became nowadays a matured technology which can be adapted to real-life applications. Field tests of QKD technology have been done over long distances of the order of 100 km [10, 11]. Commercial QKD systems are currently available in the market.

However, QKD has not been widely deployed as practical security technology, mainly due to the expensive hardware, limited distance and the requirement for a dedicated optical link. Multi-user QKD networks have been extensively investigated to provide cost-effective QKD systems. In [12], an upstream quantum access network is proposed to allow multiple end-users to share the most expensive component, that is, the single photon detector of a network node. In [10, 13], an effective secret-key allocation on users' demands is studied in a practical metro network and a quantum-secured passive optical network. The integration of QKD systems into commercial telecommunication networks has been demonstrated in [14, 15], thereby removing the constraint of a dedicated QKD link using a dark fibre.

Recently, the OPENQKD project has been launched to create a QKD-based secure communication network across Europe [16]. Funded by the European Commission, OPENQKD is expected to build multiple testbeds and try more than 25 use cases for a European quantum communication infrastructure.

2.4 | Key delivery interface

In order to widen the applications of QKD, it is necessary to establish a secure key delivery interface between QKD and the key consumers. Currently a key delivery application programming interface (API) using hypertext transfer protocol secure (HTTPS) has been standardised by the ETSI [17]. There are two stages for a secure key delivery interface: registration and key delivery. In the registration stage, both a QKD system and an encryptor create a public/private key pair, interact with the CA and install an HTTPS certificate on their system. The certificate contains a public-key that is signed by the CA. The certificate management such as enrolment, renewal and revocation is based on PKI. In the key delivery stage, a QKD system and an encryptor perform a key delivery protocol via an HTTPS channel so that a QKD key is delivered to an encryptor in a secured way.

Figure 2 illustrates a way to use the QKD key delivery APIs [17]. Each QKD key is identified with a universally unique ID, which is called a keyID.

The key delivery protocol is described as follows.

1. The encryptor A calls the key delivery API 'Get key' to get keys from the QKD A.
2. The QKD A delivers a key with its keyID to the encryptor A.
3. The encryptor A sends the keyID to the encryptor B.
4. The encryptor B calls the key delivery API 'Get key with key IDs' using the received keyID.
5. The QKD B delivers to the encryptor B the identical key with the identical keyID that are shared with the QKD A.

3 | INTEGRATING QKD INTO MACsec

Here, we propose two methods of integrating QKD with the MACsec. One is to take a long key stream from QKD and use it as a root of trust for the MACsec key hierarchy. The other is to take a symmetric-key periodically from QKD and use it as ephemeral session key for the payload encryption. We assume that a QKD link is already established over a separate channel such as a dark fibre.

3.1 | A root of trust

As described in Section 2, a CAK is the root key of a key hierarchy in the MKA protocol. According to IEEE 802.1X, a

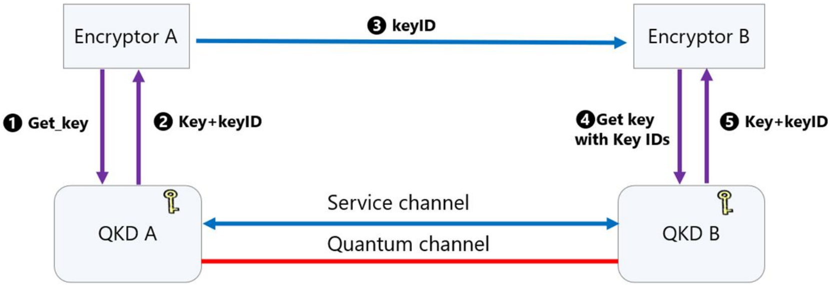

FIGURE 2 Use-case of the QKD key delivery interface defined in ETSI GS 014 [17]

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

CAK can be derived from either a pre-shared key (PSK) configured by administrator or a MSK automated by an EAP method. Hence, it is crucial to maintain the PSK or the MSK in a secure way since they are a security anchor of the key hierarchy.

Here, we propose the usage of QKD, instead of the EAP method, for the derivation of an MSK. According to the standard, an MSK should be a length of at least 64 octets (512 bits) [2]. This restriction is applied to our scenario; an MSK of at least 512 bits is taken from the QKD to ensure that a sufficient entropy is supplied to the key hierarchy. Note that the standard QKD interface allows users to get an arbitrary length of the key stream from QKD [17].

Both a CAK and a CKN are derived from MSK as follows.

$$\begin{aligned} CAK &= KDF(MSK, \text{“QCAK”}, mac1 \mid mac2, CAKlength) \\ CKN &= KDF(MSK, \text{“QCKN”}, KeyID \mid mac1 \mid mac2, \\ &\qquad CKNlength) \end{aligned}$$

where KDF stands for a key-derivation function. Both *mac*1 and *mac*2 are a part of two source addresses of the MAC. The KeyID represents a unique string identifying the MSK.

A structure of the key hierarchy in MKA using QKD is depicted in Figure 3. Note that a CAK may need to be refreshed when it is unexpectedly disclosed or it has reached the maximum life cycle. The limit of the key usage will be discussed later in Section 3.4. To update a CAK, another QKD key stream is taken as a new MSK and then, the procedure of the CAK derivation is repeated. Since each QKD key stream is generated independently, it is not possible to guess the new CAK from the knowledge of the old CAK.

KeyID A keyID is a unique random-looking octet string that identifies a cryptographic key obtained from QKD. According to a key delivery interface in Figure 2, a keyID is an universally unique identifier (UUID) format and is delivered from a transmitter to a receiver. Since a keyID is not secret, it can be delivered via any public channel. For the in-band communication, a keyID can be transmitted within an Ethernet frame with the use of the OUI Extended Ethernet Type.

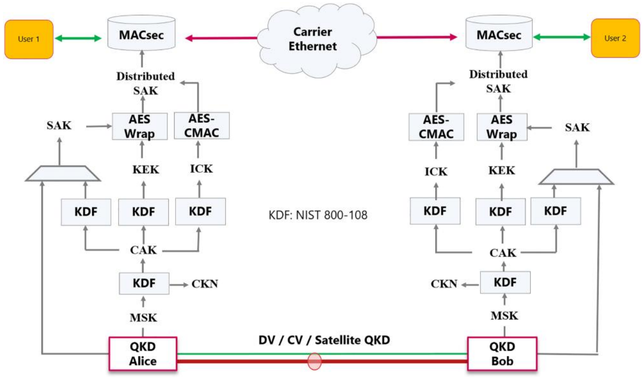

FIGURE 3 Hierarchical derivation of MACsec keys based on QKD. MSK: master session key, CAK: connectivity association key, KDF: key derivation function, ICK: integrity check value key, KEK: key encrypting key, SAK: secure association key

Key Derivation Function (KDF): In the standard MKA protocol, AES-CMAC-128 or AES-CMAC-256 is used as a KDF. Although MACsec supports both the 128-bit and the 256-bit symmetric-key crypto algorithms throughout the protocol, it is known that Grover's algorithm can achieve quadratic speedup of brute-force attack against symmetric key encryption [18]. Hence, it is recommended to use only the AES-CMAC-256 encryption algorithm for KDF to achieve the 128-bit quantum security.

3.2 | Ephemeral session key

The MACsec protocol is not necessarily implemented with the standard MKA protocol since those protocols are defined in the separate standard. In fact, a key hierarchy framework may not be suitable for the key management of a large scale of Ethernet networks. To simplify the key management of the MACsec protocol, an SAK can be generated independently by running an ephemeral key exchange protocol at every session. This is beneficial in practice since a security breach of the current session does not have any influence on the security of other session keys. On the contrary, in the standard model, if the CAK in the key hierarchy is revealed by accident, all SAKs which have been derived from the CAK are in danger.

To derive an ephemeral session key, QKD can be integrated into a hybrid key exchange protocol, as shown in Figure 4. At every session, the QKD generates a key stream of at least 256 bits. In parallel, a classical key exchange such as a Diffie-Hellman protocol is performed and a 256-bit classical key is generated. Both the QKD key and a classical key are combined in the key combiner and a new SAK is derived. In this way, the SAK can remain secure even when either a QKD link or a classical key exchange protocol is broken.

The procedure of the hybrid key exchange is stated as follows. The sequence of the protocol is summarised in Figure 5.

1. Both peers authenticate each other. In this figure, two peers are assumed to have a common PSK. This step can be replaced by a certificate-based authentication protocol.
2. Both peers perform a Diffie-Hellman protocol and derive *Key_dh*.

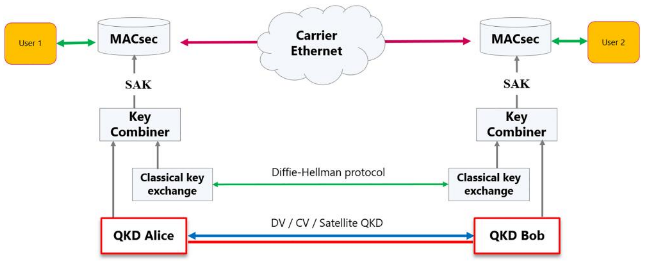

FIGURE 4 Ephemeral hybrid key exchange using QKD in MACsec

26322825, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

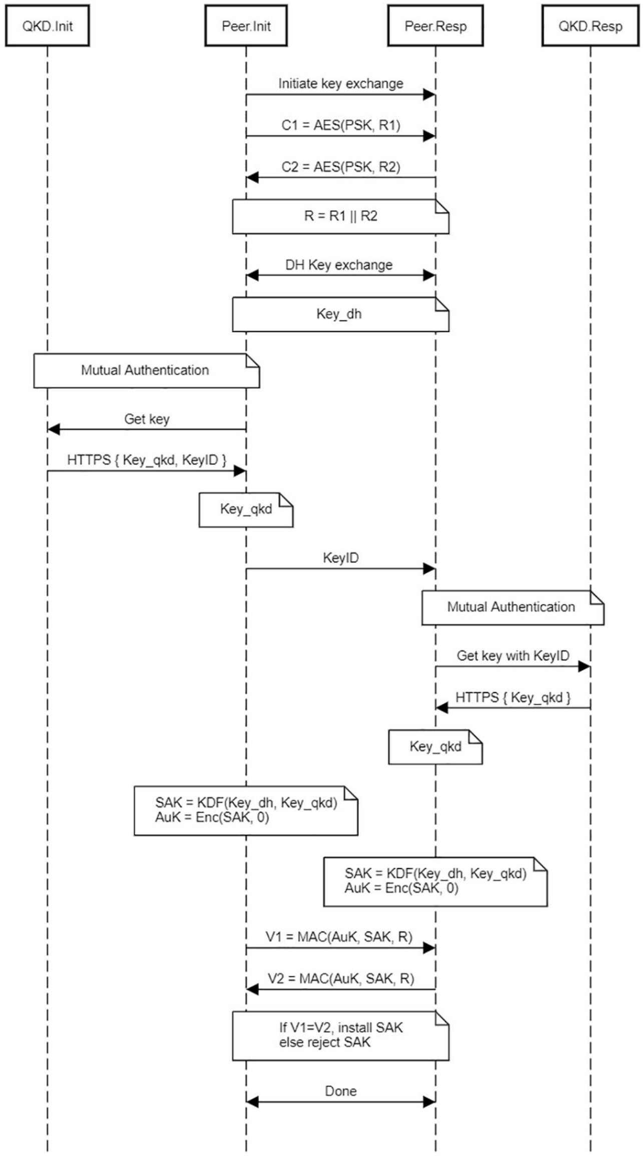

FIGURE 5 Authenticated hybrid key exchange using QKD and the Diffie-Hellman protocol

3. Both peers perform a standard key delivery interface and retrieves *Key_qkd*, as described in Section 2.4.
4. Both *Key_dh* and *Key_qkd* are combined by a KDF and an SAK is derived. In addition, a message authentication key (AUK) is derived from the SAK.
5. Both peers exchange the MAC value and verify that both peers share the same SAK successfully.

3.3 | Peer authentication

QKD itself does not provide a method for peer authentication. Hence, the QKD protocol without authentication is vulnerable to the man-in-the-middle attack. QKD in MACsec may be combined with classical authentication schemes based on RSA or ECDSA. However, they are not quantum-resistant. To defeat quantum attacks, the peer-to-peer authentication in MACsec should be done in a quantum-resistant (or post-quantum) setting. PKI is a common protocol for peer authentication. Several post-quantum PKI schemes have been proposed so far, for instance, in [19, 20]. Under these schemes, the X.509 certificate can be extensively built to support the post-quantum signature primitives. Although standardisation is still an on-going process, the third round finalists of National Institute of Standards and Technology (NIST) PQC project would be the promising candidates for post-quantum signature primitives [21]. There are three signature schemes in the final list, including FALCON and CRYSTALS-DILITHIUM, both of which are lattice-based schemes and Rainbow which is a multivariate signature scheme.

In addition, hash-based signatures (HSS) such as XMSS and LMS should be counted since they have been already standardised in the Internet Engineering Task Force (IETF) [22, 23] and supported by the NIST. HSS was initially proposed by Merkle in the late 1970s [24]. HSS does not rely on the conjectured hardness of mathematical problems. Instead, it is proven that it relies only on the properties of cryptographic hash functions. HSS schemes generally feature small private and public keys as well as fast signature generation and verification but large signatures and relatively slow key generation [22, 25, 26].

3.4 | Limits on key usage

A session key exchange occurs periodically, depending on a volume of traffic or a time interval of the channel. Lower bounds on the maximum amount of data that can be processed under a single key for AES-GCM is studied in [27]. According to TLS 1.3, up to $2^{24.5}$ full-size records (about 24 million) may be encrypted by AES-GCM on a given connection while keeping a safety margin of approximately 2¯57 for authenticated encryption (AE) security [28]. Since the maximum TLS record size is 2^{14} bytes, a single key in TLS 1.3 is recommended to be used for AES encryption up to $2^{38.5}$ data bytes, or $2^{34.5}$ blocks (1 block = 16 bytes).

Every MACsec frame contains a unique 32-bit or 64-bit packet number (PN). The (extended) PN can be used to configure a key lifetime parameter and becomes an initial vector of the GCM-AES-(XPN-)256 cipher suite under the defined MKA policy. Since the payload size of maximum Ethernet frame is 1500 bytes, a single key in MACsec can be used for AES encryption up to $2^{32} \times 1500 = 2^{42.6}$ data bytes, or $2^{38.6}$ blocks with 32-bit PN. According to [27], the security level is lower than 2^{50}. In a similar way, using a 64-bit PN, up to $2^{74.6}$ bytes of data can be encrypted with a single key while an attack success probability becomes quite high. Hence, it is recommended to switch an AES key much earlier before a 64-bit PN reaches the maximum number.

3.5 | Discussion

Here, we discuss two scenarios of using QKD in the MACsec protocol; a root of trust and an ephemeral key. An ephemeral

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Oth Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

QKD key scenario has several advantages. First of all, it provides the forward security. Since each ephemeral key is independently obtained from the QKD system, the knowledge of the current key does not provide any information on the previous key. Secondly, it can be easily integrated into a hybrid key exchange protocol, which is recommended for achieving the crypto agility.

On the other hand, a root key scenario is suitable for the standard key hierarchy used by MKA since it would be sufficient to replace an EAP method by QKD. Hence, the QKD can be easily integrated into an existing network using MKA. There is a risk though; if an MSK is revealed, the security of the key hierarchy is completely broken. Hence, the MSK should be regularly updated, depending on the key life cycle.

When QKD is integrated into the MACsec, a QKD key rate is another point to be considered. Namely, if a key rate is slow due to the distance of the quantum channel in QKD, an ephemeral key scenario may not be applicable. Suppose that the QKD key rate is a bit/sec on average and a 256-bit encryption key is requested every b seconds. Then, the QKD key rate should satisfy that $a > 256/b$. Note that the limit on key usage is determined, depending on the data rate and the security level, as discussed in Section 3.4.

3.6 | MACsec in the trusted node network

The distance limit of point-to-point QKD can be overcome by using either quantum repeaters or trusted nodes. Even though it is controversial, the trusted node network is still the most practical solution to build an arbitrary long distance of the QKD link. Now, we propose a method of the MACsec key agreement in a trusted node network under an ephemeral key exchange scenario.

In the context of QKD, a trusted node is a secure node which is allowed to collect keys from intermediate QKDs and deliver it to the next or terminal node in an OTP-encrypted form [29]. All trusted nodes are assumed to be trustworthy and tamper-resistant against adversary's attacks. The QKD protocol may differ for each QKD link and each QKD key may be produced independently by a different vendor. Hence, a key format and its metadata may be different from each other. Trust nodes should be able to interact with multi-vendor QKD systems, retrieve the QKD key in a standardised form, and relay the key to the terminal node.

A MACsec link using two trusted nodes is shown in Figure 6. Suppose that Alice and Bob establish a secure MACsec channel using a trusted node link. The role of trusted nodes TN-1 and TN-2 is to collect intermediate QKD keys and deliver to Bob via the standard key delivery interface which was described in Section 2.4. Note that TN-1 and TN-2 are vendor-agnostic; namely the QKD links in Figure 6 can be established using any QKD vendor. The MACsec frames are not terminated in the trusted node; they are re-transmitted to the next node with the encrypted payload untouched. We note that Figure 6 can be easily generalised to a meshed trusted node network.

4 | EXPERIMENTS

We implemented a MACsec protocol on the MACsec nodes connected back-to-back in the laboratory. A MACsec node was set up on the FSP150 ProVMe edge device which is composed of a field-programmable gate array (FPGA) and a Linux host [30]. The FPGA facilitates an embedded traffic generator and a packet analyser, while the Linux host runs a macsec-gw application performing the MACsec protocol.

We used an IDQ's Cerberis QKD system [31] as an external QKD key supplier. A QKD key is delivered to a MACsec-enabled device via the ETSI QKD interface [17]. An example of the QKD key format is shown as follows:

```
{[
  "key_ID":"3696ea11-baff-4924-99ac-
            b35c4014cd71",
  "key":"MRFe02s3wEx2OsV9qprOWGjFZV
            yP0vD6GCsN7zwDUVI="
]}
```

Note that *key_ID* is an ID of the key in an UUID format [32]. key is encoded in a base64 format [33].

We implemented an ephemeral key exchange, rather than the standard MKA protocol. The reason is that it is more suitable for the peer-to-peer Ethernet link in our testbed. In our experiment, two transmission channels have been established: one is for running the MACsec-enabled data communication and the other is for performing a hybrid key exchange using QKD and the Diffie-Hellman protocol.

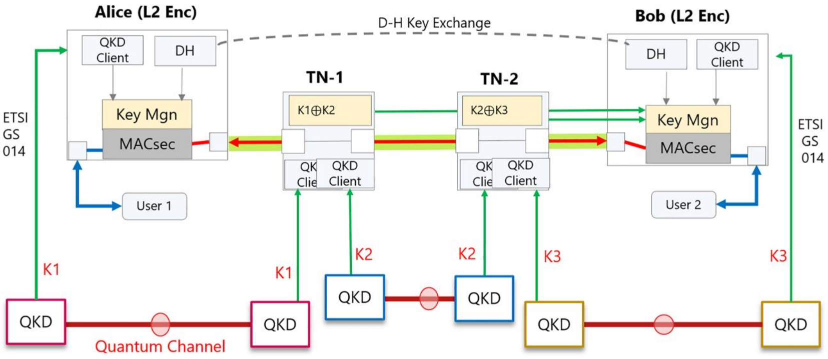

FIGURE 6 Establishment of MACsec link using two trusted nodes

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Oth Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

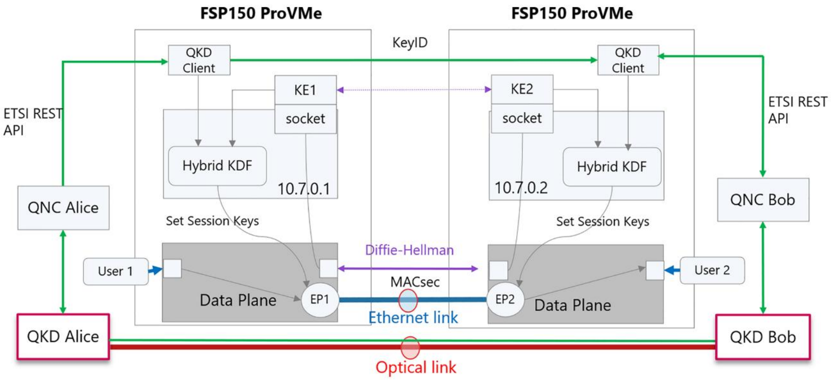

FIGURE 7 Test setup for MACsec using QKD. A hybrid key exchange is performed using QKD and the Diffie-Hellman protocol

A QKD key is accessed around every 10 min. Since the QKD key rate is around 2K bps and it is sufficient to provide a key stream for a SAK before the limit of the key usage is reached. A Diffie-Hellman key exchange engine is implemented in a separated application. It periodically derives a new session key and provides it to the hybrid KDF. In fact, a Diffie-Hellman protocol can be performed over either out-of-band or in-band channel through the regular kernel interfaces or a dedicated dataplane development kit (DPDK) Kernel NIC interface (KNI). This allows users to choose the best suitable communication path for their key exchange protocols. In our experiment, we chose an out-of-band key exchange using the DPDK KNI. A hybrid KDF block combines both a QKD key and a classical key. The output of the KDF becomes a SAK and is delivered to the MACsec application.

A MACsec packet starts with an Ethernet header with EtherType 0x88E5. Our test platform supports up to 1 G bps date rate. We tested an AES-GCM-256 MACSec in software. To get the best from x86 CPU, we used DPDK [34] with the aes-ni-gcm driver for symmetrical encryption. The throughput and average latency varied with IP packet sizes from 1300 Mbps and 49 µs for 64 Bytes to 7500 Mbps and 198 µs for 1420 bytes, respectively. For comparison, the ASIC-based MACsec adds approximately 1-3 µs of the latency and about 32 extra bytes of the overhead in a point-to-point direct link.

In this experiment, we worked with a pair of external QKD systems which support the ETSI QKD REST APIs. A typical rekey process is as following. The left endpoint (EP1) accesses a new Key from the corresponding QKD server and sends this key to the local Hybrid KDF. Then it sends a desired KeyID to the right endpoint (EP2) over an Ethernet link. The right endpoint accesses the key with the KeyID from its QKD server and sends the key to the local hybrid KDF.

After the new session key is agreed by EP1 and EP2, both endpoints must synchronise the usage of the new session key for encryption. This process is often called a key rollover; to make this process hit-less, we developed a proprietary handshake protocol. Using the handshake protocol we achieved and successfully demonstrated a hit-less rekey procedure. An overall structure of the test platform is shown in Figure 7.

5 | CONCLUSION

Currently, quantum attack is becoming a major concern on network security. Even though there is no concrete time line for a large scale of quantum computers, it is widely agreed that implementing countermeasures based on current available methods would be beneficial for a long-term network security.

Here, the authors investigate a QKD-integrated MACsec protocol for a quantum-secure Ethernet network under the assumption that a QKD network is already available. In order to comply with the IEEE 802.1X standard, QKD can be simply used as a source of trust for a standard key hierarchy. Instead of deriving a master session key from an EAP method, one can take a QKD key and use it as a root key. In this scenario, the security of the system heavily depends on that of the master session key. Alternatively, we propose a QKD-based ephemeral session key exchange protocol which can avoid such dependency. For many practical applications, it is beneficial to use an ephemeral key exchange since the unexpected disclosure of a certain session key does not compromise the security of any other session keys. Hence a hybrid key exchange protocol is highly recommended for this scenario to achieve the crypto agility. We showed by experiments that the proposed protocol can be performed with a reasonable speed and latency in Ethernet networks. In the future, we will extend our experiment to a large scale of the trusted node network using multi-vendor QKD systems under the OpenQKD project.

ACKNOWLEDGEMENTS

This research is co-funded by the OpenQKD project under the Horizon 2020 Framework Program of the European Union (Grant agreement No 857156).

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

ORCID

Joo Yeon Cho https://orcid.org/0000-0003-0351-0885

REFERENCES

1. IEEE Std 802.1AE-2018: IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Security (2018)

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Oth Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License

2. IEEE Std 802.1X-2010: IEEE Standard for Local and Metropolitan Area Networks--Port-Based Network Access Control (2010)
3. Shor, P.W.: Algorithms for Quantum Computation: Discrete Logarithms and Factoring. In Proceedings 35th Annual in Symposium on Foundations of Computer Science, pp. 124–134 (1994)
4. IEEE 802.1AEbn-2011: IEEE Standard for Local and metropolitan area networks–Media Access Control (MAC) Security–Amendment 1: Galois Counter Mode–Advanced Encryption Standard–256 (GCM-AES-256) Cipher Suite (2011)
5. IEEE 802.1AEbw-2013: IEEE Standard for Local and metropolitan area networks–Media Access Control (MAC) Security–Amendment 2: Extended Packet Numbering (2013)
6. KernelNewbies: Linux 4.6, 802.1AE MAC-level encryption (MACsec). https://kernelnewbies.org/. Accessed 7 December 2020
7. IEEE Std 802.1Q-2018: IEEE Standard for Local and Metropolitan Area Networks—Bridges and Bridged Networks (2018)
8. NIST SP 800-38B: Recommendation for Block Cipher Modes of Operation: the CMAC Mode for Authentication (2005)
9. Bennett, C.H., Brassard, G.: Quantum cryptography: Public key distribution and coin tossing. In: IEEE international conference on computers, systems, and signal processing, pp. 175–179 (1984)
10. Tang, X., et al.: Quantum-safe metro network with low-latency reconfigurable quantum key distribution. J. Lightw. Technol. 36(22), 5230–5236 (2018)
11. Dixon, A., et al.: 77 day field trial of high speed quantum key distribution with implementation security, QCrypt (2016)
12. Fröhlich, B., et al.: A quantum access network. Nature. 501, 69–72 (2013)
13. Wang, H., et al.: Dynamic secret-key provisioning in quantum-secured passive optical networks (PONs). Opt. Express. 29(2), 1578(2021). http://dx.doi.org/10.1364/oe.412188
14. Yoshino, K., et al.: Maintenance-free operation of WDM quantum key distribution system through a field fiber over 30 days. Opt. Express. 21(25), 31395–31401 (2013)
15. Ciurana, A., et al.: Quantum metropolitan optical network based on wavelength division multiplexing. Opt. Express. 22(2), 1576–1593 (2014)
16. OpenQKD: Open European Quantum Key Distribution Testbed. https://openqkd.eu/. Accessed 07 December 2020
17. ETSI GS QKD 014: Quantum Key Distribution (QKD); Protocol and data format of key delivery API to Applications (2018)
18. Grover, L.K.: A Fast Quantum Mechanical Algorithm for Database Search. In: Proceedings of the twenty-eighth annual ACM symposium on theory of computing, STOC 96, pp. 212–219
19. Bindel, N., et al.: Transitioning to a Quantum-Resistant Public Key Infrastructure. Cryptology ePrint Archive, Report 2017/460 (2017). https://eprint.iacr.org/2017/460
20. Kampanakis, P., et al.: The Viability of Post-quantum X.509 Certificates. Cryptology ePrint Archive, Report 2018/063 (2018). https://eprint.iacr.org/2018/063
21. Alagic, G., et al.: NIST IR 8309: Status Report on the Second Round of the NIST Post-Quantum Cryptography Standardization Process (2020)
22. Hüelsing, A.: XMSS: eXtended Merkle signature scheme. In RFC 8391. IRTF (2018). https://tools.ietf.org/html/rfc8391
23. McGrew, D., Curcio, M., Fluhrer, S.: Leighton-Micali Hash-Based Signatures. IRTF RFC 8554 (2019)
24. Merkle, R.: A Certified Digital Signature, Advances in Cryptology — CRYPTO' 89 Proceedings, CRYPTO 1989. Lecture Notes in Computer Science vol. 435, (1989)
25. McGrew, D., Curcio, M., Fluhrer, S.: Hash-Based Signatures. Crypto Forum Research Group, Internet-Draft (2018)
26. Bernstein, D., et al.: SPHINCS+: Submission to the NIST post-quantum project (2017)
27. Luykx, A., Paterson, K.: Limits on Authenticated Encryption Use in TLS. https://www.isg.rhul.ac.uk/kp/TLS-AEbounds.pdf
28. IETF: The Transport Layer Security (TLS) Protocol Version 1.3, Internet-Draft draft-ietf-tls-tls13-12.
29. ITU-T Y.QKDN_KM: Key management for quantum key distribution network (2019)
30. ADVA Optical Networking: FSP 150 ProVMe Series. https://www.adva.com/en/products/packet-edge-and-aggregation/edge-computing/fsp-150-provme-series
31. ID Quantique: Cerberis QKD system. https://www.idquantique.com/
32. Leach, P., Mealling, M., Salz, R.: A Universally Unique IDentifier (UUID) URN Namespace. Network Working Group, RFC 4122 (2005)
33. Josefsson, S.: The Base16, Base32, and Base64 Data Encodings. Network Working Group, RFC 4648 (2006)
34. DPDK: Data Plane Development Kit. https://www.dpdk.org. Accessed 7 December 2020

How to cite this article: Cho, J.Y., Sergeev, A.: Using QKD in MACsec for secure Ethernet networks. IET Quant. Comm. 2(3), 66–73 (2021). https://doi.org/10.1049/qtc2.12006

26328925, 2021, 3, Downloaded from https://ietresearch.onlinelibrary.wiley.com/doi/10.1049/qtc2.12006 by Oth Ostbayerische Technische Regensburg Hochschulbibliothek, Wiley Online Library on [27/03/2024]. See the Terms and Conditions (https://onlinelibrary.wiley.com/terms-and-conditions) on Wiley Online Library for rules of use; OA articles are governed by the applicable Creative Commons License