Juniper Networks | Driven by Experience™

VALIDATION OF A QUANTUM SAFE MACSEC IMPLEMENTATION

Is ETSI-QKD REST-API fit for purpose?

White Paper

TABLE OF CONTENTS

EXECUTIVE SUMMARY

The application interface between Quantum Key Distribution (QKD) and encryption devices plays a pivotal role in the acceptance of QKD for secure communication purposes. For this white paper, Juniper Networks, ID Quantique, and Deutsche Telecom successfully tested the REST-API defined by ETSI in a multivendor environment between a pair of QKD devices and a pair of firewalls featuring Media Access Control Security (MACsec) encryption. The goal was to test the ETSI-QKD REST API specification and identify practical application and implementation issues for secure communications using QKD and encryption devices. Results from these experiments revealed that further enhancements to standards are needed. A proper handling of time constraints for key requests and delivery on the classical and on the quantum side is mandatory. Going forward, the industry should extend existing security protocols and prepare guidelines and profiles for handling failure scenarios of QKD-key delivery.

Introduction

The common way to keep communications secret over the Internet uses VPNs based on MACsec[1] and IPsec[2] protocols to encrypt the communication. Architecturally, both methods form a combination of symmetric cryptography for payload encryption and asymmetric Public Key Cryptography (PKC) for key distribution. The symmetric cryptography for payload encryption is based on Advanced Encryption Standard (AES)[3], which can resist attacks by quantum computers[4] if the key size is at least 256 bits. The distribution of asymmetric keys prior to the encryption, however, causes concerns. Quantum computers could challenge public key distribution when they become powerful enough to run Shor's algorithm for large numbers.

One way to overcome this challenge is to replace PKC with Quantum Key Distribution (QKD)[5], a technology based on quantum mechanics to securely distribute cryptographic keys. The advantage of QKD is that quantum mechanics is provably secure. In a nutshell, QKD allows two distant devices to independently extract the same cryptographic key from a shared quantum channel with the information about the probability of the information being intercepted.

Since the information traversing the quantum channel is encoded in quantum states of photons, an observer cannot simply snoop on the photons to extract their states without altering them[6]. An altered state can be detected by the devices and excluded from being used as part of a cryptographic key[7]. To uncover the practical implications of using QKD to secure MACsec-based VPN traffic, Deutsche Telekom, ID Quantique (IDQ), and Juniper Networks performed a Proof of Concept (POC). By publishing these findings, the authors' intent is to contribute to and improve the interworking between ETSI's ETSI-QKD standard and IEEE's MACsec standard.

Network Setup

The POC used two Juniper Networks® SRX380 Firewalls (A) and (B) connected by a classical 10GbE link. An IEEE 802.1AE[1] compliant MACsec implementation based on Advanced Encryption Standard (AES256) encryption and IEEE 802.1X[8] as control plane protected the link from eavesdropping. Two IDQ Cerberis XG fourth-generation devices (C) and (D) were used as QKD-devices. The IDQ systems were connected via standard single-mode fiber-optic fiber with an IDQ eavesdropping simulator device (E) in between. The network setup was staged in the Deutsche Telekom laboratory in Nuremberg, Germany.

To communicate between QKD systems and firewalls, the ETSI GS QKD 014[9] specification has been used to provide a RESTful API by which an encryption device (firewall) can request quantum keys from the adjacent QKD device. In the first step, firewall (A) requests a key from the local QKD (C) and receives the key and a key identifier (key-ID) in response. In the second step, the remote firewall (B) requests a key for that key-ID from its local QKD device (D). The key-ID is not a hash code but a randomly assigned number that uniquely identifies the CryptoKey. This method requires a way to transport the key-ID from firewall (A) to firewall (B) for which protocols, parameter extensions, and encoding need to be defined.

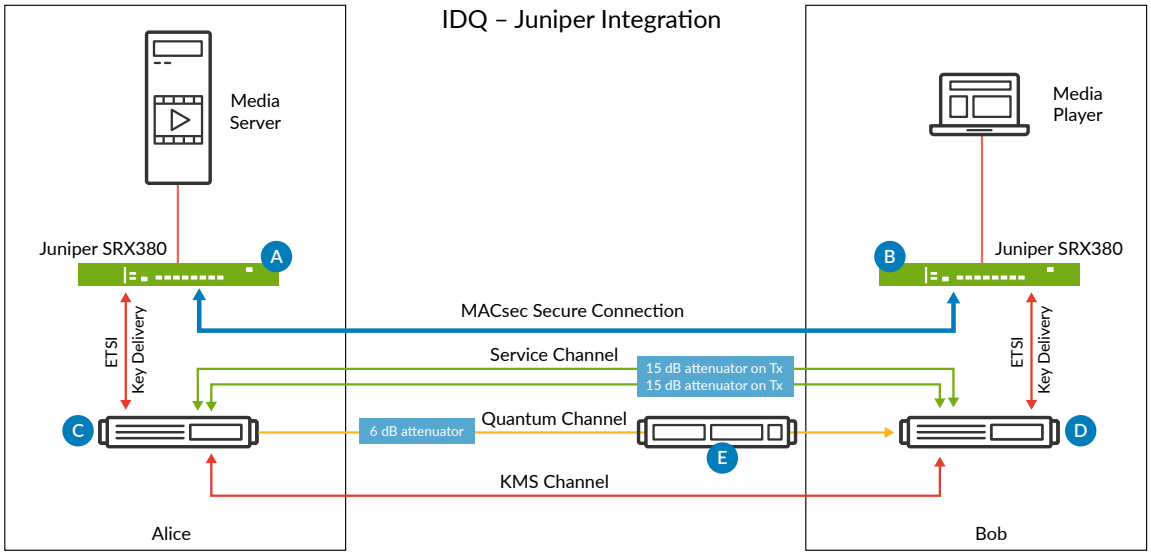

Figure 1: Proof of concept network setup

Description of each interface and device (top-down) is as follows:

1. Media server/player: The media server streams a video to the player using the encrypted link between the Juniper devices.
2. Juniper Networks SRX Series Firewalls (A) and (B): The SRX380 Firewall encrypts the link with MACsec so that all packets flowing across the link are encrypted.
3. MACsec secure tunnel: The 10 Gbps link between both Juniper devices is encrypted according to IEEE 802.1AE MACsec[1] leveraging IEEE 802.1X[8] as the control plane.
4. ETSI key delivery: The interface between the embedded IDQ Key Management Service (KMS) in devices (C, D) and SRX Series Firewalls (A, B) uses the Representational State Transfer (REST)-based key delivery API defined by ETSI[9].
5. IDQ QKD (C) and (D): Cerberis XG fourth-generation quantum systems distribute keys and embed KM/Key Management Entity (KME) services.
6. Service channel: Classical channel uses a proprietary protocol for timing and data reconciliation over standard single-mode fiber-optic (SMF).
7. Quantum Channel: Standard SMF fiber to transmit encoded photons at ~1550 nm.
8. Eavesdropping device (E): Device simulates an eavesdropper by granularly controlling a variable number of photons to be removed, delayed, and re-inserted into the path. This simulates an eavesdropper stealing photons from the fiber, manipulating and re-inserting them again unnoticed.
9. KMS channel: The key management channel over Ethernet shares topology information.

Workflows

ETSI GS QKD 014 Workflow

The ETSI-API allows two Secure Application Entities (SAEs), in this case the SRX Series Firewalls, to request the same key information from their locally attached KME, implemented within the IDQ QKD device. In a typical scenario, each firewall must first identify itself to the local KME. One firewall (A) then requests a key-cipher and key-ID from its local KME (C) and sends the key-ID to its remote peer (B). The remote firewall (B) then uses the key-ID to fetch the corresponding key-cipher from its local KME (D). After both sides are in possession of their respective key-ciphers, they can be programmed in hardware and activated.

MACsec Workflow

MACsec typically refers to a data plane and a control plane. The MACsec data plane is defined in IEEE 802.1AE[1] and deals with header definition and encrypted packet formats. The control plane is defined in IEEE 802.1X[8], which handles the encapsulation of the EAP. In a typical scenario, the MACsec-enabled interfaces of connected firewalls are configured by a pre-shared Connectivity Association Key (CAK), sometimes also denoted as the master key. The CAK is never used on the encrypted link. In a second step, one firewall derives a Session Association Key (SAK) from the CAK, encrypts it with a cryptographically modified CAK, and communicates it to the remote firewall. Now, both ends of the link know the CAK and the SAK. While the CAK is pseudo-static and stays local, the SAK used to encrypt traffic is frequently updated and shared across the link.

Findings

Certificates and Authentication over ETSI

Securing the ETSI GS QKD 014 REST-API with Transport Layer Security (TLS) requires a pair of client and server certificates, keys, and certificate authority (CA). The correct generation of these certificates is crucial for the proper functioning of consumer/provider integration. Incorrect certificate usage can cause TLS connection issues or protocol failures. In a secure environment, generating certificates and configuring them correctly cannot be applied in a plug-and-play manner. Instead, the process requires diligent configuration applied on each device.

Traffic Impact of QKD+MACsec

In this POC, there was no noticeable traffic impact caused by using QKD for MACsec. Key rollover was hitless, without any packet drops. The message delay caused by encrypting MACsec frames did not depend on the key distribution mechanism, meaning MACsec using QKD and standard MACsec with asymmetric PKC had the same negligible delay.

MACsec Control Plane Does Not Feature the Communication of a Key-ID

Two identifiers need to be communicated between firewalls to fetch the keys via the ETSI GS QKD 014[9]: key-ID and SAE (firewall) SAE-ID. Unfortunately, the MACsec control plane IEEE 802.1X[8] does not natively provide the means to request these identifiers. Since SAE-ID and key-ID are not considered security relevant, Juniper uses private options in Link Layer Discovery Protocol (LLDP) for the exchange. This allows Juniper to leverage a fully standard compliant MACsec implementation and rely on additional business logic for LLDP for the key-ID exchange. Going forward, the authors consider it preferable to perform the key-ID and SAE-ID exchange as an integral part of the standard IEEE 802.1X protocol.

Verifying ETSI GS QKD 014 Protocol Specification

The ETSI GS QKD 014[9] specification has no provisions to indicate the availability time of a key. In this POC, IDQ implemented a key availability timeout of 10 seconds after the key had been fetched from Alice's side (see Figure 1). In contrast, Juniper considered fetching keys as a non-time-critical event and implemented a request cycle where (A) and (B) performed an asynchronous request of their respective keys any time within a time range of 1 minute. The authors concluded that a future version of the ETSI GS QKD 014[9] should consider adding an option to clearly identify time constraints on both sides.

While working on a solution, the authors had the opportunity to study the network behavior during invalid key requests. In all cases, the MACsec communication between (A) and (B) was still functional and did not interrupt the service. More details about this finding are discussed in the following section.

Once the authors found a solution to work with the <10 seconds lifetime limit of keys, the correct working of the key exchange functionality was verified. Both firewalls presented themselves as SAEs to their KMEs. Every 60 seconds, the SRX Series Firewalls (A, B) sent key requests to the IDQ-KME (C, D) and subsequently configured a new CAK in MACsec. The choice of requesting 1 key/min was not security-related but chosen to accelerate the test so that it could be performed in a reasonable timeframe.

Use of QKD-Keys in MACsec

To cryptographically protect the traffic, MACsec utilizes two types of keys. The CAK is considered a long-term secret key stored locally and is not changed often. This secret key is used as a root-key to encrypt the distribution of SAK.

In contrast to a CAK, SAKs are relatively short-lived keys. SAKs are used by MACsec to encrypt the bulk of data frames on the LAN connection. When SAKs are transferred between devices, the messages are protected by a secret derived from the CAK. At present, no provision in the standards describes how keys provided by QKD (QKD-keys) shall be used by either end: as CAK, SAK, or both?

Considerations of using the QKD-key as CAK:

- The QKD-key in the form of an AES-256-bit string can be directly used as a CAK.
- Intermittent errors in the key exchange between QKD providers and consumers pose no immediate security threat to the MACsec encrypted data channel if the lifetime of the previous CAK is not exceeded.
- SAKs, used to protect traffic, are cryptographically derived from the CAK. They are frequently updated as per the MACsec standard to not exceed a certain amount of data encrypted using the same SAK. SAKs are considered secure if the CAK is secure, so frequent CAK updates also improve SAK security.

Considerations of using the QKD-key as SAK:

- The SAK is derived from the CAK and can be strengthened by merging the SAK with the QKD-key.
- This merger requires including the key-ID to the SAK key exchange.
- Intermittent errors in the key exchange (ETSI) could cause a starvation of keys available to the firewall.
- A fallback procedure needs to be defined to keep the communication secure and operational.

The authors concluded that an extension to IEEE 802.1X[8] is needed to negotiate between encryption devices (firewalls) whether QKD-keys will be used and which key type they represent: CAK or SAK. If there are multiple ways to apply the keys, such as by simply copying them or by using cryptographic derivation functions, those functions need to be defined and negotiated as well. Finally, a way to encode and exchange the SAE-ID and key-ID between the nodes is required.

Behavior Specification for MACsec with QKD-key

If the decision is made to use QKD keys for MACsec, it implies a specific security level. If the key consumer (firewall) cannot retrieve keys from the QKD Key Manager and the endurance of the available keys is exhausted, the service needs to be stalled until keys can be retrieved again. This is necessary to prioritize confidentiality over service availability.

To achieve this goal, the request frequency for QKD-keys needs to be higher or equal to the frequency for the permissible active key lifetime (AKL) $f_{QKD} >= f_{AKL}$. If the lifetime of the active key (T_{AKL}) is exhausted, an alarm should be raised, and the communication stalled. A suggested improvement for solution providers is to raise a notification whenever a configurable threshold for the key lifetime is exceeded. This way, the operator could take preventive actions to avoid traffic loss. This point is important because if QKD is used to achieve a specific security level, it is not possible to fall back to a lower security level.

From this requirement, the authors derived further requirements depending on the solution architecture. Two modes need to be considered by an operator when deploying a solution:

Mode 1: $f_{QKD} >> f_{AKL}$

> The key used in transmission is pseudo-static but more frequently rolled over using QKD-keys. In this mode, a single failed QKD request does not cause a security issue and the encryption device can continue requesting QKD-keys with the frequency f_{QKD}. Only if n consequent requests fail and ($f_{QKD} \leq \boldsymbol{n} * f_{AKL}$), the communication would need to be stalled as the active key exceeds its defined lifetime. In this mode, the

encryption device would hold a maximum of two QKD-keys for a short timeframe: the active key used to encrypt traffic and the new key fetched from the QKD during the process of a key rollover.

As an example: NIST[10] suggests a lifetime of 1 year for a symmetric master key (f_{AKL} = 1/12 month), which allows for a f_{QKD} = 1/month to fulfill the criteria of Mode 1. An implementation based on this example could continue to be active for 12 unsuccessful QKD-key requests (f_{QKD} = 1/month * 12 = f_{AKL}) before communication needs to be stalled. Thus, the probability of an encrypted link becoming stalled is comparable to today's use of non-QKD MACsec.

Mode 2: $f_{QKD} \sim f_{AKL}$

The active key used in transmission is short-lived and rolled over by fetching QKD-keys before the active key expires. In this mode, a failed QKD request will cause a security issue because the active key lifetime is exceeded, and the communication will need to go down. In this mode, it is advisable to keep a buffer of pre-fetched QKD-keys in the encryption device so that the link can remain active by fetching a key from the buffer. In case of consequent failed QKD-key requests, the buffer might drain and once empty, the encrypted communication will be stalled. If later the QKD communication becomes available again, the buffer can be replenished, and communication can restart. In this mode, the encryption device would hold a maximum of k+1 QKD keys for an extended timeframe with k = buffer size.

As an example: SAKs should be rolled over before 2^{32} data frames have been exchanged. On a 10 Gbps link at line speed using 128-byte frames, this condition is reached in about 600 seconds or $f_{AKL} \leq 1/10$ min. Note that in this mode, it is necessary for the SAE devices to fetch keys faster than strictly required by the lifetime of SAKs. This fills up a drained key buffer, should it occur. Hence, a frequency of f_{QKD} = 2/10 min would fulfill the criteria of Mode 2. For example, an implementation where SAEs maintain a buffer of k=10 keys could continue to be active for 100 minutes before the service needs to be stalled.

Note that a mode 2 implementation is demanding key generation rates that are orders of magnitude higher than for mode 1. Since QKD-keys are considered rare resources, the authors used a mode 1 implementation in the POC. As expected, only two QKD-keys were stored on the firewalls (the active key and the rollover key) during the POC. Because the lifetime of the CAK is much longer than the duration of the experiment, the lifetime expiration of CAK could not be tested. However, as a side effect of the mismatch in QKD-key time constraints (see 4), the authors observed several occasions of intermittent key request failures. None of these key request failures caused a service interruption as the CAK did not exceed its lifetime and the SAK was rolled over in time.

Conclusion

The test team successfully validated the REST-based key delivery API defined by ETSI[9] in the Deutsche Telekom lab. The API appears to be ready for a productized environment; however, further enhancements to applicable standards are needed. The authors suggest:

- Add a proper *handling of time constraints* in the ETSI REST-API
- Consider adding extensions coping with *exchanging additional identifiers* introduced by the ETSI REST-API such as key-ID and SAE-ID to existing protocol standards such as IEEE 802.1X

In parallel, the use of QKD-keys in a MACsec setup should be explicitly defined to achieve compatibility of implementations. Going forward, the industry should prepare guidelines and profiles on *handling failure scenarios* specific to QKD-key exchanges.

Bibliography

1 "IEEE Standard for Local and metropolitan area networks-Media Access Control (MAC) Security," in IEEE Std 802.1AE-2018 (Revision of IEEE Std 802.1AE-2006), vol., no., pp.1-239, 26 Dec. 2018, doi: 10.1109/IEEESTD.2018.8585421.

2 [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)," RFC 4303, DOI 10.17487/RFC4303, December 2005, <https://www.rfc-editor.org/info/rfc4303>.

3 Federal Information Processing Standard 197, Advanced Encryption Standard (AES), November 2001.

4 ETSI, "Quantum Safe Cryptography and Security; An introduction, benefits, enablers and challenges" F-06921 Sophia Antipolis Cedex - FRANCE, 2014; https://docbox.etsi.org/workshop/2014/201410_crypto/quantum_safe_whitepaper_1_0_0.pdf

5 Bennett, C. H. & Brassard, G. (1984). Quantum cryptography: Public key distribution and coin tossing. Proceedings of IEEE International Conference on Computers, Systems, and Signal Processing (p./pp. 175), India.

6 Stucki, Damien & Fasel, Sylvain & Gisin, Nicolas & Thoma, Yann & Zbinden, Hugo. (2007). Coherent one-way quantum key distribution. Proc SPIE. 10.1117/12.722952. https://arxiv.org/pdf/quant-ph/0506097.pdf

7 https://blogs.juniper.net/en-us/security/quantum-key-distribution-qkd-how-does-it-actually-work

8 "IEEE Standard for Local and Metropolitan Area Networks--Port-Based Network Access Control," in IEEE Std 802.1X-2020 (Revision of IEEE Std 802.1X-2010 Incorporating IEEE Std 802.1Xbx-2014 and IEEE Std 802.1Xck-2018) , vol., no., pp.1-289, 28 Feb. 2020, doi: 10.1109/IEEESTD.2020.9018454.

9 ETSI, "Quantum Key Distribution (QKD); Protocol and data format of REST-based key delivery API," F-06921 Sophia Antipolis Cedex - FRANCE, 2019. https://www.etsi.org/deliver/etsi_gs/QKD/001_099/014/01.01.01_60/gs_qkd014v010101p.pdf

10 Recommendation for Key Management—Part 1: General (Revised); Elaine Barker, William Barker, William Burr, William Polk, and Miles Smid; NIST Special Publication 800-57; https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-57p1r2007.pdf

About Juniper Networks

At Juniper Networks, we are dedicated to dramatically simplifying network operations and driving superior experiences for end users. Our solutions deliver industry-leading insight, automation, security and AI to drive real business results. We believe that powering connections will bring us closer together while empowering us all to solve the world's greatest challenges of well-being, sustainability and equality.

JUNIPER NETWORKS | Driven by Experience™

APAC and EMEA Headquarters
Juniper Networks International B.V.
Boeing Avenue 240
1119 PZ Schiphol-Rijk
Amsterdam, The Netherlands
Phone: +31.207.125.700
Fax: +31.207.125.701

Corporate and Sales Headquarters
Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, CA 94089 USA
Phone: 888.JUNIPER (888.586.4737)
or +1.408.745.2000 | Fax: +1.408.745.2100
www.juniper.net

2000792-001-EN July 2022