Thales Integration Guide

INTEGRATION GUIDE

Document Information	
Revision	1.1
Release Date	15th January 2025
Authors	Jarosław Ulczok, Thales Miroslaw Sopek, Quantum B Piotr Łuniewski, Ryszard Olejnik, Quantum B

CONTENTS

Integration of CN4020 with pQKD

Introduction

This document is intended to guide security administrators through the steps for the pQKD Integration with Thales HSE and outline the steps for enabling secure connection between SAE (HSE) and KME (pQKD).

Document Conventions

This section provides information on the conventions used in this document.

NOTE: Take note. Notes contain important or helpful information that you want to make stand out to the user.

CAUTION: The information included in the document is the results of practical tests on a demo platform used to identify all the steps required for a smooth and successful migration.

WARNING: Be extremely careful and obey all safety and security measures. In this situation you might do something that could result in catastrophic data loss or personal injury.

Introduction

This chapter describes core steps to integrate Thales CN4020 Encryptor with Quantum B pQKD device.

Overview

pQKD simulates the behavior of a genuine QKD system without requiring actual quantum hardware. It allows organizations to test QKD integration, interoperability, and network performance. By emulating QKD, companies can avoid the high costs associated with deploying actual quantum hardware and the necessary fiber infrastructure.

Figure 1 pQKD devices

Thales Encryptors (virtual CV or hardware CN) are part of Thales's line of certified encryption solutions trusted by governments and enterprises worldwide. They are designed to be quantum-safe, meaning they can be upgraded or integrated with quantum-resistant technologies like QKD or utilize QRA (quantum resistant algorithms). The CN4020 Network Encryptor (CN4010) provides an optical interface and high-assurance FIPS and Common Criteria certified encryption over Ethernet at full line rate speeds. The CN4020 is a versatile and simple-to-use platform that is user-configurable to provide highly secure, full line rate of network encryption to the x (FTTx) configurations.

Figure 2 CN4020 Encryptor

Integrating a QKD emulator with Thales encryptors enables organizations to prepare their infrastructure for future genuine QKD deployment. It provides a platform to validate the compatibility and effectiveness of QKD protocols with existing encryption hardware. Also, early integration and testing reduce the risks of adopting new, complex technologies.

Emulators eliminate the need for expensive quantum hardware and specialized components. Genuine QKD often requires dedicated fiber installations, which can be costly and logistically challenging. By understanding the requirements through emulation, companies can plan and allocate budgets more effectively.

Tested Platforms

The following platforms were tested:

Platform Tested	FM/SW Version
CN 4020	5.2.1
pQKD	2.6

Network connections

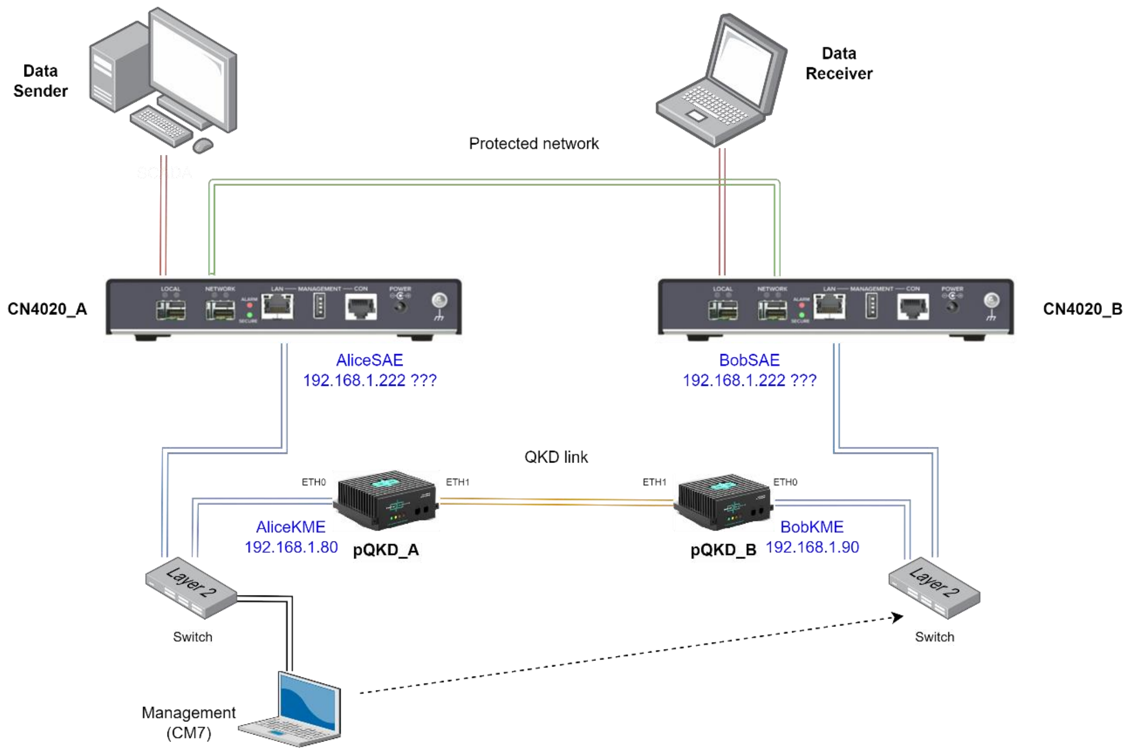


Figure 3 Network connection of CN4020s and pQKD devices

Prerequisites

Thales HSE setup

Refer to the Thales HSE documentation for installation steps and details regarding the configuration and initial setup of the HSE. Before you get started, ensure the following:

- Install the CM7 management application on the management computer.
- Assign an IP address to the HSE LAN/Management port via the serial interface.
- Discover and Activate the HSE in CM7.

Connection of CN4020 with pQKD (see Figure 3 for details)

- Connect the CN4020 device via the LAN port to the same local network as the pQKD's ETH0 port and the computer running CM7.
- Connect the NETWORK ports of both Thales devices directly with a network Cat5 cable (it will simulate the protected network line).
- Connect the LOCAL port to the devices meant to communicate via the encryptor (Data Sender / Data Receiver).

CN4020 configuration for encrypting the network traffic

- Enable Line mode
- Change Global Mode to “bypass all”
- Test if traffic flows from Data Sender to Data Receiver

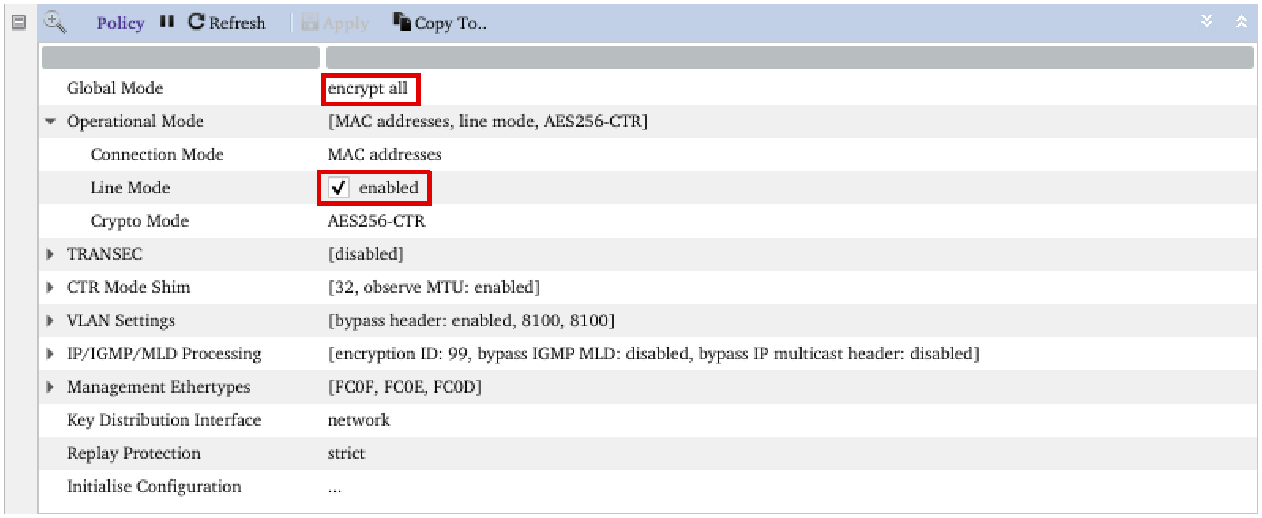


Figure 4 CN4020 Global Policy

- Change Global Mode to “encrypt all”
- Test if traffic still flows from Data Sender to Data Receiver. If not, refer to the HSE documentation for troubleshooting.

Quantum Blockchains pQKD setup

This chapter describes steps to configure pQKD device. On the new device, the configuration panel is available via port ETH0 at http://192.168.1.80 (port 80).

If you want to try to configure the devices right at the beginning, we advise you to go to the **Config Wizard** section first. It allows you to setup the device for operations:

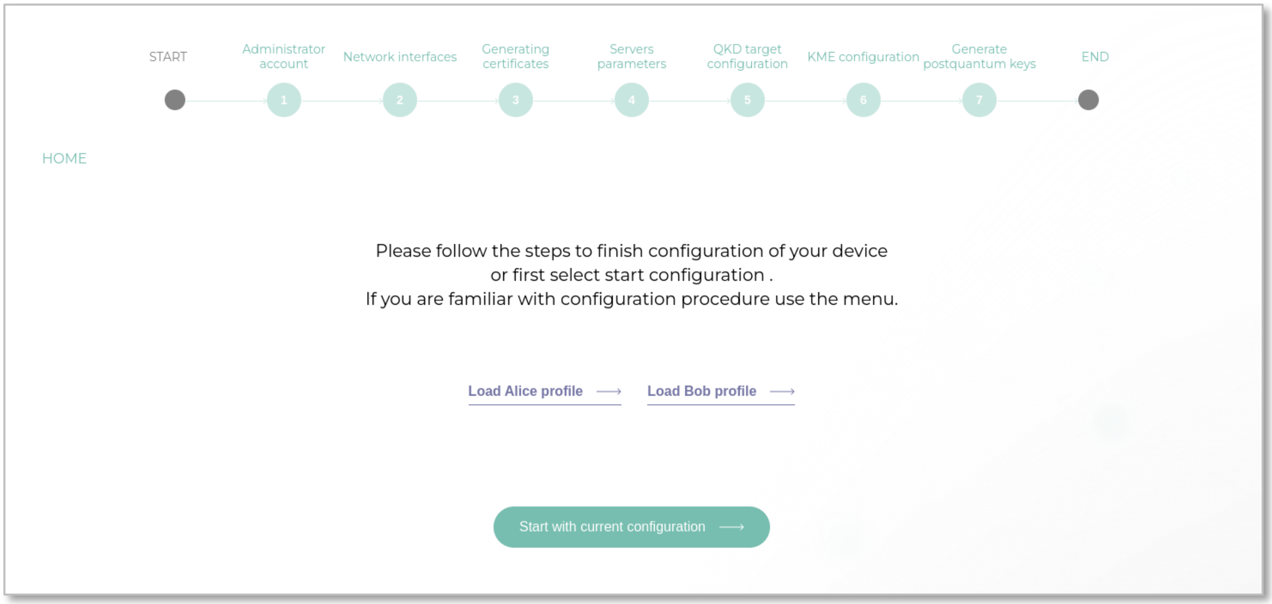


Figure 5 pQKD Config Wizard

You can use the wizard when you first configure the devices and also to quickly change the most relevant parameters of the existing configuration.

To start the configuration process first choose if your current device shall be configured from scratch as “Alice”, as “Bob” or if the wizard shall start from the existing configuration. Use “Load Alice profile”, “Load Bob profile” links or the button “Start with current configuration” respectively. All these actions only load the profiles into the wizard. To make them effective you must always go through the entire wizard and save the configuration.

Administrator account

When you first configure the device for the role of Alice or Bob, or when you do so after factory defaults reset, the administrator credentials will be set to admin/admin.

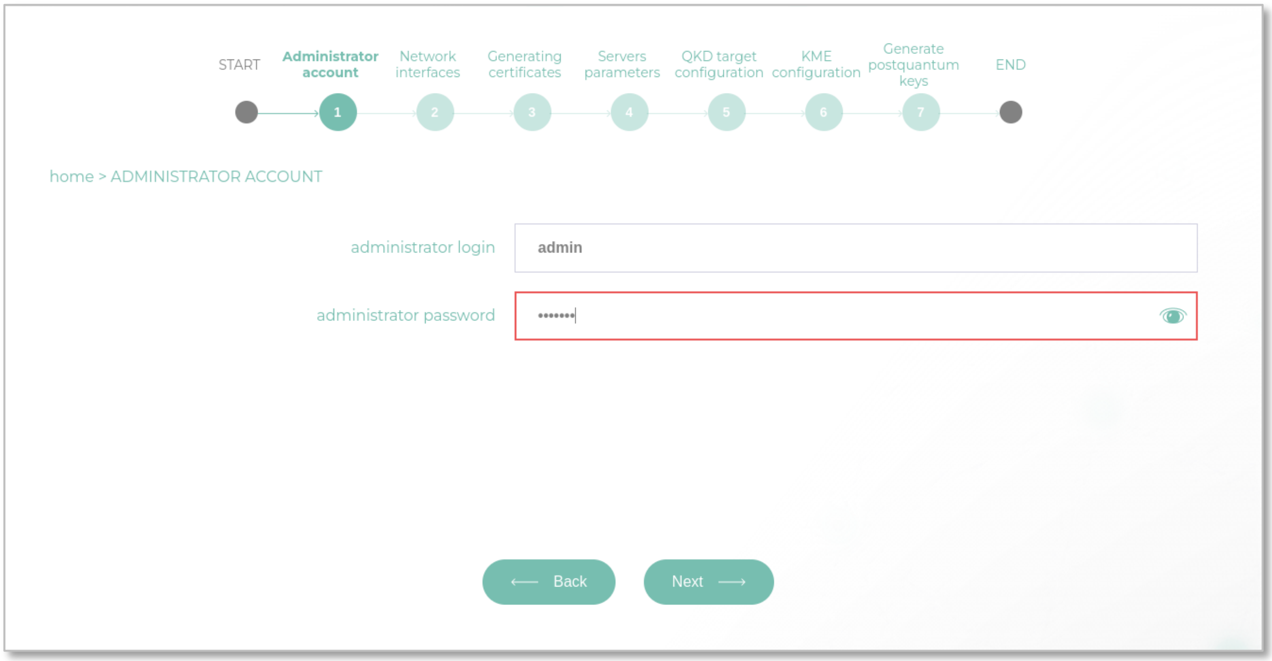


Figure 6 Administrator account

Network interfaces

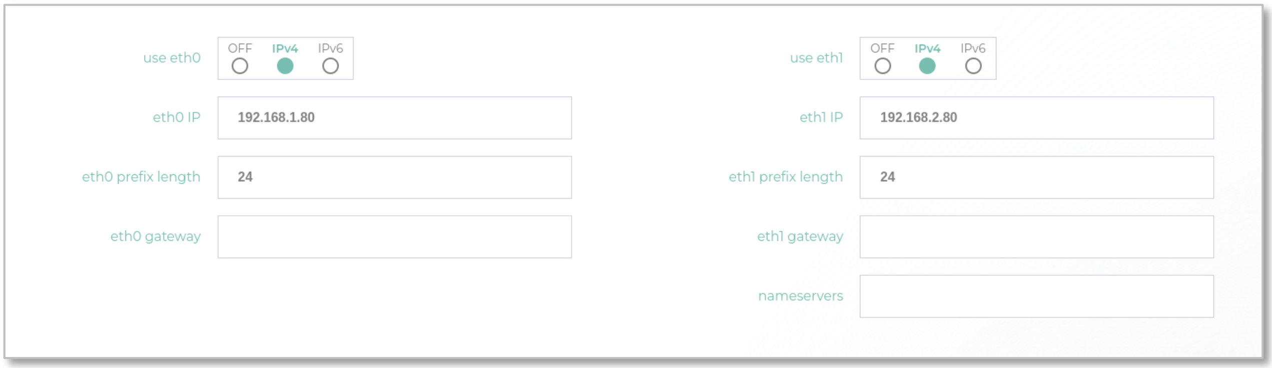


Figure 7 Network interfaces

This wizard section allows you to set up the IP address for the two device network interfaces: ETH0, the interface behind the rightmost LAN port, and ETH1, the interface behind the leftmost LAN port. For each interface, you can choose IPv4 or IPv6 addresses (or switch them off).

The minimum setup requirement is to set the IP address and the netmask using CIDR notation—i.e., the prefix length. If your device communicates with the other end of the pair beyond the local network, you also need to set up the gateway and, in some circumstances, when name resolution is required, the nameserver (DNS) IP addresses.

NOTE: By default, ETH1 is the simulated quantum channel, and ETH0 is the classical (service) channel.

Generating Certificates

In this section we can generate the basic certificates required for secured communication between SAE devices (like HSE) and KME (pQKD devices):

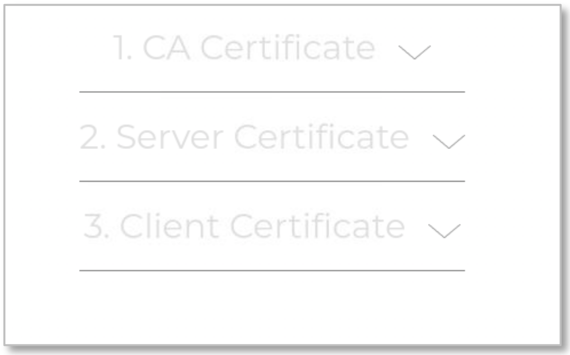


Figure 8 Generating certificates

There are three categories of certificates:

- **CA Certificate** – self-signed (root) certificate you can generate when no external CA is used.
- **Server Certificate** – the certificate of the device when it is in the role of the server (for example, for the ETSI-014 web API calls to KME)
- **Client Certificate** – the certificate for the client (e.g. web browser) access to the device. Initially, you can access the configuration panel via an insecure HTTP mode of communication. Then, after generating the client certificate and loading it to your browser, you can use HTTPS mode.

CA Certificate

The information in this section is required to generate your own Certificate Authority certificate correctly. The data fields describe your organization and its role as CA.

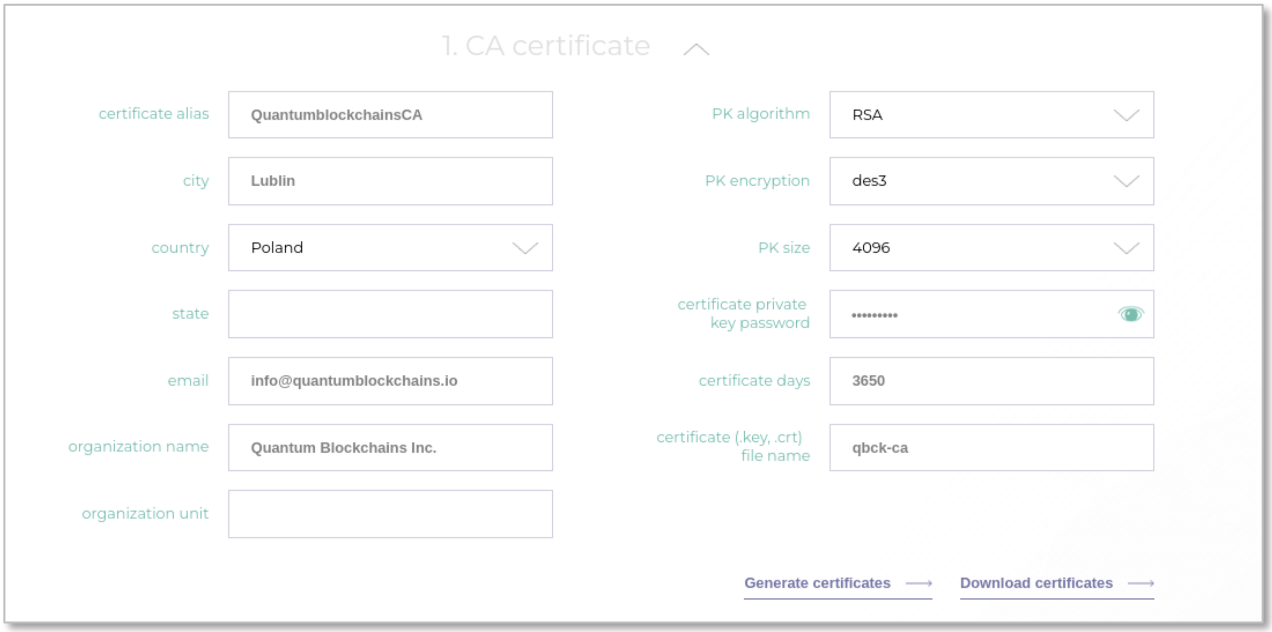


Figure 9 CA Certificate

The following fields specify the certificates details and files which will contain the certificates:

- **certificate alias, city, country, state, email,**
- **organization name** and **organization unit** – standard fields for the CA certificate.
- **PK algorithm** – private key algorithm: for RSA algorithm selected: **PK encryption, PK size**. For the EC algorithm selected: **EC curve**.

- **certificate private key password** - the password for the certificate file.
- **certificate days** – the validity time for the certificate.
- **certificate file name** – file to which the certificate will be stored.

Finally, you can generate and download the certificates using “Generate certificates” and “Download certificates”.

Server Certificate

This section allows for the generation of digital certificates for the server role of the pQKD device:

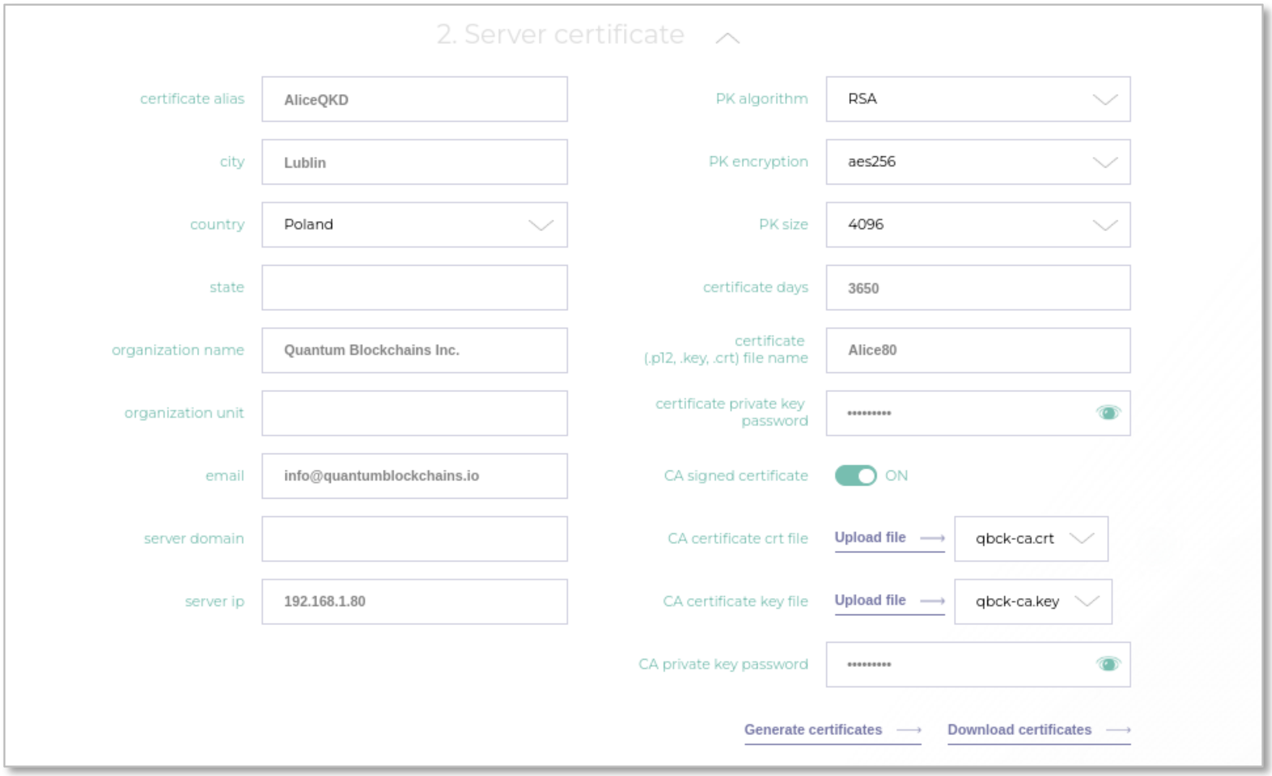


Figure 10 Server Certificate

The first fields from “certificate alias” through “email” are standard data elements required in the certificate generation process. The “**server domain**” name (also known as “Common Name” (CN)) is useful when FQDN is used in communication.

“**Server ip**” is the server's IP address, which is automatically copied here from the IP address of the KME server assigned earlier in the wizard. Other fields:

- **PK algorithm** – private key algorithm: for RSA algorithm selected: **PK encryption, PK size**. For the EC algorithm selected: **EC curve**.
- **certificate days** is the number of days the certificate is valid from the date of its creation.
- **certificate file name** shall be filled with the respective certificate file names.
- **certificate private key password** can be filled with a password that will protect “p12” and “key” file content.
    - If you have an externally signed CA certificate for your server, you can upload its .crt and .key files after switching on the “CA signed certificate” switch. You can add the key file password if it is encrypted.

Finally, you can generate and download the certificates using “Generate certificates” and “Download certificates”, respectively.

Client Certificate

The Client Certificate section is similar to the server certificate section:

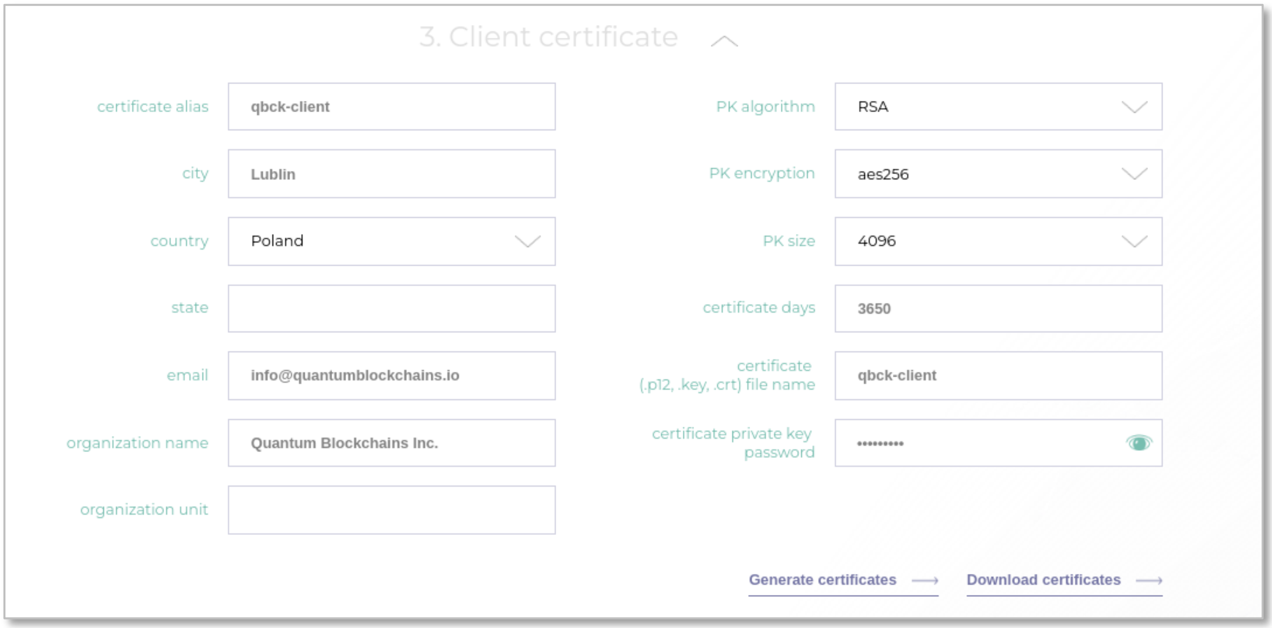


Figure 11 Client Certificate

but it does not contain an IP or domain name as they are irrelevant here, and we can't sign it with a CA certificate (because it is the client certificate).

Setting up servers parameters

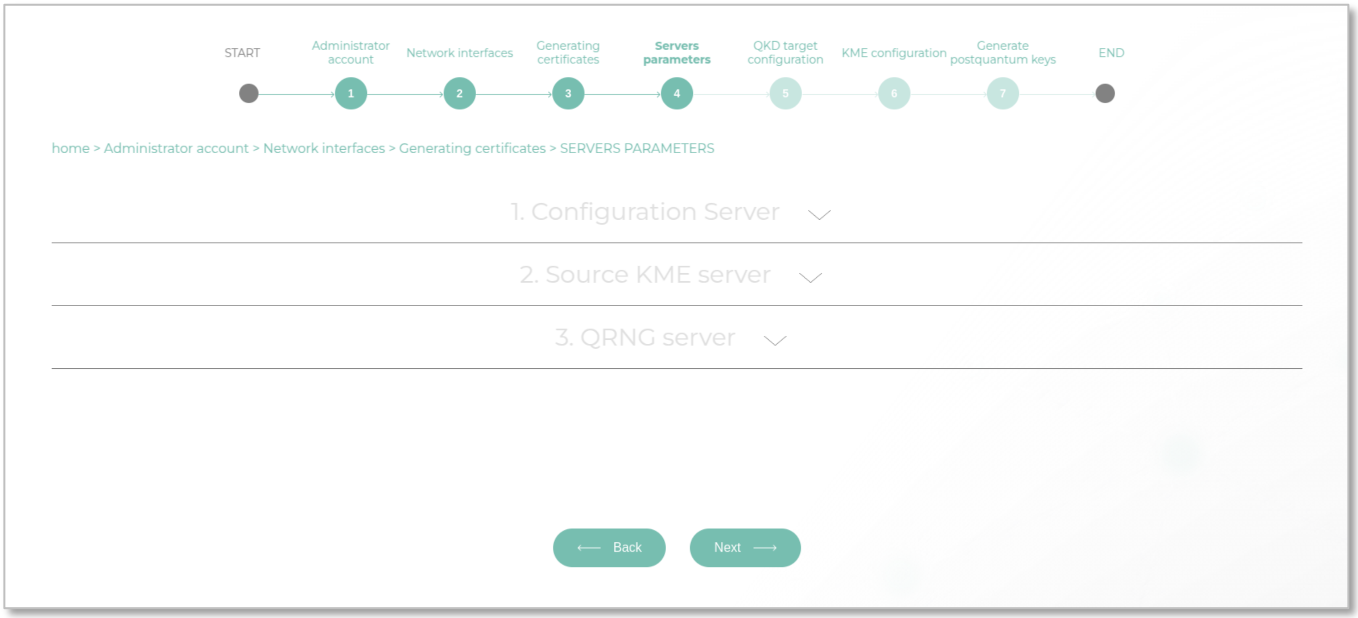


Figure 12. Servers parameters

This section allows to setup the parameters of the three essential servers embedded in the device:

1 **Configuration server** – the webserver responsible for the administrative access to the devices (it servers all the web pages described in this documentation section).

2 **Source KME server** – Key Management Entity server
3 **QRNG Server** – the server responsible for the access to the embedded Quantum Random Number Generator

You open the respective section by clicking on the respective label:

Configuration server:

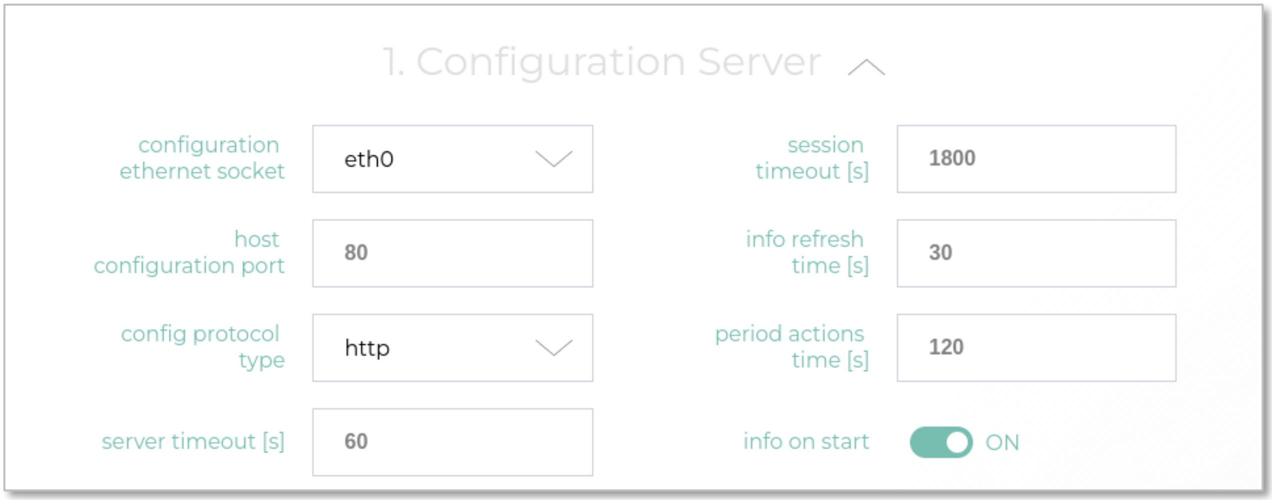


Figure 13 Configuration server

This section we setup the most important network parameters for the server responsible for administrative access:

- **configuration ethernet socket** – allows for the choice of the device ETH port on which access to the configuration server is provided
- **host configuration port** – allows to setup the TCP port number on which the configuration server will be accessed. We recommend using standard 80 TCP port number. The IP address of the service is the one assigned in the previous section to the ethernet port selected here, followed by the port number. For example, in the default configuration, the access to the configuration web service is: http://192.168.1.90.
- **config protocol type** – the choice of server protocol (HTTP or HTTPS). If HTTPS protocol is selected:
  - **configuration server certificate p12 file** – allows to upload server certificate in p12 file format.
  - **configuration server private key password** – the password for the above p12 file
  - **configuration client certificate p12 file** – allows to upload client certificate in p12 file format.
  - **configuration client private key password** – the password for the above p12 file

And for all protocols, we can define:

- **server timeout** – refers to the allowable waiting time for socket responses in many places where data transmission occurs.
- **session timeout** – for the ETSI004 protocol, it's the idle time after which the key exchange session is deleted.
- **Info refresh time** – is the information refresh time given to statistics (information).
- **period actions time** – is the interval for checking system states (key validity, deletion of unnecessary data, etc.), usually set within 1 to 3600 seconds. Setting this time to a higher value may result in extending the lifespan of keys or ETSI004 sessions.
- **info on start** – allowing the disabling of automatic opening of pQKD information upon login – then the main menu opens.

Source KME Server:

From the perspective of general QKD system architecture, pQKD devices implements the KME Server – Key Management Entity Server.

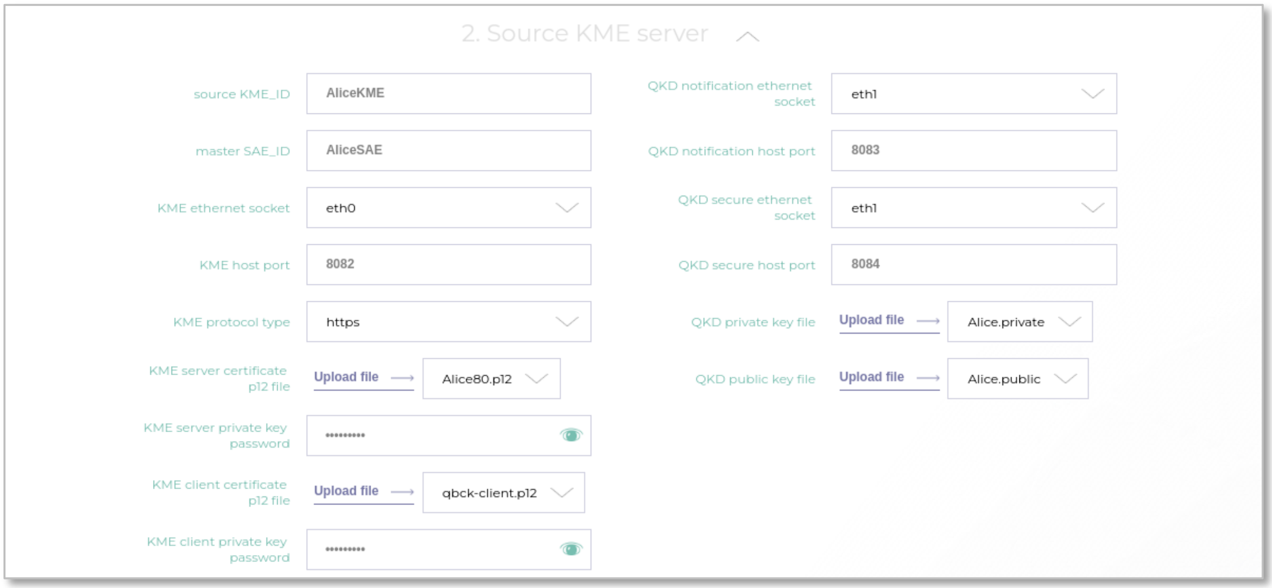


Figure 14 Source KME server

This section allows for the setup of the Source KME Server. We can specify here:

- **source KME_ID –** the identification string of the source KME – Key Management Entity.
- **master SAE_ID –** the identification string for master (current) SAE - Secure Application Entity.
- **KME ethernet socket –** allows to indicate which pQKD LAN socket will be used for KME.
- **KME host port** – allows to specify KME host TCP port.
- **KME protocol type** – allows to choose which protocol KME will use for communication. If HTTPS protocol is selected:
    - **KME server certificate p12 file** – allows to upload server certificate in p12 file format.
    - **KME server private key password** – the password for the above p12 file
    - **KME client certificate p12 file** – allows to upload client certificate in p12 file format.
    - **KME client private key password** – the password for the above p12 file

And for all protocols, we can define:

- **QKD notification ethernet socket** – allows to indicate which pQKD LAN socket will be used for QKD notification communication.
- **QKD notification host port** – allows to specify QKD notification host TCP port.
- **QKD secure ethernet socket** – allows to indicate which pQKD LAN socket will be used for the simulated Quantum channel.
- **QKD secure host port** – allows to specify simulated Quantum channel host TCP port.
- **QKD private key file** – allows to upload the private PQC key (needed when it is required to go beyond default PQC keys)
- **QKD public key file** – allows to upload the public PQC key (needed when it is required to go beyond default PQC keys)

QRNG Server:

As the pQKD implements real Quantum Random Number Generator that can be used independently of QKD emulation function, this section allows to set it up.

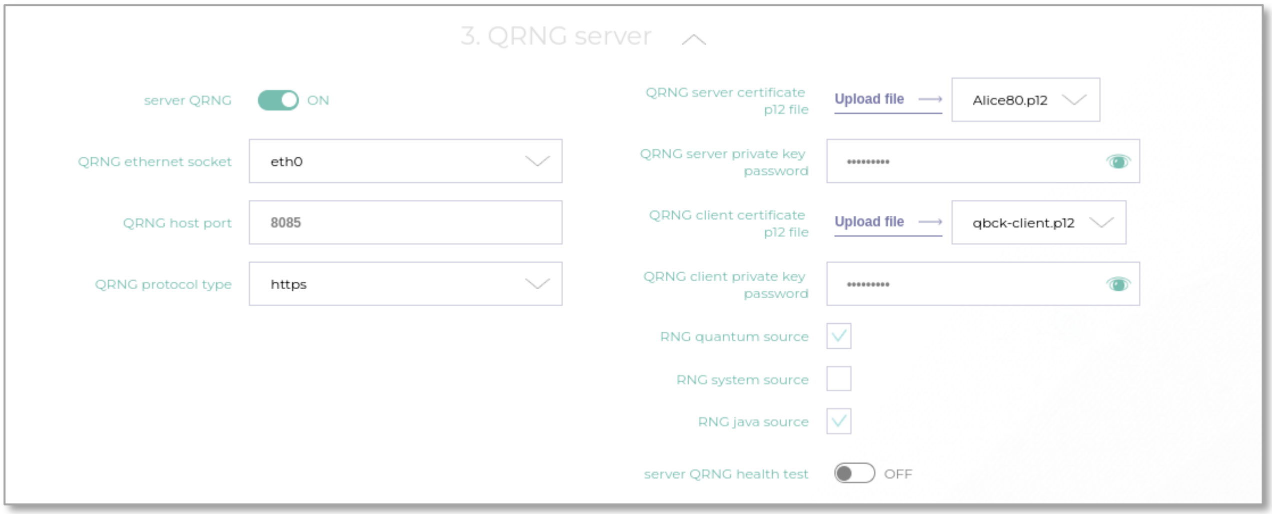


Figure 15 QRNG server

You can switch it on and off to enable or disable **the QRNG server.** You can also order the device system software to perform a QRNG health test. Tests are performed at every system startup. Next, you can choose the **ethernet socket** to access the QRNG. The QRNG server's IP address is the one assigned in the previous section to the ethernet port selected here, followed by the **port number**.

The three checkboxes, **RNG quantum source**, **RNG system source**, and **RNG Java source**, specify the sources of entropy for the mixed entropy model of the msQRNG type of entropy generation used by our device.

QKD Target Configuration

In this section of the wizard, you can set up the target addresses and ports for communication between devices of the simulated QKD pair. If the pQKD device works in Multi Target variant, this section displays the list of all target KMEs and their associated SAEs:

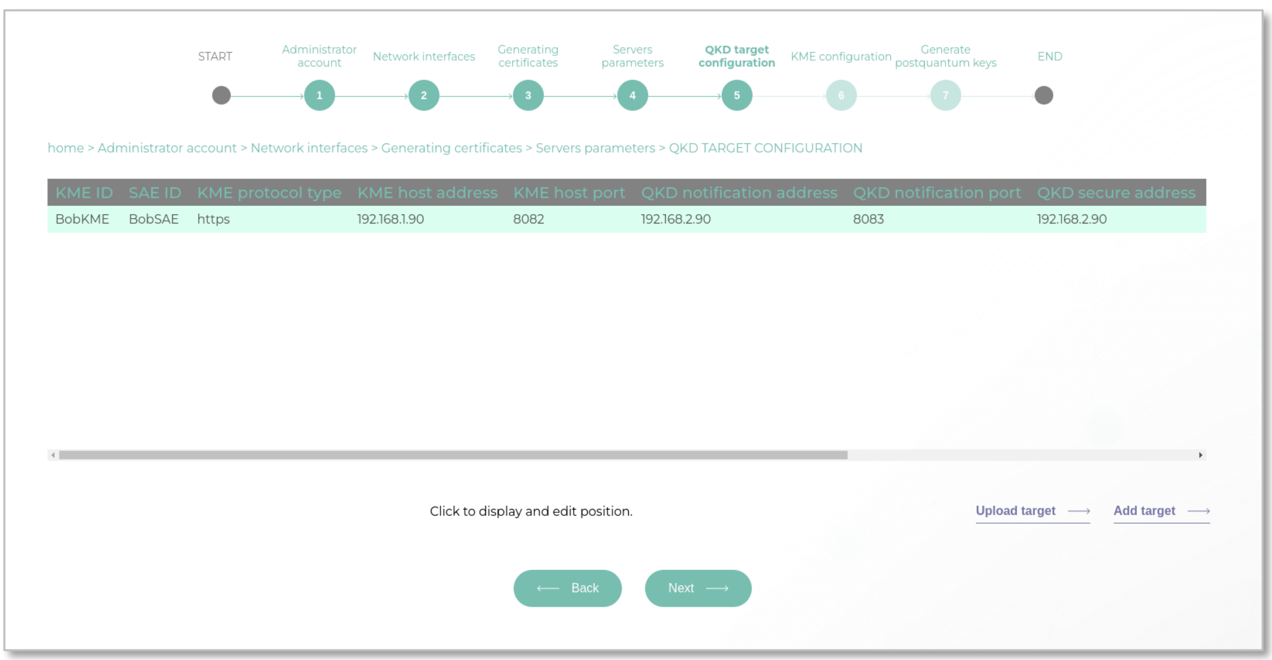


Figure 16 QKD Target Configuration

The **Upload Target** button allows you to upload the target definition created on the target pQKD (in QKD Target configuration in the main menu of the target pQKD). The **Add Target** button allows you to add a target manually.

Note: If pQKD works in standard configuration it shows the single Target definition, as seen on Figure 17**):**

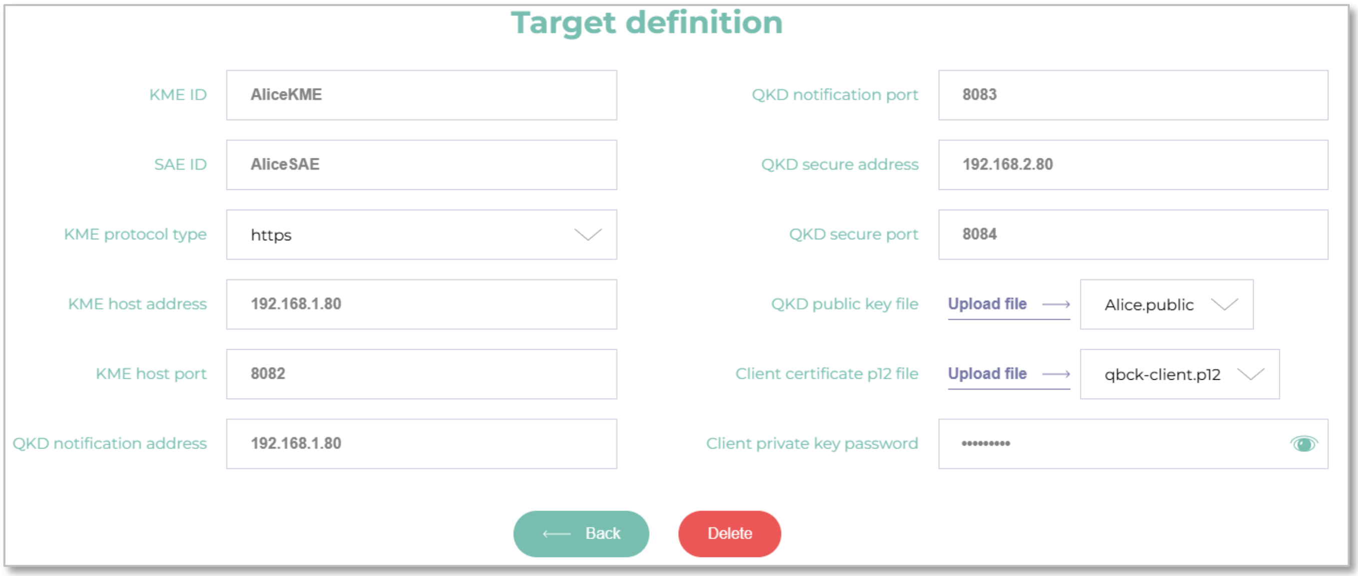


Figure 17 Target Definition

By clicking on the respective line showing the target, we enter the target definition:

- **KME ID** – Target KME Identifier
- **SAE ID** – Target SAE Identifier
- **KME protocol type** – choice of HTTP or HTTPS as the communication protocol between KMEs
- **KME host address** – Target KME host TCP address
- **KME host port** – Target KME host TCP port
- **QKD notification address** – Target QKD notification host TCP address (i.e. the classical channel in QKD)
- **QKD notification port** – Target QKD notification host TCP port (i.e. the classicals channel in QKD)
- **QKD secure address** – Target QKD secure communication host TCP address (emulates quantum channel in QKD)
- **QKD secure port** – Target QKD secure communication host TCP port (emulates quantum channel in QKD)
- **QKD public key file –** the upload of the public key for post-quantum KYBER KEM algorithm (it is created in “Generate Post-Quantum Keys” of the menu on the target pQKD.
- **Client certificate p12 file** – the upload of the (classical) certificate of the target pQKD (it is created in “Generating certificates | Client certificates” on the target pQKD. It works for https communication.
- **Client private key password** – the password for the certificate p12 file.

KME configuration

In this section of the wizard we can setup the basic parameters of the Key Management Entity:

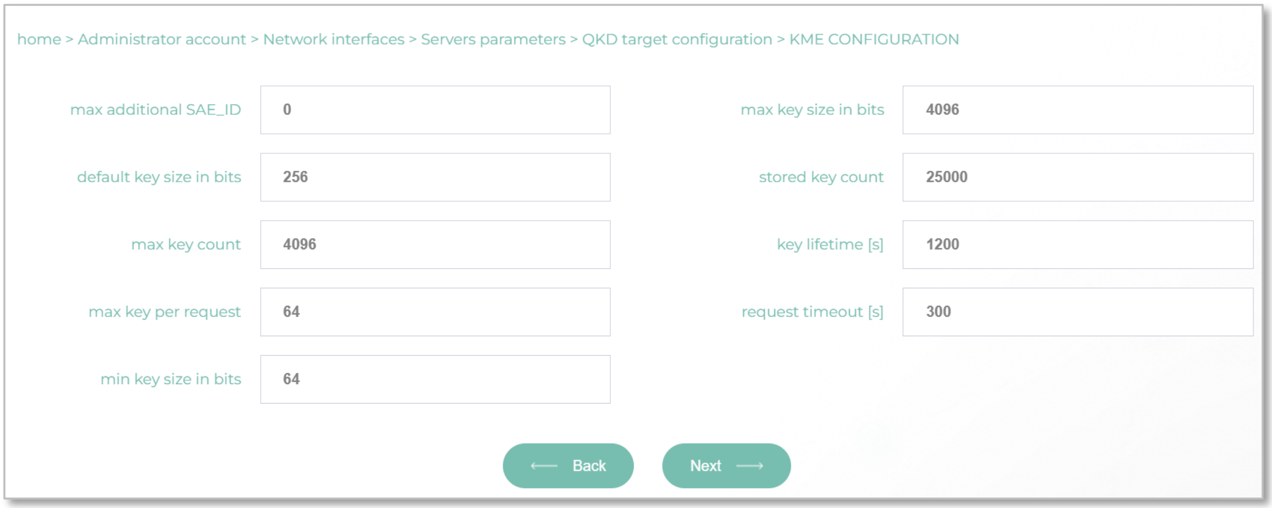


Figure 18 KME configuration

- **max additional SAE_ID** - A field not used by pQKD, introduced to maintain compliance with the ETSI014 standard for info queries. We set it to 0.
- **default key size in bits** – the default size of the secret QKD key being transmitted over the Simulated Quantum Channel (corresponds to QKD key being generated on Alice and Bob)
- **max key count** – the maximum number of keys stored in memory.
- **max key per request** - the maximum number of keys per single request
- **min key size in bits** – the shortest key size that can be requested
- **max key size in bits** – the longest key size that can be requested
- **stored key count** – the current number of keys stored in memory (generated but not yet retrieved by ETSI 014 dec_keys call).
- **key lifetime [s]** – the key lifetime
- **request timeout [s]** – the maximum allowable data transmission time from the client to the pQKD (from the arrival of the header to the end of the post).

Post Quantum Configuration

In this section we can indicate where the private and public post-quantum keys used by KYBER (KEM – Key Encapsulation Mechanism type algorithm) will be stored:

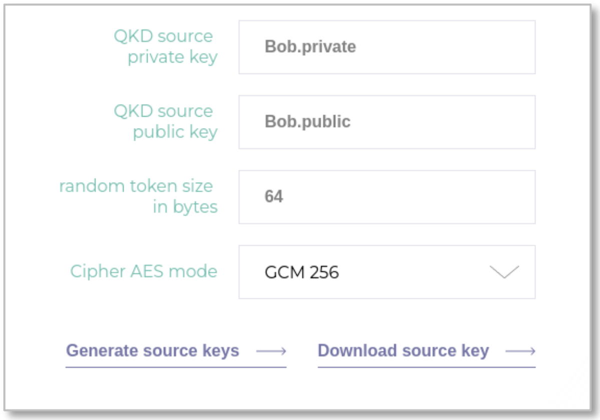


Figure 19 Postquantum Configuration

- **QKD source private key** – is the file name in which KYBER private key will be stored. The file is a binary file locally stored on the pQKD device.
- **QKD source public key** – is the file name in which KYBER public key will be stored. The file is a binary file locally stored on the pQKD device.

We can transfer the generated KYBER certificate files to another computer by downloading and uploading the file. Such a transfer of PQC keys is required only when new keys need to be generated—by default, the corresponding pairs are present on the devices to be paired.

- **random token size in bytes –** this is the length of the signed random token to ensure the correctness and authorization of the simulated QKD key exchange.

In this section, we have also placed the choice of symmetric algorithm for key encryption:

- **Cipher AES mode –** allows for the choice of the following symmetric encryption modes: GCM 256 (Galois/Counter Mode with 256 bit keys), CBC 256 (Cipher Block Chaining with 256 bit keys) and ECB (Electronic Code Book mode) with key lengths: 256, 512, 1024, 2048 and 4096.

By pressing “Generate source keys” tehpQKD generates all four PQC key files:

Certificate files

Name	Length	Date
Bob.private	3168	2023-07-23 20:03:54
Bob.public	1568	2023-07-23 20:03:54
Alice.private	3168	2023-07-21 10:08:23
Alice.public	1568	2023-07-21 10:08:23

Figure 20 Certificate files

And by pressing “Download source key” we can download the current public key which can be uploaded on the another device.

End of wizard

The last section of the wizard allows for Saving the entire configuration:

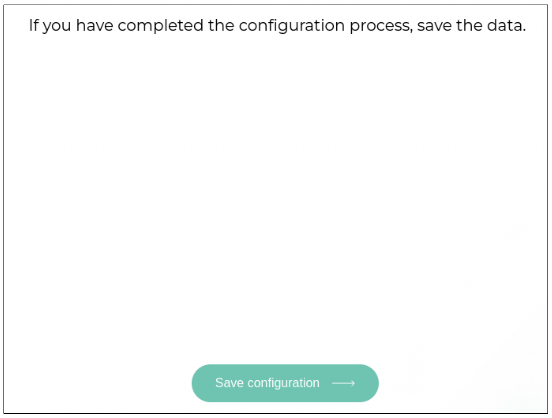


Figure 21 End of wizard

After completion of the entire configuration, your device is ready to work!

Configure HSE with pQKD for quantum key distribution

This chapter describes steps to configure ETSI QKD interface using HTTPS connection.

Using CM7, enable ESTI QKD in both encryptors:

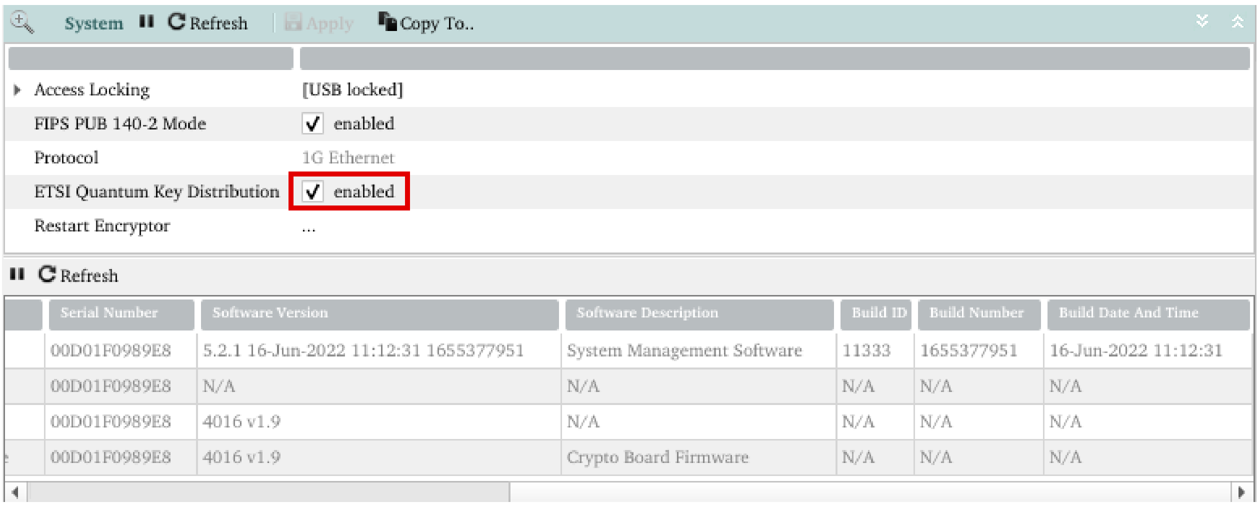
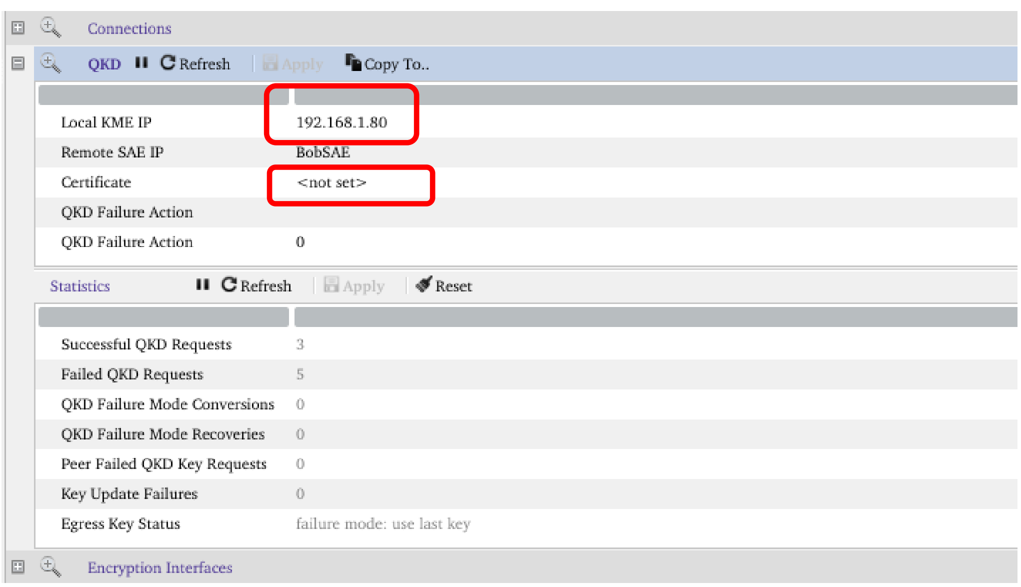


Figure 22 ETSI QKD Settings in HSE

Using CM7, enter the IP address of KME in QKD Section:



NOTE. The encryptor does not explicitly indicate a port for KME in QKD mode. If the certificate field shows `<no set>`, the default port is 80 (HTTP); if a certificate is loaded, then the port is 443 (HTTPS).

To use an HTTPS connection, you must upload the pQKD device certificate using "Import PEM" in the Certificate section:

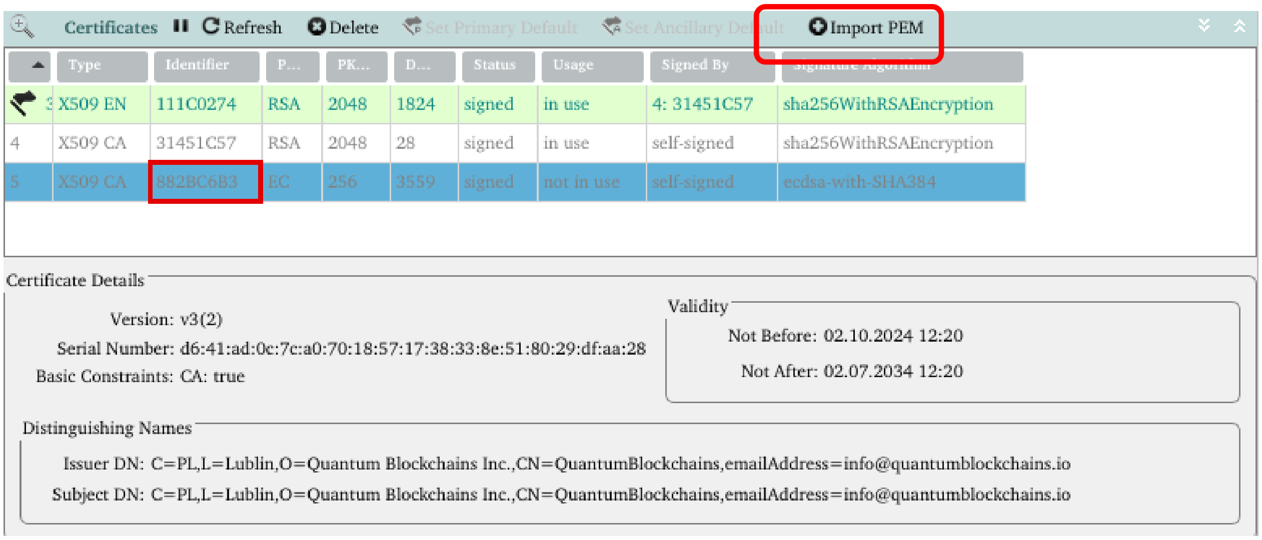


Figure 23. Importing certificates into HSE

Please note. The encryptors only support TLS v1.2 and ECDSA certificates. You can upload either the full pQKD (KME) server certificate or just the CA certificate that signed the pQKD server certificate (If it is CA signed certificate). After importing the certificate, select the loaded certificate identifier in the QKD section under Certificate section:

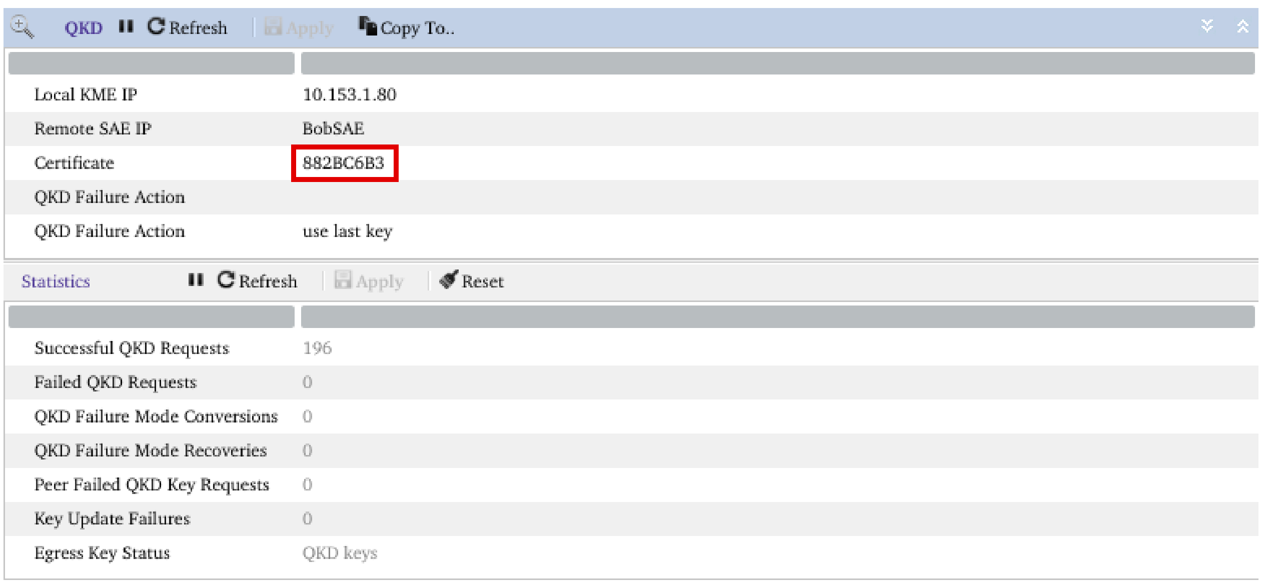
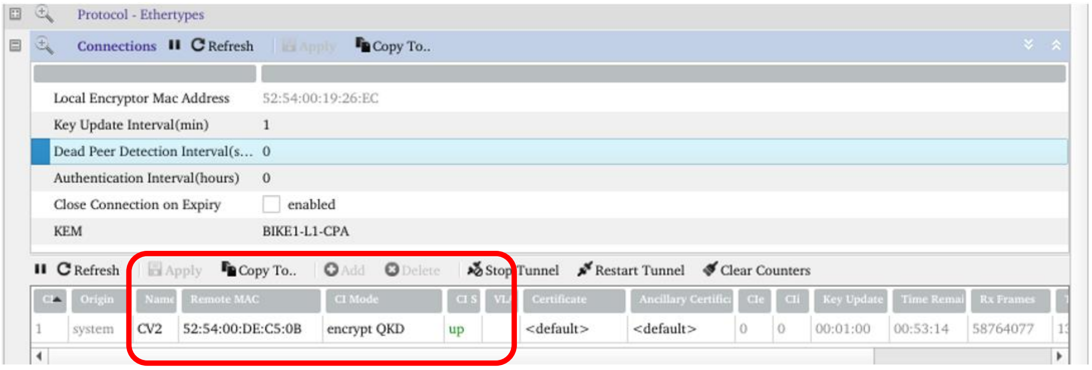


Figure 24 Selecting certificate in QKD Settings

Once configured, the connection to pQKD should look like this in Connection section (for both encryptors):



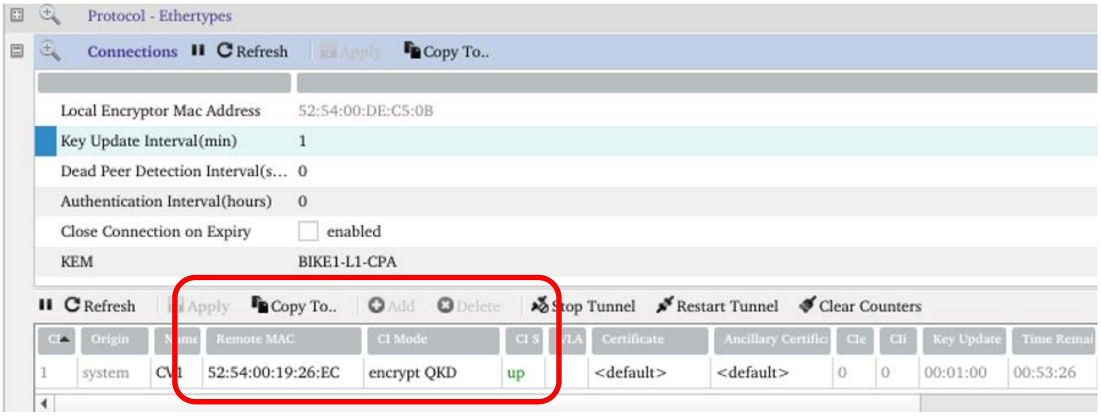


Figure 25 Connection to KEM established

On the pQKD devices, the logs shall show:

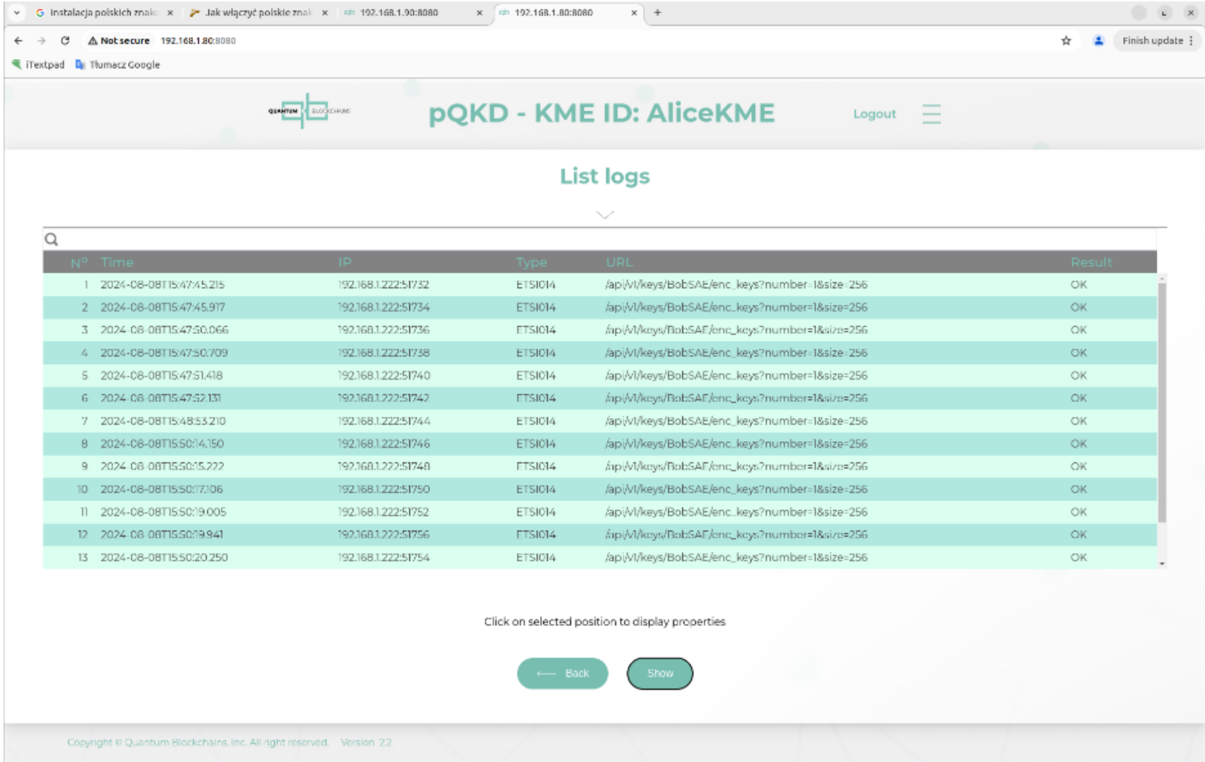


Figure 26 Logs view on AliceKME

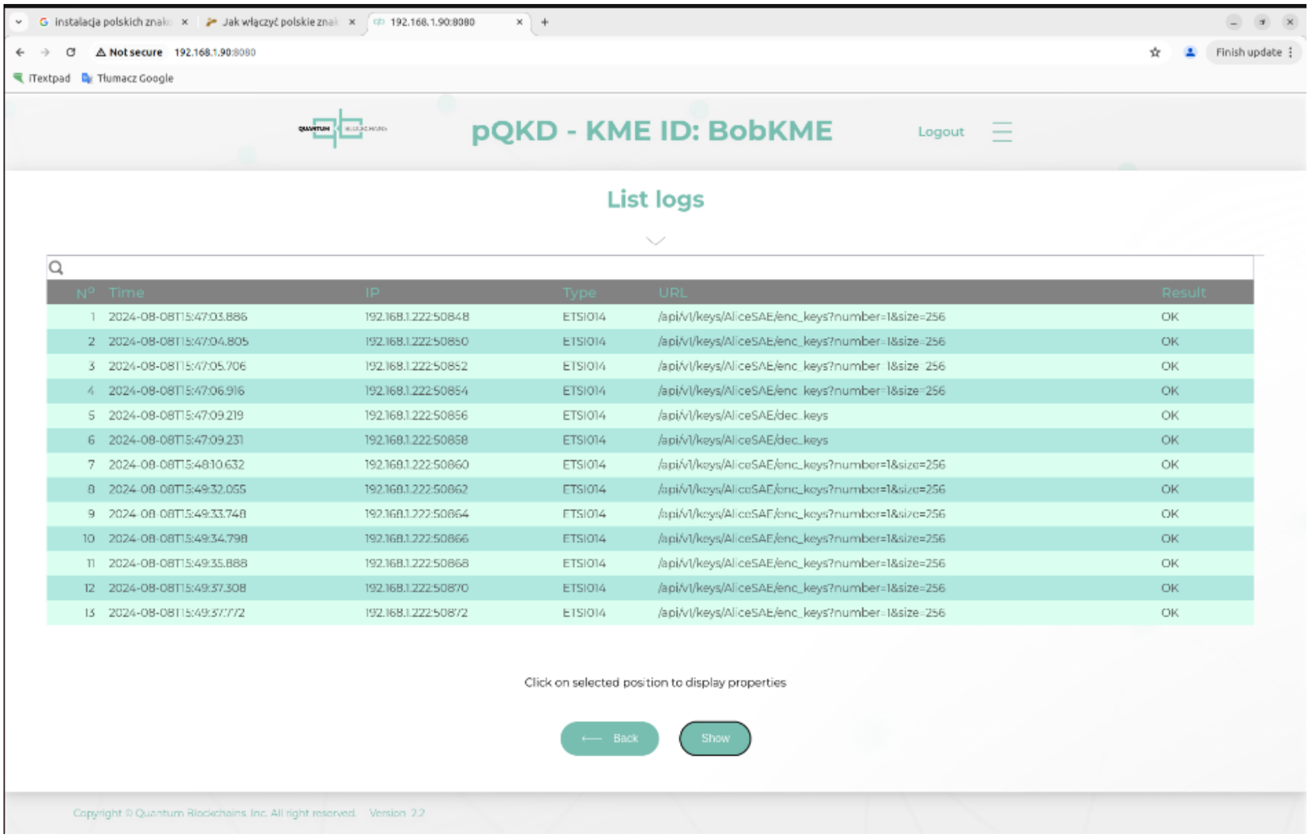


Figure 27 Logs view on BobKME

Tests

For the purpose of testing, two physical **Thales HSE emulators** were deployed and integrated with two **pQKD (Post-Quantum Key Distribution) devices**, configured as standard **Alice** and **Bob** roles, as illustrated in **Figure 3**.

Key Exchange Configuration

- The **key exchange frequency** was set to **1 minute**, which represents the **minimum possible interval** supported by the system.
- This configuration ensured frequent key refresh cycles, allowing comprehensive validation of the key exchange process under continuous load conditions.

Network Communication Setup

- Secure communication channels were established between the devices using encryption facilitated by the Thales HSE emulators and pQKD devices.
- Standard network protocols were utilized for validation, including:
  - **SSH (Secure Shell)** for remote command-line access and configuration verification.
  - **HTTP** for application-level data exchange.
  - **File Transfer Protocols (e.g., SCP/SFTP)** for secure data transfer across the encrypted links.

Test Duration and Conditions

- The network connection was continuously monitored and tested over a **multi-day period** to assess the reliability and stability of the key exchange process.
- During the testing phase, traffic patterns were deliberately varied to simulate typical operational loads and ensure real-world applicability.

Test Results

- No **key exchange errors** or **communication disruptions** were observed during the entire testing period.
- The encryption and key management systems consistently performed as expected, maintaining data integrity and security across all communication protocols.

This testing phase confirmed the robustness of the key exchange mechanism and validated the integration between the Thales HSE emulators and pQKD devices under sustained operational conditions.

Final Notes

Integrating pQKD (Quantum Key Distribution emulator) with Thales HSE (High-Speed Encryption) is straightforward.

Although the guide covers integrating pQKD with CN4020, we also successfully integrated and tested CV1000 (virtual encryption) with pQKD.

By combining pQKD with HSE, organizations can test and validate QKD technologies within existing network environments without incurring the high costs and logistical challenges of deploying physical QKD systems.

The HSE further ensures high-assurance encryption, making the integration a practical and effective solution for future-proofing security.

