QUANTUMXCHANGE

Phio TX version 4.5.6

Common Criteria Security Target

Version 1.7
28/09/2026

Prepared by:

Digital Cubes S.L.
www.digital-cubes.com
Digital Cubes

DOCUMENT HISTORY

Version	Date	Description
0.1	01/07/2025	Initial draft
1.0	01/10/2025	First release
1.1	09/12/2025	Solving CPSTIC validation
1.2	10/03/2026	Solving ITSEF comments
1.3	31/03/2026	Solving ITSEF comments
1.4	08/04/2026	Solving ITSEF comments and updating TOE and deliverables versions
1.5	04/05/2026	Update SSH HMAC-SHA algorithm and cryptographic key destruction method
1.6	26/08/2026	Solving ITSEF comments
1.7	28/09/2026	Solving ITSEF comments

TABLE OF CONTENTS

1 Security Target Introduction 7

1.1 ST reference 7

1.2 TOE reference 7

1.3 TOE overview 7

1.3.1 TOE type 7

1.3.2 TOE Usage and Major Security Features 7

1.3.3 Non-TOE Hardware/Software/Firmware 910

1.4 TOE Description 10

1.4.1 Physical Scope 10

1.4.2 Logical Scope 1011

2 Conformance claims 13

2.1 Common Criteria conformance claim 13

2.2 Protection Profile conformance 13

2.3 Package claim 13

2.4 Conformance Rationale 13

2.5 Evaluation Methods and Activities 13

3 Security Problem Definition 14

3.1 Threats 14

3.1.1 T.UNAUTHORIZED_ADMINISTRATOR_ACCESS 14

3.1.2 T.WEAK_CRYPTOGRAPHY 14

3.1.3 T.UNTRUSTED_COMMUNICATION_CHANNELS 14

3.1.4 T.WEAK_AUTHENTICATION_ENDPOINTS 14

3.1.5 T.UPDATE_COMPROMISE 14

3.1.6 T.UNDETECTED_ACTIVITY 14

3.1.7 T.SECURITY_FUNCTIONALITY_COMPROMISE 15

3.1.8 T.PASSWORD_CRACKING 15

3.1.9 T.SECURITY_FUNCTIONALITY_FAILURE 15

3.1.10 T.KME_LINK_DISCLOSURE 15

3.1.11 T.CRYPTO_KEY_COMPROMISE 15

3.2 Organizational Security Policies (OSP) 15

3.2.1 P.ACCESS_BANNER 15

3.3 Assumptions 15

3.3.1 A.PHYSICAL_PROTECTION 16

3.3.2 A.LIMITED_FUNCTIONALITY 16

3.3.3 A.NO_THRU_TRAFFIC_PROTECTION 16

3.3.4 A.TRUSTED_ADMINISTRATOR 16

3.3.5 A.REGULAR_UPDATES 16

3.3.6 A.ADMIN_CREDENTIALS_SECURE 17

3.3.7 A.RESIDUAL_INFORMATION 17

3.3.8 A.OPERATIVE_SYSTEM 17

4 Security objectives 18

4.1 Security Objectives for the TOE 18

4.1.1 O.TOE_ADMINISTRATION ... 18
4.1.2 O.STRONG_AUTHENTICATION ... 18
4.1.3 O.PROTECTED_COMMUNICATIONS ... 18
4.1.4 O.PROTECTED_KME_COMM ... 18
4.1.5 O.STRONG_CRYPTOGRAPHY ... 18
4.1.6 O.UPDATES ... 18
4.1.7 O.AUDIT ... 18
4.1.8 O.TSF_SELF_TEST ... 18

4.2 Security objectives for the Operational Environment ... 18
4.2.1 OE.PHYSICAL ... 19
4.2.2 OE.NO_GENERAL_PURPOSE ... 19
4.2.3 OE.NO_THRU_TRAFFIC_PROTECTION ... 19
4.2.4 OE.TRUSTED_ADMIN ... 19
4.2.5 OE.UPDATES ... 19
4.2.6 OE.ADMIN_CREDENTIALS_SECURE ... 19
4.2.7 OE.RESIDUAL_INFORMATION ... 19

4.3 Security objectives rationale ... 19
4.3.1 Security objectives for the TOE ... 19
4.3.2 Security objectives for the Operational Environment ... 21

5 Extended components definition ... 23

5.1 Security Audit (FAU) ... 23
5.1.1 Security Audit Data Generation (FAU_STG_EXT) ... 23

5.2 Cryptographic support (FCS) ... 24
5.2.1 SSH Server Protocol (FCS_SSHS_EXT) ... 24
5.2.2 TLS Client Protocol (FCS_TLSC_EXT) ... 25
5.2.3 TLS Server Protocol (FCS_TLSS_EXT) ... 27
5.2.4 Random Bit Generation (FCS_RBG_EXT) ... 29

5.3 Identification and Authentication (FIA) ... 30
5.3.1 Password Management (FIA_PMG_EXT) ... 30
5.3.2 Password Management (FIA_UIA_EXT) ... 31
5.3.3 Password Management (FIA_UAU_EXT) ... 32
5.3.4 Authentication using X.509 certificates (FIA_X509_EXT) ... 32

5.4 Protection of the TSF (FPT) ... 34
5.4.1 Protection of TSF Data (FPT_SKP_EXT) ... 34
5.4.2 Protection of Administrator Passwords (FPT_APW_EXT) ... 35
5.4.3 TSF Self-Test (FPT_TST_EXT) ... 35
5.4.4 Trusted Update (FPT_TUD_EXT) ... 36
5.4.5 Time stamps (FPT_STM_EXT) ... 38

5.5 TOE Access (FTA) ... 38
5.5.1 TSF-initiated Session Locking (FTA_SSL_EXT) ... 38

6 Security Requirements ... 40

6.1 Security Functional Requirements (SFRs) ... 40
6.1.1 Security Audit (FAU) ... 40
6.1.2 Cryptographic Support (FCS) ... 43
6.1.3 Identification and Authentication (FIA) ... 46
6.1.4 Security Management (FMT) ... 48
6.1.5 Protection of the TSF (FPT) ... 49

6.1.6 TOE Access (FTA) 50
6.1.7 Trusted path/channels (FTP) 50
6.1.8 SFR Dependencies Analysis 51

6.2 Security Assurance Requirements (SARs) 54

6.3 Security requirements rationale 55
6.3.1 Security Functional requirements (SFR) Rationale 55
6.3.2 Security Assurance requirements (SAR) Rationale 57

7 TOE Summary specification 58

7.1.1 Security Audit 58
7.1.2 Cryptographic Support 59
7.1.3 Identification and Authentication 60
7.1.4 Security Management 61
7.1.5 Protection of the TSF 62
7.1.6 TOE Access 63
7.1.7 Trusted Path/Channels 64

8 References and Abbreviations 65

8.1 References 65

8.2 Abbreviations 65

LIST OF TABLES

Table 1: Security Target reference 7
Table 2: TOE reference 7
Table 3: Components of the environment 10
Table 4. TOE delivery items 1011
Table 5. Security objectives for the TOE traceability matrix 21
Table 6. Security objectives for the Operational Environment traceability matrix 22
Table 7: Security Functional Requirements and Auditable Events 42
Table 8: Security Assurance Requirements 55
Table 9. Mapping between SFRs and TOE objectives 56

1 SECURITY TARGET INTRODUCTION

This section specifies the Security Target (ST) and the Target of Evaluation (TOE) and provides a short description of the TOE and its security properties.

1.1 ST reference

Title	Phio TX version 4.5.6, Common Criteria Security Target
Version	1.7
Date	28/09/2026
Author	Digital Cubes

Table 11: Security Target reference

1.2 TOE reference

TOE Developer	Quantum Xchange, Inc.
TOE Name	PhioTX-V
TOE Version	4.5.6

1.3 Table 22: TOE referenceTOE overview

The Target of Evaluation (TOE) is the Phio Trusted Xchange™ PhioTX-V (hereafter referred to as Phio TX), a virtualized network device whose function is to distribute quantum-safe symmetric keys as an enterprise service.

The TOE separates the key-management plane from the data plane, supports multiple post-quantum KEM algorithms (including ML-KEM, BIKE, Classic-McEliece, FrodoKEM, HQC, SNTRUP), and can be configured in “Hives” to share symmetric cryptographic keys.

The TOE integrates seamlessly atop existing HSMs, encryptors, VPNs, and QKD systems without requiring rip-and-replace, preserving network devices while upgrading to quantum-safe operations. Comprehensive audit logging (syslog) ensure that the administration and key exchange remain auditable.

1.3.1 TOE type

The TOE is A Key Management Entity (KME), as defined in [ETSIQKD], is a central component in a Quantum Key Distribution (QKD) network. It is responsible for managing cryptographic keys and cooperating with other KMEs across a network to ensure secure key distribution with post-quantum cryptography protection.

1.3.2 TOE Usage and Major Security Features

The Target of Evaluation (TOE) functions as a Key Management Entity (KME) within a quantum-safe key distribution infrastructure. The TOE exposes a REST-based key delivery API conforming to ETSI GS QKD 014 and Cisco SKIP API", enabling Secure Application Entities (SAEs)—such as optical switches, encryption modules, and security management systems—to request and retrieve symmetric cryptographic keys for secure communication.

In this architecture, SAEs are client devices or applications authorized to interact with the TOE to obtain cryptographic material. Each SAE is uniquely identified by a Secure Application Entity Identifier (SAE ID), and each TOE instance is identified by a unique KME ID. These identifiers are required for authentication and routing of key delivery within the QKD network.

SAEs are categorized into two roles during key retrieval: the **Master SAE** and the **Slave SAE**. The Master SAE initiates the key acquisition process by issuing a "Get key" request to the TOE, specifying the identity of the corresponding Slave SAE. At this point, the TOE generates the symmetric key material internally using a FIPS-validated cryptographic module, ensuring compliance with high-assurance security standards. The TOE then assigns universally unique Key IDs to the generated keys. The Master SAE subsequently communicates these Key IDs to the designated Slave SAE, this communication and its protection is out-of-scope of this Security Target.

The Slave SAE subsequently performs a "Get key with key IDs" operation to retrieve the same symmetric key material from the TOE by referencing the Key IDs previously provided by the Master SAE. This master-slave key request mechanism ensures that both endpoints receive identical key material for establishing a secure channel, while maintaining strict delivery control and traceability of each key instance. Keys are transmitted only once and are destroyed from memory after delivery; they are never written to disk.

The TOE is deployed within a Trusted Node, a secure site that contains one or more SAEs and at least one TOE instance. Communications between the TOE and its connected SAEs occur within this trusted perimeter using

SKIP or ETSI secured by TLS 1.2 or higher. The TOE authenticates each request and enforces mutual TLS authentication, verifying the identity of the requesting SAE using its certificate and SAE ID.

Each TOE maintains synchronization with its peer KMEs (i.e. other TOE instance in another site) through pre-established links. These links ensure that key material generated or received by one TOE can be securely relayed and referenced by others. This exchange of keys between KMEs is considered unprotected against post-quantum cryptography attack so this channel supports multiple post-quantum KEM algorithms (including ML-KEM, BIKE, Classic-McEliece, FrodoKEM, HQC, SNTRUP) to provide this protection.

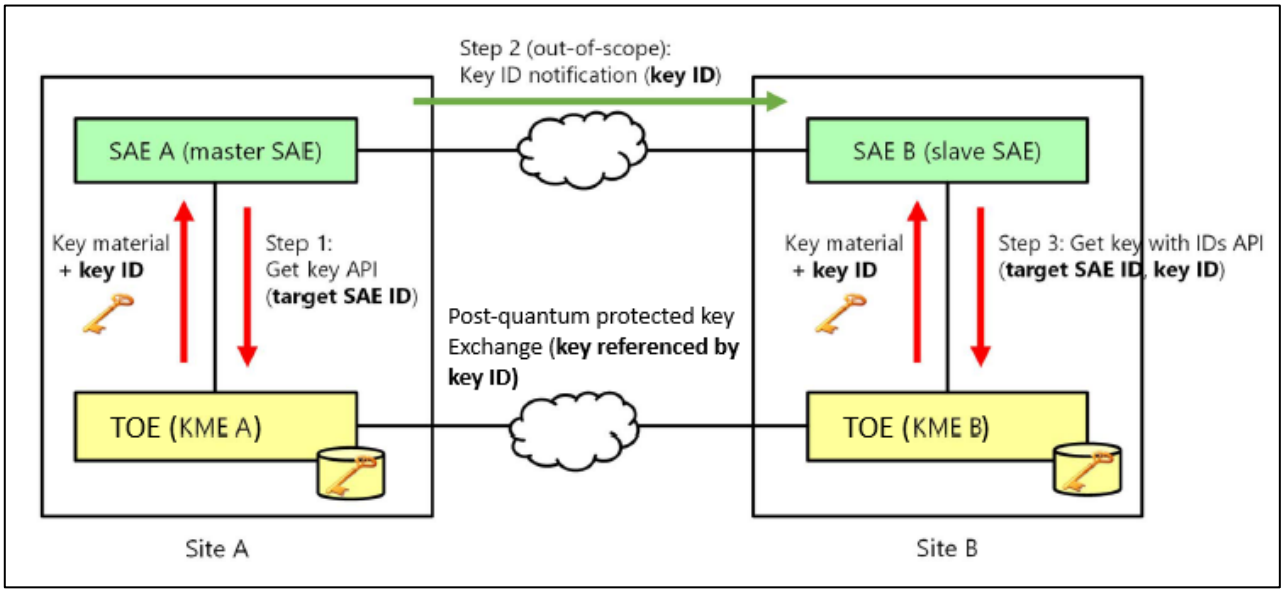


Figure 11. Use-case of key delivery

The TOE provides the following security functionality:

- Security audit.
    - Transmit audit data to external entities using a trusted channel.
    - Store audit data locally on the TOE.
    - Overwrite oldest records in local storage when full to maintain continuous logging.
- Cryptographic support.
    - Generate, establish, and destroy cryptographic keys.
    - Perform encryption, decryption, signature, and hashing operations.
    - Utilize cryptography for securing communication channels and firmware updates.
- Identification and authentication.
    - Provides password-based authentication for administrators.
    - Authentication required before any Command Line Interface (CLI) interaction.
    - Configuration includes management of password policies and number of authentication failures.
- Security management.
    - Trusted Update mechanism.
    - Restriction of TSF management to the Security Administrator.
    - Local and remote administration of the TOE by the Security Administrator.
- Protection of the TSF.
    - Prevents disclosure of pre-shared, symmetric, private key and passwords.
    - Conducts self-tests on start-up to ensure software integrity.
    - Enables administrators to manage firmware/software updates using digital signatures for integrity check.
    - Provides reliable time stamps by its internal system clock.
- TOE access.
    - Displays a Security Administrator-specified advisory banner before establishing an administrative user session.
    - Allows administrators to initiate termination of their own interactive sessions.
    - Automatically terminates interactive sessions after a Security Administrator-defined period of inactivity.
- Trusted path/channels.
    - Supports secure communication channels for trusted interactions with the audit server.
    - Provides a secure channel for remote administrators using SSH.
    - Channels used between the TOE and other KMEs are post-quantum protected.

1.3.3 Non-TOE Hardware/Software/Firmware

The TOE is available as a virtual machine image for the following platforms:

- KVM as a qcow2 disk image.
- VMware as an OVA.
- Amazon Web Services as an AMI.
- Microsoft Azure as a VHD.

The minimum system requirements for all of these platforms are the same:

- CPU: 1 vCPU, x86_64
- Memory: 512MiB
- Disk: 8GiB

The following hardware, software, and firmware elements are part of the operational environment that the TOE works with.

Component	Required	Description
Local Console	Yes	The initial administration channel is the local console. The user requires a PC connected though serial port to do the first configuration.
Administration Remote PC	Optional	The TOE can be administrated through the Local Console, but commonly the TOE administration is done remotely through SSH.
Secure Application Entities (SAEs)	Optional	SAEs are client devices or applications authorized to interact with the TOE to obtain cryptographic material.

Table 33: Components of the environment

1.4 TOE Description

1.4.1 Physical Scope

The TOE is a **software-only security product** delivered as a **virtual appliance and supporting software packages**, intended to be deployed in a customer-controlled IT environment. Specifically, the TOE is provided as a **preconfigured virtual machine image (OVA format)**.

The TOE does not rely on dedicated proprietary hardware and does not include any physical components. It operates on standard virtualization infrastructures and commodity hardware provided by the end user.

The delivery of the TOE is performed **electronically**, using standard commercial software distribution practices. No physical shipment or removable media are used at any stage of delivery. The TOE is made available to authorized recipients through the developer-controlled cloud distribution service hosted on **Box.com**. Distribution is performed using **recipient-specific, time-limited download links**, ensuring that access to the TOE artifacts is restricted to authorized users and limited in duration.

The following TOE items are made available to the end user via Box.com as part of the official delivery package:

TOE item	Format
phiotx_4.5.6_vm-vga_deb_x86_64.ova	Binary (OVA virtual machine image)
phiotx_4.5.6_vm-vga_deb_x86_64.ova.sha256	Text file
phiotx_4.5.6_vm-vga_deb_x86_64.ova.sha512	Text file
phiotx_4.5.6_open-source_deb_x86_64.tar.gz	Compressed archive
phiotx_4.5.6_open-source_deb_x86_64.tar.gz.sha256	Text file
phiotx_4.5.6_open-source_deb_x86_64.tar.gz.sha512	Text file
PhioTX Install and Admin Guide.pdf	PDF document

Table 44. TOE delivery items

1.4.2 Logical Scope

This section outlines the logical boundaries of the security functionality of the TOE.

1.4.2.1 Security Audit

The TOE is required to generate audit records for a range of events, including administrative actions and various other system activities. In terms of audit data storage, the TOE must be capable of transmitting this data to an external entity using a trusted channel, as well as storing it locally on the TOE itself. The local storage is designed to overwrite the oldest records when it becomes full, ensuring continuous logging capability.

1.4.2.2 Cryptographic support

The TOE's cryptography involves generating, establishing, and destroying cryptographic keys, performing encryption, decryption, signature, and hashing. This cryptography is used by other security functionalities like:

- Generation of symmetric keys to be delivered to SAEs when requested by a SAE acting as master.
- Usage of post-quantum cryptography algorithms to warranty the confidentiality of key symmetric keys (those defined in previous bullet) between TOE and other KMEs.
- Generate cryptographic material for securing the communication channels between TOE and SAEs, KMEs and audit servers with TLS.
- checking the integrity of firmware updates.

1.4.2.3 Identification and authentication

The TOE provides password-based authentication for administrators. This authentication is produced before any CLI interaction. The configuration of this authentication allows managing authentication failures and setting robust password policies.

1.4.2.4 Security Management

The security management capabilities allow security administrators local or remotely to manually install firmware updates, managing cryptographic keys, managing SAEs and KMEs configuration, session inactivity configurations and date/time configuration.

1.4.2.5 Protection of the TSF

The TOE prevents the reading of all pre-shared, symmetric, and private keys, storing and protecting administrative passwords in non-plaintext forms, conducting self-tests on start-up for software integrity, enabling administrators to manage and authenticate firmware/software updates using digital signatures, and providing reliable time stamps.

1.4.2.6 TOE access

Before establishing an administrative user session, the TOE displays a Security Administrator-specified advisory banner. Administrators have the capability to initiate termination of their own interactive sessions. If interactive sessions become inactive for a Security Administrator-defined period, they are automatically terminated.

1.4.2.7 Trusted path/channels

The TOE supports secure communication channels using TLS protocol for trusted interaction with the SAEs, KMEs and external audit server, ensuring endpoint identification and data protection. It allows the TOE to stablish communication with other entities acting as:

- Client in trusted communication with an audit server.
- Server in trusted communication with SAEs.
- Either a client or server when interacting with other KMEs, depending on which party initiates the communication.

For remote administrators it provides a secure channel using SSH protocol.

2 CONFORMANCE CLAIMS

2.1 Common Criteria conformance claim

The ST and TOE claim conformance to the Common Criteria Version CC:2022 revision 1:

- Part 2 extended [CC2].
- Part 3 conformant [CC3].

2.2 Protection Profile conformance

This Security Target does not claim conformance to any Protection Profile.

2.3 Package claim

This Security Target does not claim conformance to any security function requirements or security assurance requirements packages, neither as package-conformant or package-augmented.

2.4 Conformance Rationale

The assurance level for this Security Target is EAL2. The security problem definition, security objectives, and security requirements in this ST are based on the Protection Profile for Network Devices [PPND] but it cannot show exact conformance with it due to the next reasons:

- The assurance level of this ST is higher than the defined in [PPND].
- The exchange of symmetric keys using post-quantum protected algorithms included in this ST is out of the scope of [PPND].

The differences with respect [PPND] are:

- The cryptographic dependencies of some SFRs used in this ST to FCS_CKM.4 in Common Criteria version 3.1r5 are mapped to FCS_CKM.3 in [CC2], so FCS_CKM.4 has been removed and FCS_CKM.3 has been added.
- Some SFRs in [PPND] held dependency to FCS_CKM.4 (as defined by CC v3.1r5), but since these map to FCS_CKM.3 in [CC2], FCS_CKM.4 was removed and FCS_CKM.3 added.
- Some FCS_CKM.1 in [PPND] held dependency to FCS_CKM.4 (as defined by CC v3.1r5), but this dependency maps to FCS_CKM.6 in [CC2], FCS_CKM.6 has been added.
- There has been added cryptographic and channel security functionality (SFRs) to cover the quantum cryptography protection for the communication between the TOE and other KMEs (FTP_ITC.1/QKD, FCS_CKM.2/QKD).
- The required RSA key sizes specified in FCS_CKM.1.1 have been increased from 2048 to 3072.

2.5 Evaluation Methods and Activities

This Security Target does not identify any specific Evaluation Methods and Activities.

3 SECURITY PROBLEM DEFINITION

3.1 Threats

3.1.1 T.UNAUTHORIZED_ADMINISTRATOR_ACCESS

Threat agents may attempt to gain Administrator access to the Network Device by nefarious means such as masquerading as an Administrator to the device, masquerading as the device to an Administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between Network Devices. Successfully gaining Administrator access allows malicious actions that compromise the security functionality of the device and the network on which it resides.

3.1.2 T.WEAK_CRYPTOGRAPHY

Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.

3.1.3 T.UNTRUSTED_COMMUNICATION_CHANNELS

Threat agents may attempt to target Network Devices that do not use standardized secure tunnelling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the Network Device itself.

3.1.4 T.WEAK_AUTHENTICATION_ENDPOINTS

Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints, e.g. a shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the Administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the Network Device itself could be compromised.

3.1.5 T.UPDATE_COMPROMISE

Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.

3.1.6 T.UNDETECTED_ACTIVITY

Threat agents may attempt to access, change, and/or modify the security functionality of the Network Device without Administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the Administrator would have no knowledge that the device has been compromised.

3.1.7 T.SECURITY_FUNCTIONALITY_COMPROMISE

Threat agents may compromise credentials and device data enabling continued access to the Network Device and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or device credentials for use by the attacker.

3.1.8 T.PASSWORD_CRACKING

Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the device. Having privileged access to the device provides the attacker unfettered access to the network traffic and may allow them to take advantage of any trust relationships with other Network Devices.

3.1.9 T.SECURITY_FUNCTIONALITY_FAILURE

An external, unauthorized entity could make use of failed or compromised security functionality and might therefore subsequently use or abuse security functions without prior authentication to access, change or modify device data, critical network traffic or security functionality of the device.

3.1.10 T.KME_LINK_DISCLOSURE

Threat agents may attempt to eavesdrop the communication links between the TOE and other Key Management Entities (KMEs) in a Quantum Key Distribution (QKD) network. If this communication is not adequately protected against post-quantum cryptographic attacks, attackers could obtain cryptographic key material exchanged. This could lead to loss of confidentiality of the key delivery process, compromise of downstream applications relying on those keys, and ultimately undermine the trust in the QKD network.

3.1.11 T.CRYPTO_KEY_COMPROMISE

An attacker may gain unauthorized access to the cryptographic keys distributed to SAEs or inter KME communication keys. The compromise of these keys would allow an attacker to disclose or modify protected SAE data, potentially leading to a breach of the secure key distribution service.

3.2 Organizational Security Policies (OSP)

An organizational security policy is a set of rules, practices, and procedures imposed by an organization to address its security needs.

3.2.1 P.ACCESS_BANNER

The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.

SFR Rationale:

- An advisory notice and consent warning message is required to be displayed by FTA_TAB.1

3.3 Assumptions

This section describes the assumptions made in identification of the threats and security requirements for Network Devices. The Network Device is not expected to provide assurance in any of these areas, and as a result, requirements are not included to mitigate the threats associated.

3.3.1 A.PHYSICAL_PROTECTION

The Network Device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, this ST does not include any requirements on physical tamper protection or other physical attack mitigations. This ST does not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device.

Covered by : OE.PHYSICAL

3.3.2 A.LIMITED_FUNCTIONALITY

The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the device should not provide a computing platform for general purpose applications (unrelated to networking functionality).

Covered by: OE.NO_GENERAL_PURPOSE

3.3.3 A.NO_THRU_TRAFFIC_PROTECTION

A standard/generic Network Device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the Network Device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the Network Device, destined for another network entity, is not covered by this ST.

Covered by: OE.NO_THRU_TRAFFIC_PROTECTION

3.3.4 A.TRUSTED_ADMINISTRATOR

The Security Administrator for the Network Device are assumed to be trusted and to act in the best interest of security for the organization. This includes appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The Network Device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device.

For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are expected to fully validate (e.g. offline verification) any CA certificate (root CA certificate or intermediate CA certificate) loaded into the TOE's trust store (aka 'root store', ' trusted CA Key Store', or similar) as a trust anchor prior to use (e.g. offline verification).

The Security Administrator shall not create or delete users in the TOE's Operating System, as specified the installation and user guides.

Covered by: OE.TRUSTED_ADMIN

3.3.5 A.REGULAR_UPDATES

The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

Covered by: OE.UPDATES

3.3.6 A.ADMIN_CREDENTIALS_SECURE

The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.

Covered by: OE.ADMIN_CREDENTIALS_SECURE

3.3.7 A.RESIDUAL_INFORMATION

The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

Covered by: OE.RESIDUAL_INFORMATION

3.3.8 A.OPERATIVE_SYSTEM

It is assumed that the underlying Operating System is outside the scope of the TOE and that the Security

Administrator does not perform direct administrative, configuration, or user management actions on the Operating System (including user management, modifying OS configurations, or managing system services), strictly adhering to the operational procedures specified in the TOE installation and user guidance documentation.

Covered by: OE.TRUSTED_ADMIN

4 SECURITY OBJECTIVES

4.1 Security Objectives for the TOE

The following subsections describe objectives for the TOE and the threats and OSPs countered by each security objective.

4.1.1 O.TOE_ADMINISTRATION

The TOE ensures that **only authenticated administrators acting under a high-privilege role** can access management functions, that such sessions are always established via protected channels, and displays a legal-use and warning banner before login.

4.1.2 O.STRONG_AUTHENTICATION

The TOE enforces strong password policies, lockout after failed attempts, and certificate-based authentication for remote services, preventing brute-force or impersonation attacks.

4.1.3 O.PROTECTED_COMMUNICATIONS

The TOE ensures that all sensitive exchanges—including Key Management Entity (KME) links—use approved, robust protocols (TLS, SSH v2, PQ algorithms) for confidentiality and integrity.

4.1.4 O.PROTECTED_KME_COMM

The TOE ensures that the communication involved in cryptographic material exchange between itself and other KMEs (i.e. another TOE) are protected against Post-Quantum Cryptoanalysis.

4.1.5 O.STRONG_CRYPTOGRAPHY

The TOE employs only approved algorithms, key lengths and modes (NIST/FIPS or post-quantum), uses high-quality entropy, prevents plaintext key extraction, and clears residual data.

4.1.6 O.UPDATES

The TOE verifies integrity and authenticity of all updates via hash check by administrator before install and rejects invalid updates to maintain a secure state.

4.1.7 O.AUDIT

The TOE generates tamper-resistant records of all security events, stores them so they cannot be modified, provides secure export, and allows administrators to review logs for security events.

4.1.8 O.TSF_SELF_TEST

The TOE at boot and on request, verifies integrity of critical components and crypto modules; on failure, enters a safe state.

4.2 Security objectives for the Operational Environment

The following subsections describe objectives for the Operational Environment and the threats, OSPs and assumptions countered by each security objective.

4.2.1 OE.PHYSICAL

Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

4.2.2 OE.NO_GENERAL_PURPOSE

There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.

4.2.3 OE.NO_THRU_TRAFFIC_PROTECTION

The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.

4.2.4 OE.TRUSTED_ADMIN

Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner.

For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE's trust store in case such certificate can no longer be trusted.

4.2.5 OE.UPDATES

The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

4.2.6 OE.ADMIN_CREDENTIALS_SECURE

The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.

4.2.7 OE.RESIDUAL_INFORMATION

The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

4.3 Security objectives rationale

This section provides the rationale for the defined security objectives for both the Target of Evaluation (TOE) and its operational environment. Each security objective is justified by demonstrating how it addresses one or more identified threats, enforces one or more Organizational Security Policies (OSPs) and the assumptions it upholds.

This structured justification ensures that all identified threats, policies, and assumptions are adequately and consistently addressed by the security objectives.

4.3.1 Security objectives for the TOE

O.TOE_ADMINISTRATION prevents T.UNAUTHORIZED_ADMINISTRATOR_ACCESS by ensuring that only authenticated administrators with high-privilege roles may perform management tasks over secure channels, thereby blocking masquerading or replay attacks. It also counters T.WEAK_AUTHENTICATION_ENDPOINTS by

enforcing strong endpoint authentication for both local and remote sessions. Finally, by displaying the legal-use banner before login, it fulfills P.ACCESS_BANNER and captures user consent.

O.STRONG_AUTHENTICATION addresses T.PASSWORD_CRACKING by enforcing complex passwords, obscured entry, and lockout on failure to defeat brute-force attacks. It mitigates T.WEAK_AUTHENTICATION_ENDPOINTS through certificate-based mutual authentication of administrator and device endpoints, preventing impersonation or replay. By protecting credentials against theft and modification, it also reduces the risk of T.SECURITY_FUNCTIONALITY_COMPROMISE.

O.PROTECTED_COMMUNICATIONS counters T.UNTRUSTED_COMMUNICATION_CHANNELS by mandating that all sensitive traffic—including management commands and audit exports —use approved secure protocols (e.g., TLS, SSH, post-quantum algorithms) to guarantee confidentiality, integrity and peer authentication. It defeats T.WEAK_CRYPTOGRAPHY by requiring strong cryptographic ciphers and key lengths.

O.PROTECTED_KME_COMM counters T.KME_LINK_DISCLOSURE by implementing post-quantum cryptography key exchange algorithms that avoids the disclosure of the transmitted information using Post-Quantum Cryptoanalysis techniques.

O.STRONG_CRYPTOGRAPHY addresses T.WEAK_CRYPTOGRAPHY by restricting operations to approved algorithms, key sizes, and modes, and by ensuring high-quality randomness. It prevents T.CRYPTO_KEY_COMPROMISE through secure generation, storage, and destruction of key material. In addition, by applying post-quantum-resistant ciphers and robust key-encapsulation mechanisms it mitigates T.KME_LINK_DISCLOSURE, ensuring that links between the TOE and Key Management Entities remain confidential against Post-Quantum Cryptanalysis.

O.UPDATES mitigates T.UPDATE_COMPROMISE by verifying the integrity and authenticity of every firmware and software update before installation, rejecting any update that fails validation. This prevents attackers from introducing malicious or tampered code into the TOE.

O.AUDIT counters T.UNDETECTED_ACTIVITY by recording all security-relevant events in protected, tamper-resistant logs and by detecting and reporting any failure to store or transmit logs. Administrators can review these logs to detect unauthorized changes or misconfigurations, ensuring that no malicious activity goes unnoticed.

O.TSF_SELF_TEST addresses T.SECURITY_FUNCTIONALITY_FAILURE by automatically running integrity checks on critical components and cryptographic modules at boot, and it handles T.SECURITY_FUNCTIONALITY_COMPROMISE by re-testing on administrator request. If any test fails, the TOE enters a secure failure state, preventing operation in a compromised or degraded security condition.

	T.UNAUTHORIZED_ADMINISTRATOR_ACCESS	T.WEAK_AUTHENTICATION_ENDPOINTS	T.PASSWORD_CRACKING	T.SECURITY_FUNCTIONALITY_COMPROMISE	T.UNTRUSTED_COMMUNICATION_CHANNELS	T.WEAK_CRYPTOGRAPHY	T.KME_LINK_DISCLOSURE	T.CRYPTO_KEY_COMPROMISE	T.UPDATE_COMPROMISE	T.UNDETECTED_ACTIVITY	T.SECURITY_FUNCTIONALITY_FAILURE	P.ACCESS_BANNER
O.TOE_ADMINISTRATION	X	X										X
O.STRONG_AUTHENTICATION		X	X	X								
O.PROTECTED_COMMUNICATIONS					X	X						
O.PROTECTED_KME_COMM							X					
O.STRONG_CRYPTOGRAPHY						X	X	X				
O.UPDATES									X			
O.AUDIT										X		
O.TSF_SELF_TEST				X							X	

Table 55. Security objectives for the TOE traceability matrix

4.3.2 Security objectives for the Operational Environment

OE.PHYSICAL counters A.PHYSICAL_PROTECTION by requiring that the deployment environment supplies adequate safeguards against physical tampering and interference, thereby relieving the TOE from implementing its own physical tamper-resistance features.

OE.NO_GENERAL_PURPOSE counters A.LIMITED_FUNCTIONALITY by forbidding any non-networking services or applications on the TOE, ensuring that only the intended networking functions are supported and reducing exposure to threats associated with arbitrary code execution.

OE.NO_THRU_TRAFFIC_PROTECTION counters A.NO_THRU_TRAFFIC_PROTECTION by explicitly excluding transit traffic from the TOE's security scope and delegating its protection to other network elements, allowing the TOE to focus solely on data it originates or terminates.

OE.TRUSTED_ADMIN counters A.TRUSTED_ADMINISTRATOR and A.OPERATIVE_SYSTEM by assuming that administrators will act in good faith, enforce credential strength, and validate certificates correctly, so the TOE need not guard against malicious or negligent administrator behavior.

OE.UPDATES counters A.REGULAR_UPDATES by presuming that administrators will install published patches and firmware releases promptly, thereby allowing the TOE to depend on the environment for timely update management instead of enforcing its own update schedule.

OE.ADMIN_CREDENTIALS_SECURE counters A.ADMIN_CREDENTIALS_SECURE by expecting that private keys and passwords are safeguarded externally, enabling the TOE to concentrate on protecting credentials within its own boundary and during use.

OE.RESIDUAL_INFORMATION counters A.RESIDUAL_INFORMATION by relying on administrator-driven sanitization procedures for product end-of-life, thus obviating the need for the TOE to implement its own data-wiping mechanisms for discarded virtual devices.

	A.PHYSICAL_PROTECTION	A.LIMITED_FUNCTIONALITY	A.NO_THRU_TRAFFIC_PROTECTION	A.TRUSTED_ADMINISTRATOR	A.REGULAR_UPDATES	A.ADMIN_CREDENTIALS_SECURE	A.RESIDUAL_INFORMATION	A.OPERATIVE_SYSTEM
OE.PHYSICAL	X							
OE.NO_GENERAL_PURPOSE		X						
OE.NO_THRU_TRAFFIC_PROTECTION			X					
OE.TRUSTED_ADMIN				X				X
OE.UPDATES					X			
OE.ADMIN_CREDENTIALS_SECURE						X		
OE.RESIDUAL_INFORMATION							X	

Table 66. Security objectives for the Operational Environment traceability matrix

5 EXTENDED COMPONENTS DEFINITION

This section contains the definitions for the extended requirements that are used in this ST. Formatting conventions for selections and assignments in the extended requirements are those defined in [CC2].

5.1 Security Audit (FAU)

5.1.1 Security Audit Data Generation (FAU_STG_EXT)

Family Behaviour

This component defines the requirements for the TSF to be able to securely transmit audit data between the TOE and an external IT entity.

Component levelling



FAU_STG_EXT.1 Protected audit event storage requires the TSF to use a trusted channel implementing a secure protocol.

Management: FAU_STG_EXT.1

The following actions could be considered for the management functions in FMT:

a) The TSF shall have the ability to configure the cryptographic functionality.

Audit: FAU_STG_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) No audit necessary.

5.1.1.1 FAU_ STG_EXT.1 Protected Audit Event Storage

Component relationships

Hierarchical to:	No other components.
Dependencies:	FAU_GEN.1 Audit data generation. FTP_ITC.1 Inter-TSF Trusted Channel.

FAU_STG_EXT.1.1

The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1.

FAU_STG_EXT.1.2

The TSF shall be able to store generated audit data on the TOE itself. In addition [selection:

- *The TOE shall consist of a single standalone component that stores audit data locally,*

- *The TOE shall be a distributed TOE that stores audit data on the following TOE components: [assignment: identification of TOE components],*

- *The TOE shall be a distributed TOE with storage of audit data provided externally for the following TOE components: [assignment: list of TOE components that do not store audit data locally and the other TOE components to which they transmit their generated audit data]*.

FAU_STG_EXT.1.3

The TSF shall [selection: *drop new audit data, overwrite previous audit records according to the following rule: [assignment: rule for overwriting previous audit records], [assignment: other action]*] when the local storage space for audit data is full.

5.2 Cryptographic support (FCS)

5.2.1 SSH Server Protocol (FCS_SSHS_EXT)

Family Behaviour

The component in this family addresses the ability for a server to offer SSH to protect data between a client and the server using the SSH protocol.

Component levelling



FCS_SSHS_EXT.1 SSH Server requires that the server side of SSH be implemented as specified.

Management: FCS_SSHS_EXT.1

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_SSHS_EXT.1

The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:

a) Failure of SSH session establishment
b) SSH session establishment
c) SSH session termination

5.2.1.1 FCS_SSHS_EXT.1 SSH Server Protocol

Hierarchical to:	No other components.
Dependencies:	FCS_CKM.1Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification) FCS_COP.1/Hash Cryptographic operation (Hash Algorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol in accordance with: RFCs 4251, 4252, 4253, 4254, [selection: *4256, 4344, 5647, 5656, 6187, 6668, 8268, 8308* section 3.1, 8332].

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, [selection: *password-based, no other method*].

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than *[assignment: number of bytes]* bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: *[assignment: encryption algorithms]*.

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [selection: *ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, x509v3-ssh-rsa, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521, x509v3-rsa2048-sha256*] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses *[assignment: list of MAC algorithms]* as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7 The TSF shall ensure that *[assignment: list of key exchange methods]* are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8 The TSF shall ensure that within SSH connections, the same session keys are used for a threshold of no longer than one hour, and each encryption key is used to protect no more than one gigabyte of data. After any of the thresholds are reached, a rekey needs to be performed.

5.2.2 TLS Client Protocol (FCS_TLSC_EXT)

Family Behaviour

The component in this family addresses the ability for a client to use TLS to protect data between the client and a server using the TLS protocol.

Component levelling



FCS_TLSC_EXT.1 TLS Client requires that the client side of TLS be implemented as specified.

FCS_TLSC_EXT.2 TLS Client requires that the client side of the TLS implementation include mutual authentication.

Management: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_TLSC_EXT.1, FCS_TLSC_EXT.2

The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:

a) Failure of TLS session establishment
b) TLS session establishment
c) TLS session termination

5.2.2.1 FCS_TLSC_EXT.1 TLS Client Protocol without Mutual Authentication

Hierarchical to:	No other components.
Dependencies:	FCS_CKM. 1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification) FCS_COP.1/Hash Cryptographic operation (Hash Algorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation FIA_X509_EXT.1 X.509 Certificate Validation FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSC_EXT.1.1

The TSF shall implement [selection: TLS 1.3 (RFC 8446), *TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

- *[*assignment*: list of optional ciphersuites and reference to RFC in which each is defined]* and no other ciphersuites.

FCS_TLSC_EXT.1.2

The TSF shall verify that the presented identifier matches [selection: *the reference identifier per RFC 6125 section 6, IPv4 address in CN or SAN, IPv6 address in the CN or SAN, IPv4 address in SAN, IPv6 address in the SAN, the identifier per RFC 5280 Appendix A using [selection: id-at-commonName, id-at-countryName, id-at-dnQualifier, idat-generationQualifier, id-at-givenName, id-at-initials, id-at-localityName, id-at-name, id-atorganizationalUnitName, id-at-organizationName, id-at-pseudonym, id-at-serialNumber, idat-stateOrProvinceName, id-at-surname, id-at-title] and no other attribute types*].

FCS_TLSC_EXT.1.3

When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [selection:

- *Not implement any administrator override mechanism*
- *require administrator authorization to establish the connection if the TSF fails to [selection: match the reference identifier, validate certificate path, validate expiration date, determine the revocation status] of the presented server certificate*

].

FCS_TLSC_EXT.1.4 The TSF shall [selection: *not present the Supported Elliptic Curves/Supported Groups Extension, present the Supported Elliptic Curves/Supported Groups Extension with the following curves/groups:* [selection*: secp256r1, secp384r1, secp521r1, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192] and no other curves/groups*] in the Client Hello.

5.2.2.2 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication

Hierarchical to:	No other components.
Dependencies:	FCS_CKM. 1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification) FCS_COP.1/Hash Cryptographic operation (Hash Algorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation FCS_TLSC_EXT.1 TLS Client Protocol without mutual authentication FIA_X509_EXT.1 X.509 Certificate Validation FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSC_EXT.2.1 The TSF shall support TLS communication with mutual authentication using X.509v3 certificates.

5.2.3 TLS Server Protocol (FCS_TLSS_EXT)

Family Behaviour

The component in this family addresses the ability for a server to use TLS to protect data between a client and the server using the TLS protocol.

Component levelling



FCS_TLSS_EXT.1 TLS Server requires that the server side of TLS be implemented as specified.

FCS_TLSS_EXT.2: TLS Server requires the mutual authentication be included in the TLS implementation.

Management: FCS_TLSS_EXT.1, FCS_TLSS_EXT.2

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_TLSS_EXT.1, FCS_TLSS_EXT.2

The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:

a) Failure of TLS session establishment

b) TLS session establishment
c) TLS session termination

5.2.3.1 FCS_TLSS_EXT.1 TLS Server Protocol without Mutual Authentication

Hierarchical to:	No other components.
Dependencies:	FCS_CKM.1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification) FCS_COP.1/Hash Cryptographic operation (Hash Algorithm) FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm) FCS_RBG_EXT.1 Random Bit Generation FIA_X509_EXT.1 X.509 Certificate Validation FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSS_EXT.1.1

The TSF shall implement [selection: *TLS 1.3 (RFC 8446), TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)*] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

- *[assignment: list of optional ciphersuites and reference to RFC in which each is defined]* and no other ciphersuites.

FCS_TLSS_EXT.1.2

The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [selection: *TLS 1.1, TLS 1.2, none*].

FCS_TLSS_EXT.1.3

The TSF shall perform key establishment for TLS using [selection: *RSA with key size* [selection*: 2048 bits, 3072 bits, 4096 bits], Diffie-Hellman parameters with size* [selection*: 2048 bits, 3072 bits, 4096 bits, 6144 bits, 8192 bits], Diffie-Hellman groups* [selection: *ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, no other groups], ECDHE curves* [selection: *secp256r1, secp384r1, secp521r1] and no other curves*]].

FCS_TLSS_EXT.1.4

The TSF shall support [selection: *no session resumption or session tickets, session resumption based on session IDs according to* RFC 4346 *(TLS1.1) or* RFC 5246 *(TLS1.2), session resumption based on session tickets according to* RFC 5077 (TLS 1.2), session resumption according to RFC 8446 (TLS 1.3)].

5.2.3.2 FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication

Hierarchical to:	No other components
Dependencies:	FCS_CKM.1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_COP.1/DataEncryption Cryptographic operation (AES Data encryption/decryption) FCS_COP.1/SigGen Cryptographic operation (Signature Generation and Verification)

FCS_COP.1/Hash Cryptographic operation (Hash Algorithm)
FCS_COP.1/KeyedHash Cryptographic operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1 Random Bit Generation
FCS_TLSS_EXT.1 TLS Server Protocol without mutual authentication
FIA_X509_EXT.1 X.509 Certificate Validation
FIA_X509_EXT.2 X.509 Certificate Authentication

FCS_TLSS_EXT.2.1 The TSF shall support TLS communication with mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the client certificate is invalid. The TSF shall also [selection:

- *Not implement any administrator override mechanism*
- *require administrator authorization to establish the connection if the TSF fails to* [selection: *match the reference identifier, validate certificate path, validate expiration date, determine the revocation status] of the presented client certificate*

FCS_TLSS_EXT.2.3 The TSF shall not establish a trusted channel if the identifier contained in a certificate does not match an expected identifier for the client. If the identifier is a Fully Qualified Domain Name (FQDN), then the TSF shall match the identifiers according to RFC 6125, otherwise the TSF shall parse the identifier from the certificate and match the identifier against the expected identifier of the client as described in the TSS.

5.2.4 Random Bit Generation (FCS_RBG_EXT)

Family Behaviour

Components in this family address the requirements for random bit/number generation. This is a new family defined for the FCS class.

Component levelling



FCS_RBG_EXT.1 Random Bit Generation requires random bit generation to be performed in accordance with selected standards and seeded by an entropy source.

Management: FCS_RBG_EXT.1

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FCS_RBG_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: failure of the randomization process.

5.2.4.1 FCS_RBG_EXT.1: Random Bit Generation

Hierarchical to:	No other components.
Dependencies:	No other components.

FCS_RBG_EXT.1.1

The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [selection: *Hash_DRBG (any), HMAC_DRBG (any), CTR_DRBG (AES)*].

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [selection: *[assignment: number of software-based sources] software-based noise source, [assignment: number of platform-based sources] platform-based noise source*] with a minimum of [selection: *128 bits, 192 bits, 256 bits*] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 C.2, "Security strengths, Entropy and Seed length requirements for the AES-128, 192 and 256 Block Cipher", of the keys and hashes that it will generate.

5.3 Identification and Authentication (FIA)

5.3.1 Password Management (FIA_PMG_EXT)

Family Behaviour

The TOE defines the attributes of passwords used by administrative users to ensure that strong passwords and passphrases can be chosen and maintained.

Component levelling



FIA_PMG_EXT.1 Password management requires the TSF to support passwords with varying composition requirements, minimum lengths, maximum lifetime, and similarity constraints.

Management: FIA_PMG_EXT.1

No management functions.

Audit: FIA_PMG_EXT.1

No specific audit requirements.

5.3.1.1 FIA_PMG_EXT.1 Password Management

Hierarchical to:	No other components.
Dependencies:	No other components.

FIA_PMG_EXT.1.1

The TSF shall provide the following password management capabilities for administrative passwords:

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [selection: *"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [assignment: other characters]*];

b) Minimum password length shall be configurable to between [*assignment: minimum number of characters supported by the TOE*] and [*assignment: number of characters greater than or equal to 15*] characters.

5.3.2 Password Management (FIA_UIA_EXT)

Family Behaviour

The TSF allows certain specified actions before the non-TOE entity goes through the identification and authentication process.

Component levelling

FIA_UIA_EXT User Identification and Authentication	1

FIA_UIA_EXT.1 User Identification and Authentication requires Administrators (including remote Administrators) to be identified and authenticated by the TOE, providing assurance for that end of the communication path. It also ensures that every user is identified and authenticated before the TOE performs any mediated functions.

Management: FIA_UIA_EXT.1

The following actions could be considered for the management functions in FMT:

a) Ability to configure the list of TOE services available before an entity is identified and authenticated

Audit: FIA_UIA_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) All use of the identification and authentication mechanism
b) Provided user identity, origin of the attempt (e.g. IP address)

5.3.2.1 UIA_EXT.1 User Identification and Authentication

Hierarchical to:	No other components.
Dependencies:	FTA_TAB.1 Default TOE Access Banners

FIA_UIA_EXT.1.1

The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- [selection: *no other actions, automated generation of cryptographic keys, [assignment: list of services, actions performed by the TSF in response to non-TOE requests]*].

FIA_UIA_EXT.1.2

The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

5.3.3 Password Management (FIA_UAU_EXT)

Family Behaviour

Provides for a locally based administrative user authentication mechanism.

Component levelling



Management: FIA_UAU_EXT.2

The following actions could be considered for the management functions in FMT:

b) None

Audit: FIA_UAU_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: All use of the authentication mechanism

5.3.3.1 FIA_UAU_EXT.2 Password-based Authentication Mechanism

Hierarchical to:	No other components.
Dependencies:	No other components.

FIA_UAU_EXT.2.1

The TSF shall provide a local [selection: *password-based, SSH public key-based, certificate-based, [assignment: other authentication mechanism(s)]*] authentication mechanism to perform local administrative user authentication.

5.3.4 Authentication using X.509 certificates (FIA_X509_EXT)

Family Behaviour

This family defines the behaviour, management, and use of X.509 certificates for functions to be performed by the TSF. Components in this family require validation of certificates according to a specified set of rules, use of certificates for authentication for protocols and integrity verification, and the generation of certificate requests.



FIA_X509_EXT.1 X509 Certificate Validation, requires the TSF to check and validate certificates in accordance with the RFCs and rules specified in the component.

FIA_X509_EXT.2 X509 Certificate Authentication, requires the TSF to use certificates to authenticate peers in protocols that support certificates, as well as for integrity verification and potentially other functions that require certificates.

FIA_X509_EXT.3 X509 Certificate Requests, requires the TSF to be able to generate Certificate Request Messages and validate responses.

Management: FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3

The following actions could be considered for the management functions in FMT:

a) Remove imported X.509v3 certificates
b) Approve import and removal of X.509v3 certificates
c) Initiate certificate requests

Audit: FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Minimal: No specific audit requirements are specified.

5.3.4.1 FIA_X509_EXT.1 X.509 Certificate Validation

Hierarchical to:	No other components
Dependencies:	FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.1.1

The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certification path validation.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The TSF shall validate the revocation status of the certificate using [selection: the Online Certificate Status Protocol (OCSP) as specified in RFC 6960, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5, no revocation method].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.

FIA_X509_EXT.1.2

The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

5.3.4.2 FIA_X509_EXT.2 X.509 Certificate Authentication

Hierarchical to:	No other components
Dependencies:	FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [selection*: DTLS, HTTPS, IPsec, TLS, SSH, [assignment: other protocols], no protocols*], and [selection: *code signing for system software updates [assignment: other uses], no additional uses*].

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [selection: *allow the Administrator to choose whether to accept the certificate in these cases, accept the certificate, not accept the certificate*].

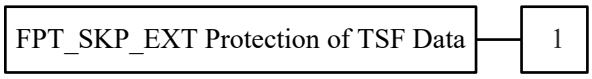
5.4 Protection of the TSF (FPT)

5.4.1 Protection of TSF Data (FPT_SKP_EXT)

Family Behaviour

Components in this family address the requirements for managing and protecting TSF data, such as cryptographic keys. This is a new family modelled after the FPT_PTD Class.

Component levelling



FPT_SKP_EXT.1 Protection of TSF Data (for reading all symmetric keys), requires preventing symmetric keys from being read by any user or subject. It is the only component of this family.

Management: FPT_SKP_EXT.1

The following actions could be considered for the management functions in FMT:

a) There are no management activities foreseen.

Audit: FPT_SKP_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) There are no auditable events foreseen.

5.4.1.1 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all symmetric keys)

Hierarchical to:	No other components.
Dependencies:	No other components.

FPT_SKP_EXT.1.1

The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

5.4.2 Protection of Administrator Passwords (FPT_APW_EXT)

Family Behaviour

Components in this family ensure that the TSF will protect plaintext credential data such as passwords from unauthorized disclosure.

Component levelling



FPT_APW_EXT.1 Protection of Administrator passwords requires that the TSF prevent plaintext credential data from being read by any user or subject.

Management: FPT_APW_EXT.1

The following actions could be considered for the management functions in FMT:

a) No management functions.

Audit: FPT_APW_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) No audit necessary.

5.4.2.1 FPT_APW_EXT.1 Protection of Administrator Passwords

Hierarchical to:	No other components.
Dependencies:	No other components.

FPT_APW_EXT.1.1

The TSF shall store administrative passwords in non-plaintext form.

FPT_APW_EXT.1.2

The TSF shall prevent the reading of plaintext administrative passwords.

5.4.3 TSF Self-Test (FPT_TST_EXT)

Family Behaviour

Components in this family address the requirements for self-testing the TSF for selected correct operation.

Component levelling



FPT_TST_EXT.1 TSF Self-Test requires a suite of self-tests to be run during initial start-up in order to demonstrate correct operation of the TSF.

Management: FPT_TST_EXT.1

The following actions could be considered for the management functions in FMT:

a) No management functions.

Audit: FPT_TST_EXT.1

The following actions should be considered for audit if FAU_GEN Security audit data generation is included in the PP/ST:

a) Indication that TSF self-test was completed
b) Failure of self-test

5.4.3.1 FPT_TST_EXT.1 TSF Testing

Hierarchical to:	No other components.
Dependencies:	No other components.

FPT_TST_EXT.1.1

The TSF shall run a suite of the following self-tests [selection: during initial start-up (on power on), periodically during normal operation, at the request of the authorised user, at the conditions [assignment: conditions under which self-tests should occur]] to demonstrate the correct operation of the TSF: [assignment: list of self-tests run by the TSF].

5.4.4 Trusted Update (FPT_TUD_EXT)

Family Behaviour

Components in this family address the requirements for updating the TOE firmware and/or software.

Component levelling



FPT_TUD_EXT.1 Trusted Update requires management tools be provided to update the TOE firmware and software, including the ability to verify the updates prior to installation.

FPT_TUD_EXT.2 Trusted update based on certificates applies when using certificates as part of trusted update and requires that the update does not install if a certificate is invalid.

Management: FPT_TUD_EXT.1, FPT_TUD_EXT.2

The following actions could be considered for the management functions in FMT:

a) Ability to update the TOE and to verify the updates

b) Ability to update the TOE and to verify the updates using the digital signature capability (FCS_COP.1/SigGen) and [selection: *no other functions, [assignment: other cryptographic functions (or other functions) used to support the update capability]*]
c) Ability to update the TOE, and to verify the updates using [selection: *digital signature, published hash, no other mechanism*] capability prior to installing those updates

Audit: FPT_TUD_EXT.1, FPT_TUD_EXT.2

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Initiation of the update process.
b) Any failure to verify the integrity of the update.

5.4.4.1 FPT_TUD_EXT.1 Trusted Update

Hierarchical to:	No other components.
Dependencies:	FCS_COP.1/SigGen Cryptographic operation (for Cryptographic Signature and Verification), or FCS_COP.1/Hash Cryptographic operation (for cryptographic hashing)

FPT_TUD_EXT.1.1

The TSF shall provide [assignment: Administrators] the ability to query the currently executing version of the TOE firmware/software and [selection: the most recently installed version of the TOE firmware/software; no other TOE firmware/software version].

FPT_TUD_EXT.1.2

The TSF shall provide [assignment: Administrators] the ability to manually initiate updates to TOE firmware/software and [selection: support automatic checking for updates, support automatic updates, no other update mechanism].

FPT_TUD_EXT.1.3

The TSF shall provide means to authenticate firmware/software updates to the TOE using a [selection: X.509 certificate, digital signature, published hash] prior to installing those updates.

5.4.4.2 FPT_TUD_EXT.2 Trusted Update Based on Certificates

Hierarchical to:	No other components.
Dependencies:	FPT_TUD_EXT.1

FPT_TUD_EXT.2.1

The TSF shall check the validity of the code signing certificate before installing each update.

FPT_TUD_EXT.2.2

If revocation information is not available for a certificate in the trust chain that is not a trusted certificate designated as a trust anchor, the TSF shall [selection: not install the update, allow the Administrator to choose whether to accept the certificate in these cases].

FPT_TUD_EXT.2.3

If the certificate is deemed invalid because the certificate has expired, the TSF shall [selection: allow the Administrator to choose whether to install the update in these cases, not accept the certificate].

FPT_TUD_EXT.2.4

If the certificate is deemed invalid for reasons other than expiration or revocation information being unavailable, the TSF shall not install the update.

5.4.5 Time stamps (FPT_STM_EXT)

Family Behaviour

Components in this family extend FPT_STM requirements by describing the source of time used in timestamps.

Component levelling



FPT_STM_EXT.1 Reliable Time Stamps is hierarchic to FPT_STM.1: it requires that the TSF provide reliable time stamps for TSF and identifies the source of the time used in those timestamps.

Management: FPT_STM_EXT.1

The following actions could be considered for the management functions in FMT:

a) Management of the time
b) Administrator setting of the time.

Audit: FTA_SSL_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Discontinuous changes to the time.

5.4.5.1 FPT_STM_EXT.1 Reliable Time Stamps

Hierarchical to:	No other components.
Dependencies:	FPT_TUD_EXT.1

FPT_STM_EXT.1.1

The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2

The TSF shall [selection: *allow the Security Administrator to set the time, synchronise time with an NTP server*].

5.5 TOE Access (FTA)

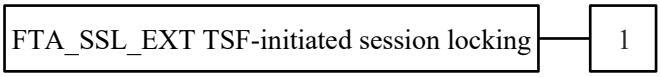
5.5.1 TSF-initiated Session Locking (FTA_SSL_EXT)

Family Behaviour

Components in this family address the requirements for TSF-initiated and user-initiated locking, unlocking, and termination of interactive sessions.

The extended FTA_SSL_EXT family is based on the FTA_SSL family.

Component levelling



FTA_SSL_EXT.1 TSF-initiated session locking requires system initiated locking of an interactive session after a specified period of inactivity. It is the only component of this family.

Management: FTA_SSL_EXT.1

The following actions could be considered for the management functions in FMT:

a) Specification of the time of user inactivity after which lock-out occurs for an individual user.

Audit: FTA_SSL_EXT.1

The following actions should be auditable if FAU_GEN Security audit data generation is included in the PP/ST:

a) Any attempts at unlocking an interactive session.

5.5.1.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

Hierarchical to:	No other components.
Dependencies:	FIA_UAU.1 Timing of authentication

FTA_SSL_EXT.1.1

The TSF shall, for local interactive sessions, [selection:

- *lock the session - disable any activity of the Administrator's data access/display devices other than unlocking the session, and requiring that the Administrator re-authenticate to the TSF prior to unlocking the session;*
- *terminate the session*] after a Security Administrator-specified time period of inactivity.

6 SECURITY REQUIREMENTS

The typographical conventions used in descriptions of the SFRs for CC approved set of operations (iteration, assignment, selection, and refinement are:

- Unaltered SFRs are stated in the form used in [CC2] or their extended component definition (ECD).
- **Refinement:** Allows the addition of details. The text is identified with **bold text** for additions and ~~strikethroughs~~ for deletions.
- **Selection:** Chooses a subset of given options from a list. The selection values are indicated with underlined text.
- **Assignment:** Gives a value to an identified parameter. The assignment values are indicated with *italicized text*.
- **Iteration:** Allows a SFR to be used more than once with varying operations. SFRs are distinguished by adding a string starting with "/" (i.e. 'FCS_COP.1/Hash').

6.1 Security Functional Requirements (SFRs)

6.1.1 Security Audit (FAU)

6.1.1.1 FAU_GEN.1 Audit data generation

FAU_GEN.1.1 The TSF shall be able to generate audit data of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events for the not specified level of audit; and

c) *All administrative actions comprising:*

- *Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).*
- *Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).*
- *Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).*
- *Resetting passwords (name of related user account shall be logged).*
- [no other actions];

d) *Specifically defined auditable events listed in ~~Table 2~~**Table 77Table 7***.

FAU_GEN.1.2 The TSF shall record within the audit data at least the following information:

a) Date and time of the auditable event, type of event, subject identity (if applicable), and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP, PP-Module, functional package or ST, *information specified in column three of ~~Table 2~~**Table 77Table 7***.

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG_EXT.1	None.	None.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.
FCS_CKM.3	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_RBG_EXT.1	None.	None.
FCS_SSH_EXT.1	Failure to establish an SSH session.	Reason for failure.
FCS_TLSC_EXT.1	Failure to establish a TLS Session.	Reason for failure.
FCS_TLSS_EXT.1	Failure to establish a TLS Session.	Reason for failure.
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None	None
FIA_X509_EXT.1	None	None
FIA_X509_EXT.2	None	None
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update.	None.
FMT_MTD.1/CoreData	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.

FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an	For discontinuous changes to time: The old and new values for the time. Origin of the
	automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1 (if "lock the session" is selected)	Any attempts at unlocking of an interactive session.	None.
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.
FTP_ITC.1	• Initiation of the trusted channel. • Termination of the trusted channel. • Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Admin	• Initiation of the trusted path. • Termination of the trusted path. • Failure of the trusted path functions.	None.

Table 77: Security Functional Requirements and Auditable Events

6.1.1.2 FAU_GEN.2 User identity association

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

6.1.1.3 FAU_ STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1.

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself. In addition [

- The TOE shall consist of a single standalone component that stores audit data locally]

FAU_STG_EXT.1.3 The TSF shall [overwrite previous audit records according to the following rule: [*delete older records when audit record is full (111KB)*]] when the local storage space for audit data is full.

6.1.2 Cryptographic Support (FCS)

6.1.2.1 FCS_CKM.1 Cryptographic Key Generation (Refinement)

FCS_CKM.1.1 The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm: [

- *RSA schemes using cryptographic key sizes of 3072-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;*
- *ECC schemes using 'NIST curves' [P-256, P-384] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;*

~~] and specified cryptographic key sizes [assignment: *cryptographic key sizes*] that meet the following: [assignment: *list of standards*]~~.

6.1.2.2 FCS_CKM.2 Cryptographic Key Establishment (Refinement)

FCS_CKM.2.1 The TSF shall ~~distribute~~ **perform** cryptographic ~~keys~~ **key establishment** in accordance with a specified cryptographic key ~~distribution~~ **establishment** method: [

- *Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";*

~~] that meets the following: [assignment: *list of standards*]~~.

6.1.2.3 FCS_CKM.2/QKD Cryptographic Key Establishment

FCS_CKM.2.1/QKD The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [*ML-KEM 1024*] that meets the following: [*FIPS 203 (ML-KEM EncapDecap, KeyGen)*].

6.1.2.4 FCS_CKM.3 Cryptographic key access

FCS_CKM.3.1 The TSF shall perform [*cryptographic key archival and cryptographic key recovery*] in accordance with a specified cryptographic key access method [*TSF internal access only*] that meets the following: [*None*].

6.1.2.5 FCS_CKM.6 Timing and event of cryptographic key destruction

FCS_CKM.6.1 The TSF shall destroy [*SSH/TLS private keys and post-quantum KEM keys*] when [[*SSH and TLS keys are destroyed upon administrator command, while KEM-generated keys are single-use and are destroyed immediately after retrieval*]].

FCS_CKM.6.2 The TSF shall destroy cryptographic keys and keying material specified by FCS_CKM.6.1 in accordance with a specified cryptographic key destruction method [*release of memory and rewrite of all disk sectors*] that meets the following: [*FIPS 140-3*].

6.1.2.6 FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/ Decryption)

FCS_COP.1.1/DataEncryption The TSF shall perform [*encryption/decryption*] in accordance with a specified cryptographic algorithm *AES used in* [*CBC, CTR, GCM*] *mode* and cryptographic key sizes [*128 bits, 192 bits, 256 bits*] that meet the following: [*AES as specified in ISO 18033-3, CBC as specified in ISO 10116, CTR as specified in ISO 10116, GCM as specified in ISO 19772*].

6.1.2.7 FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)

FCS_COP.1.1/SigGen The TSF shall perform [*cryptographic signature services (generation and verification)*] in accordance with a specified cryptographic algorithm [

- *RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [3072],*
- *Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256 bits and 384 bits]*

]

that meet the following: [

- *For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,*

- *For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"* [*P-256, P-384*]; *ISO/IEC 14888-3, Section 6.4*

].

6.1.2.8 FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)

FCS_COP.1.1/Hash The TSF shall perform [*cryptographic hashing services*] in accordance with a specified cryptographic algorithm [*SHA-256, SHA-384, SHA-512*] ~~and cryptographic key sizes [*assignment: cryptographic key sizes*~~] and **message digest sizes [*256, 384, 512*] bits** that meet the following: [*ISO/IEC 10118-3:2004*].

6.1.2.9 FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm)

FCS_COP.1.1/KeyedHash The TSF shall perform [*keyed-hash message*] *authentication* in accordance with a specified cryptographic algorithm [*HMAC-SHA256, HMAC-SHA-384, HMAC-SHA-512*] and cryptographic key sizes *[256, 384, 512]* **and message digest sizes [*256, 384, 512*] bits** that meet the following: [*ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*].

6.1.2.10 FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [*CTR_DRBG (AES)*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [*[1]* software-based noise source, *[3]* platform-based noise source] with a minimum of [*256 bits*] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions", of the keys and hashes that it will generate.

6.1.2.11 FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol in accordance with: RFCs 4251, 4252, 4253, 4254, [*5647, 5656 6668*].

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods as described in RFC 4252: public key-based, [*password-based*].

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than *[35.000]* bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: *[aes128-gcm aes256-gcm]*.

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [*ecdsa-sha2-nistp256, ecdsa-sha2-nistp384*] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses *[hmac-sha2-256, hmac-sha2-512]* as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7 The TSF shall ensure that *[ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521]* are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8 The TSF shall ensure that within SSH connections, the same session keys are used for a threshold of no longer than one hour, and each encryption key is used to protect no more than one gigabyte of data. After any of the thresholds are reached, a rekey needs to be performed.

6.1.2.12 FCS_TLSC_EXT.1 TLS Client Protocol without Mutual Authentication

FCS_TLSC_EXT.1.1 The TSF shall implement [*TLS 1.3 (RFC 8446), TLS 1.2 (RFC 5246)*] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites: *[*

- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5289)*
- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (RFC 5289)*
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384

] and no other ciphersuites.

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches [*the reference identifier per RFC 6125 section 6 and no other attribute types*].

FCS_TLSC_EXT.1.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [

- *Not implement any administrator override mechanism*

].

FCS_TLSC_EXT.1.4 The TSF shall [*present the Supported Elliptic Curves/Supported Groups Extension with the following curves/groups: [secp256r1, secp384r1] and no other curves/groups*] in the Client Hello.

6.1.2.13 FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication

FCS_TLSC_EXT.2.1 The TSF shall support TLS communication with mutual authentication using X.509v3 certificates.

6.1.2.14 FCS_TLSS_EXT.1 TLS Server Protocol without Mutual Authentication

FCS_TLSS_EXT.1.1

The TSF shall implement [*TLS 1.3 (RFC 8446), TLS 1.2 (RFC 5246)*] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites: *[*

- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (RFC 5289)
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (RFC 5289)
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384

] and no other ciphersuites.

FCS_TLSS_EXT.1.2

The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [*TLS 1.1*].

FCS_TLSS_EXT.1.3

The TSF shall perform key establishment for TLS using [*ECDHE curves [secp256r1, secp384r1] and no other curves*]].

FCS_TLSS_EXT.1.4

The TSF shall support [*no session resumption or session tickets*].

6.1.2.15 FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication

FCS_TLSS_EXT.2.1 The TSF shall support TLS communication with mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the client certificate is invalid. The TSF shall also [

- *Not implement any administrator override mechanism*

] of the presented client certificate

FCS_TLSS_EXT.2.3 The TSF shall not establish a trusted channel if the identifier contained in a certificate does not match an expected identifier for the client. If the identifier is a Fully Qualified Domain Name (FQDN), then the TSF shall match the identifiers according to RFC 6125, otherwise the TSF shall parse the identifier from the certificate and match the identifier against the expected identifier of the client as described in the TSS.

6.1.3 Identification and Authentication (FIA)

6.1.3.1 FIA_AFL.1 Authentication Failure Management

FIA_AFL.1.1 The TSF shall detect when [an Administrator configurable positive integer within [*tree attempts*]] unsuccessful authentication attempts occur related to [*Administrators attempting to authenticate remotely using a password*].

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been [met], the TSF shall [*prevent the offending Administrator from successfully establishing a remote session using any authentication method that involves a password until [other administrator user unlocks it via command line] is taken by an Administrator; prevent the offending Administrator from successfully establishing a remote session using any authentication method that involves a password until an Administrator defined time period has elapsed]*.

6.1.3.2 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, *[other printable ASCII characters]*];

b) Minimum password length shall be configurable to between [*12 characters*] and [*256 characters*] characters.

6.1.3.3 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1.1 The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- [no other actions].

FIA_UIA_EXT.1.2 The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

6.1.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local [*password-based, SSH public key-based*] authentication mechanism to perform local administrative user authentication.

6.1.3.5 FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1 The TSF shall provide only *obscured feedback* to the **administrative** user while the authentication is in progress **at the local console**.

6.1.3.6 FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1 The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certification path validation.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The TSF shall validate the revocation status of the certificate using [no revocation method].

The TSF shall validate the extendedKeyUsage field according to the following rules:

- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field

FIA_X509_EXT.1.2 The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

6.1.3.7 FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [*TLS, SSH*], and [*no additional uses*].

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [*not accept the certificate*].

6.1.4 Security Management (FMT)

6.1.4.1 FMT_MOF.1/ManualUpdate Management of Security Functions Behaviour

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to [enable] the functions [*to perform manual updates*] to [*Security Administrators*].

6.1.4.2 FMT_MTD.1/CoreData Management of TSF Data

FMT_MTD.1.1/CoreData The TSF shall restrict the ability to [manage] the [*TSF data*] to [*Security Administrators*].

6.1.4.3 FMT_MTD.1/CryptoKeys Management of TSF Data

FMT_MTD.1.1/CryptoKeys The TSF shall restrict the ability to [*manage*] the [*cryptographic keys*] to [*Security Administrators*].

6.1.4.4 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions: [

- *Ability to administer the TOE locally and remotely;*
- *Ability to configure the access banner;*
- *Ability to configure the session inactivity time before session termination or locking;*
- *Ability to update the TOE, and to verify the updates using hash comparison capability prior to installing those updates;*
- *Ability to configure the authentication failure parameters for FIA_AFL.1;*
- *Ability to start and stop services;*
- *Ability to configure audit behaviour (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full);*
- *Ability to modify the behaviour of the transmission of audit data to an external IT entity;*
- *Ability to configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1;*
- *Ability to manage the cryptographic keys; o Ability to configure the cryptographic functionality;*
- *Ability to configure the interaction between TOE components;*
- *Ability to re-enable an Administrator account;*
- *Ability to set the time which is used for time-stamps; o Ability to configure NTP;*

- *Ability to configure the reference identifier for the peer;*
- *Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;*
- *Ability to import X.509v3 certificates to the TOE's trust store.*

]

6.1.4.5 FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

- *Security Administrator*.

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions [

- *The Security Administrator role shall be able to administer the TOE locally;*
- *The Security Administrator role shall be able to administer the TOE remotely* are satisfied.

]

6.1.5 Protection of the TSF (FPT)

6.1.5.1 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)

FPT_SKP_EXT.1.1 The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

6.1.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

FPT_APW_EXT.1.1 The TSF shall store administrative passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext administrative passwords.

6.1.5.3 FPT_TST_EXT.1 TSF Testing

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [*during initial start-up (on power on), periodically during normal operation, at the request of the authorised user*] to demonstrate the correct operation of the TSF: [*Cryptographic algorithms verified by running test vectors and comparing the calculated results and expected values and Firmware integrity by checking signature*].

6.1.5.4 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide [*Security Administrators*] the ability to query the currently executing version of the TOE firmware/software and [no other TOE firmware/software version].

FPT_TUD_EXT.1.2 The TSF shall provide [*Security Administrators*] the ability to manually initiate updates to TOE firmware/software and [*no other update mechanism*].

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [*published hash*] prior to installing those updates.

6.1.5.5 FPT_STM_EXT.1 Reliable Time Stamps

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [synchronise time with an NTP server].

6.1.6 TOE Access (FTA)

6.1.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [terminate the session] after a Security Administrator-specified time period of inactivity.

6.1.6.2 FTA_SSL.3 TSF-initiated Termination

FTA_SSL.3.1: The TSF shall terminate ~~an~~ **a remote** interactive session after a [*Security Administrator-configurable time interval of session inactivity*].

6.1.6.3 FTA_SSL.4 User-initiated Termination

FTA_SSL.4.1: The TSF shall allow ~~user~~ **Administrator**-initiated ~~user's~~ **Administrator's** own interactive session.

6.1.6.4 FTA_TAB.1 Default TOE Access Banners

FTA_TAB.1.1: Before establishing ~~a~~ **an administrative** user session the [TSF] shall display ~~an~~ [*a Security Administrator-specified advisory notice and consent warning regarding use of the TOE*] message.

6.1.7 Trusted path/channels (FTP)

6.1.7.1 FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1 The TSF shall **be capable of using TLS to** provide a **trusted** communication channel between itself and ~~another trusted IT product~~ **authorized IT entities supporting the following capabilities: audit server, SAEs** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from ~~modification or~~ **disclosure and detection of modification of the channel data**.

FTP_ITC.1.2 The TSF shall permit [the TSF, another trusted IT product] to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for *[send symmetric keys to SAEs on request]*.

6.1.7.2 FTP_ITC.1/QKD Inter-TSF Trusted Channel

FTP_ITC.1.1/QKD The TSF shall **be capable of using ML-KEM to** provide a **trusted** communication channel between itself and ~~another trusted IT product~~ **other authorized KME entity** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from ~~modification or~~ **disclosure and detection of modification of the channel data**.

FTP_ITC.1.2/QKD The TSF shall permit [the TSF, another trusted IT product] to initiate communication via the trusted channel.

FTP_ITC.1.3/QKD The TSF shall initiate communication via the trusted channel for *[exchange symmetric keys with other KMEs]*.

6.1.7.3 FTP_TRP.1/Admin Trusted Path

FTP_TRP.1.1/Admin The TSF shall **be capable of using *SSH* to** provide a communication path between itself and **authorized** [remote] ~~users~~ **Administrators** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [*disclosure and provides detection of modification of the channel data*]**.**

FTP_TRP.1.2/Admin The TSF shall permit [remote ~~users~~ **Administrators**] to initiate communication via the trusted path.

FTP_TRP.1.3/Admin The TSF shall require the use of the trusted path for [*initial ~~user~~ Administrator authentication and all remote administration actions*].

6.1.8 SFR Dependencies Analysis

The dependencies between SFRs implemented by the TOE are addressed as follows.

SFR	Dependencies	Rationale Statement
FAU_GEN.1	FPT_STM.1	FPT_STM_EXT.1 included (which is hierarchic to FPT_STM.1).
FAU_GEN.2	FAU_GEN.1 and FIA_UID.1	FAU_GEN.1 included Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator identification timing
FAU_STG_EXT.1	FAU_GEN.1 and FTP_ITC.1	FAU_GEN.1 included FTP_ITC.1 included
FCS_CKM.1	[FCS_CKM.2 or FCS_CKM.5 or FCS_COP.1] and FCS_CKM.3 and [FCS_RBG.1 or FCS_RNG.1] and [FCS_CKM.6]	FCS_CKM.2 included FCS_CKM.3 included FCS_RBG_EXT.1 included FCS_CKM.6 included
FCS_CKM.2	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	FCS_CKM.1 included FCS_CKM.3 included
FCS_CKM.2/QKD	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	FTP_ITC.1/QKD included FCS_CKM.3 included
FCS_CKM.3	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5]	FCS_CKM.1 included
FCS_CKM.6	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1 included

FCS_COP.1/DataEncryption	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	FCS_CKM.1 included FCS_CKM.3 included
FCS_COP.1/SigGen	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	FCS_CKM.1 included FCS_CKM.3 included
FCS_COP.1/Hash	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	This SFR specifies keyless hashing operations, so initialisation and destruction of keys are not relevant
FCS_COP.1/KeyedHash	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1 or FCS_CKM.5] and FCS_CKM.3	FCS_CKM.1 included FCS_CKM.3 included
FCS_RBG_EXT.1	None	
FCS_SSHS_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1 Random Bit Generation	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption included FCS_COP.1/SigGen included FCS_COP.1/Hash included FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 Random Bit Generation included
FCS_TLSC_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1 FIA_X509_EXT.1 FIA_X509_EXT.2	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption included FCS_COP.1/SigGen included FCS_COP.1/Hash included FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included FIA_X509_EXT.1included FIA_X509_EXT.2 included
FCS_TLSC_EXT.2	FCS_CKM. 1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash	FCS_CKM. 1 included FCS_CKM.2 included FCS_COP.1/DataEncryption included FCS_COP.1/SigGen included

	FCS_COP.1/KeyedHash FCS_RBG_EXT.1 FCS_TLSC_EXT.1 FIA_X509_EXT.1 FIA_X509_EXT.2	FCS_COP.1/Hash included FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included FCS_TLSC_EXT.1 included FIA_X509_EXT.1 included FIA_X509_EXT.2 included
FCS_TLSS_EXT.1	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1 FIA_X509_EXT.1 FIA_X509_EXT.2	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption included FCS_COP.1/SigGen included FCS_COP.1/Hash included FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included FIA_X509_EXT.1 included FIA_X509_EXT.2 includee
FCS_TLSS_EXT.2	FCS_CKM.1 FCS_CKM.2 FCS_COP.1/DataEncryption FCS_COP.1/SigGen FCS_COP.1/Hash FCS_COP.1/KeyedHash FCS_RBG_EXT.1 FCS_TLSS_EXT.1 FIA_X509_EXT.1 FIA_X509_EXT.2	FCS_CKM.1 included FCS_CKM.2 included FCS_COP.1/DataEncryption included FCS_COP.1/SigGen included FCS_COP.1/Hash included FCS_COP.1/KeyedHash included FCS_RBG_EXT.1 included FCS_TLSS_EXT.1 included FIA_X509_EXT.1 included FIA_X509_EXT.2 included
FIA_X509_EXT.1	FIA_X509_EXT.2	FIA_X509_EXT.2 included
FIA_X509_EXT.2	FIA_X509_EXT.1	FIA_X509_EXT.1 included
FIA_AFL.1	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator authentication
FIA_PMG_EXT.1	None	
FIA_UIA_EXT.1	FTA_TAB.1	FTA_TAB.1 included
FIA_UAU_EXT.2	None	
FIA_UAU.7	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator authentication

FMT_MOF.1/ManualUpdate	FMT_SMR.1 and FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_MTD.1/CoreData	FMT_SMR.1 and FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_MTD.1/CryptoKeys	FMT_SMR.1 and FMT_SMF.1	FMT_SMR.2 included FMT_SMF.1 included
FMT_SMF.1	None	
FMT_SMR.2	FIA_UID.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator identification
FPT_SKP_EXT.1	None	
FPT_APW_EXT.1	None	
FPT_TST_EXT.1	None	
FPT_TUD_EXT.1	FCS_COP.1/SigGen or FCS_COP.1/Hash	FCS_COP.1/SigGen and FCS_COP.1/Hash included
FPT_STM_EXT.1	None	
FTA_SSL_EXT.1	FIA_UAU.1	Satisfied by FIA_UIA_EXT.1, which specifies the relevant Administrator authentication
FTA_SSL.3	FMT_SMR.1	FMT_SMR.2 included
FTA_SSL.4	None	
FTA_TAB.1	None	
FTP_ITC.1	None	
FTP_ITC.1/QKD	None	
FTP_TRP.1/Admin	None	

6.2 Security Assurance Requirements (SARs)

The assurance requirements are identified in the table below. This assurance level corresponds with EAL2 “Structurally tested” as defined by [CC5].

Assurance Class	**Assurance Components**
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)

	Security objectives (ASE_OBJ.2)
	Derived security requirements (ASE_REQ.2)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Security architecture description (ADV_ARC.1)
	Basic functional specification (ADV_FSP.2)
	Basic design (ADV_TDS.1)
Guidance Documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life Cycle Support (ALC)	Use of a CM system (ALC_CMC.2)
	Parts of the TOE CM coverage (ALC_CMS.2)
	Delivery procedures (ALC_DEL.1)
Tests (ATE)	Evidence of coverage (ATE_COV.1)
	Functional testing (ATE_FUN.1)
	Independent testing – sample (ATE_IND.2)
Vulnerability Assessment (AVA)	Vulnerability analysis (AVA_VAN.2)

Table 88: Security Assurance Requirements

6.3 Security requirements rationale

6.3.1 Security Functional requirements (SFR) Rationale

In this section it is demonstrated how the SFRs meet all the security objectives for the TOE. Table 99 provides a traceability matrix between the SFRs and the security objectives, indicating that each security objective is mapped to at least one SFR, and there are no SFRs that remain unmapped to a security objective.

After Table 99 there is a rationale explaining how the security objectives for the TOE are covered by the SFRs.

	O.TOE_ADMINISTRATION	O.STRONG_AUTHENTICATION	O.PROTECTED_COMMUNICATIONS	O.PROTECTED_KME_COMM	O.STRONG_CRYPTOGRAPHY	O.UPDATES	O.AUDIT	O.TSF_SELF_TEST
FAU_GEN.1						✓	✓	✓

FAU_GEN.2						✓	✓	
FAU_STG_EXT.1			✓				✓	
FCS_CKM.1			✓		✓			
FCS_CKM.2			✓		✓			
FCS_CKM.2/QKD				✓	✓			
FCS_CKM.3			✓		✓			
FCS_CKM.6			✓	✓	✓			
FCS_COP.1/DataEncryption			✓		✓			
FCS_COP.1/SigGen			✓		✓			
FCS_COP.1/Hash			✓	✓	✓			
FCS_COP.1/KeyedHash			✓	✓	✓			
FCS_RBG_EXT.1			✓	✓	✓			
FCS_TLSC_EXT.1			✓					
FCS_TLSC_EXT.2			✓					
FCS_TLSS_EXT.1		✓	✓					
FCS_TLSS_EXT.2		✓	✓					
FCS_SSHS_EXT.1	✓	✓	✓					
FIA_AFL.1	✓	✓						
FIA_PMG_EXT.1	✓	✓						
FIA_UIA_EXT.1	✓	✓						
FIA_UAU_EXT.2	✓	✓						
FIA_UAU.7	✓	✓						
FIA_X509_EXT.1	✓	✓				✓		
FIA_X509_EXT.2	✓	✓						
FMT_MOF.1/ManualUpdate						✓		
FMT_MTD.1/CoreData	✓							
FMT_MTD.1/CryptoKeys					✓			
FMT_SMF.1	✓				✓	✓	✓	
FMT_SMR.2	✓							
FPT_SKP_EXT.1					✓			
FPT_APW_EXT.1		✓						
FPT_TST_EXT.1								✓
FPT_TUD_EXT.1						✓		
FPT_STM_EXT.1							✓	
FTA_SSL_EXT.1	✓							
FTA_SSL.3	✓							
FTA_SSL.4	✓							
FTA_TAB.1	✓							
FTP_ITC.1			✓				✓	
FTP_ITC.1/QKD				✓				
FTP_TRP.1/Admin	✓		✓					

Table 99. Mapping between SFRs and TOE objectives

O.TOE_ADMINISTRATION

Enforced admin identification & authentication and roles (FIA_UIA_EXT.1, FIA_UAU_EXT.2, FMT_SMR.2),

password strength/lockout/feedback (FIA_PMG_EXT.1, FIA_AFL.1, FIA_UAU.7), pre-login banner (FTA_TAB.1), safe session handling (FTA_SSL_EXT.1, FTA_SSL.3, FTA_SSL.4), secure remote path via SSH (FTP_TRP.1/Admin + FCS_SSHS_EXT.1), and cert-based peer auth where applicable (FIA_X509_EXT.1/.2). This implements the objective's requirement for authenticated high-privilege admin over secure channels with a pre-login banner. The administrative actions that the user can done are defined in (FMT_SMF.1) and can be only executed by administrators (FMT_MTD.1/CoreData).

O.STRONG_AUTHENTICATION
Strong passwords and lockout (FIA_PMG_EXT.1, FIA_AFL.1, FIA_UAU.7), enforced admin I&A (FIA_UIA_EXT.1, FIA_UAU_EXT.2), X.509 validation and use for protocol authentication (FIA_X509_EXT.1/.2) SSH protocol (FCS_SSHS_EXT.1) for administrators and (FCS_TLSS_EXT.1/.2) for remote entities, plus protection of stored admin passwords (FPT_APW_EXT.1). This directly counters weak/guessable auth endpoints described for this objective.

O.PROTECTED_COMMUNICATIONS
Trusted channels and path (FTP_ITC.1, FTP_TRP.1/Admin), TLS client/server including mutual auth (FCS_TLSC_EXT.1/.2, FCS_TLSS_EXT.1/.2), SSH server (FCS_SSHS_EXT.1), with approved crypto operations, key management and RBG (FCS_COP.1/*, FCS_CKM.1/.2/.3/.6, FCS_RBG_EXT.1). This ensures confidentiality/integrity/peer auth of sensitive exchanges required by the objective including secure export of audit (FAU_STG_EXT.1) that uses those channels.

O.PROTECTED_KME_COMM
Inter-KME trusted channel protected with post-quantum KEM (FTP_ITC.1/QKD; FCS_CKM.2/QKD) using ML-KEM, appropriate key lifecycle (FCS_CKM.6) and cryptographic integrity primitives/RBG (FCS_COP.1/KeyedHash, FCS_COP.1/Hash, FCS_RBG_EXT.1). This addresses the objective's requirement to resist Post-Quantum cryptanalysis on KME links.

O.STRONG_CRYPTOGRAPHY
Approved algorithms and sizes (FCS_COP.1/*), robust key generation/establishment/access/destruction (FCS_CKM.1/.2/.2/QKD/.3/.6), high-quality randomness (FCS_RBG_EXT.1), protection against reading keys (FPT_SKP_EXT.1), and controlled crypto/key management (FMT_MTD.1/CryptoKeys, FMT_SMF.1). This is exactly the substance of the objective.

O.UPDATES
Trusted update with pre-install verification (FPT_TUD_EXT.1), only admins can initiate (FMT_MOF.1/ManualUpdate), management functions for updates (FMT_SMF.1); update actions auditable (FAU_GEN.1/2). This fulfils the objective to prevent compromised updates.

O.AUDIT
Audit generation/association (FAU_GEN.1/2), reliable timestamps (FPT_STM_EXT.1), protected storage and secure export (FAU_STG_EXT.1 with FTP_ITC.1), and management of audit behaviour (FMT_SMF.1). That enables detection of unauthorized activity per the objective.

O.TSF_SELF_TEST
Self-tests at boot/on demand for crypto and firmware integrity (FPT_TST_EXT.1), with auditable outcomes (FAU_GEN.1), meeting the objective to detect and fail safely on TSF malfunctions.

6.3.2 Security Assurance requirements (SAR) Rationale

The security assurance requirements are those from the predefined package of security requirements EAL2 as defined by [CC5]. This assurance level has been chosen commensurate with the threat environment that is experienced by typical consumers of the TOE.

7 TOE SUMMARY SPECIFICATION

This section provides evaluators and potential consumers of the TOE with a high-level description of each SFR, thereby enabling them to gain a general understanding of how the TOE is implemented. These descriptions are intentionally not overly detailed, thereby disclosing no proprietary information. These sections refer to SFRs defined in Section 6.1.

7.1.1 Security Audit

The TOE generates audit records for the events defined in Table 77Table 7 (e.g., admin logon/logoff, TSF configuration changes with value changed, key lifecycle operations with unique key reference, password resets) and includes timestamp, event type, subject identity (when applicable), and outcome the events list and contents are as specified in (FAU_GEN.1, FAU_GEN.2).

It ensures that all relevant security events are captured and logged in an auditable manner. At start-up, the system executes cryptographic algorithm self-tests through the txfips process and firmware integrity checks through the txscan process. These activities occur before the module enters operational mode, and both their successful completion and any failures are reported as status events that can be exported to syslog. In this way, the start-up and shut-down of the audit functions are fully covered (FAU_GEN.1).

Administrative actions are also comprehensively audited. When administrators log in or log out of the system via SSH/CLI, the events are authenticated and logged with the user identity and the outcome of the attempt. Configuration changes performed with utilities such as tx_install_cf and tx_get_cf are validated and reported with sufficient detail to show not only that a change occurred but also what was modified. The management of cryptographic keys is strictly controlled, and every generated, imported, or deleted key is associated with a universally unique identifier (key_ID) as defined in ETSI GS QKD 014, which ensures that all key lifecycle events are traceable. Similarly, password resets executed via the CLI utilities for GUI user management are logged with the specific username affected (FAU_GEN.1).

The TOE produces comprehensive and standards-compliant audit records. It captures all required administrative, key management, and system events, and includes in each record the date and time, type of event, subject identity, outcome, and event-specific information such as unique key identifiers or usernames. Logs are exported through syslog to enterprise monitoring systems, ensuring effective compliance, accountability, and forensic readiness (FAU_GEN.1).

The TOE records all mandatory information in each audit record. Every log entry includes a date and time stamp, provided by syslog and synchronized with system configuration through the tx_set_tz utility. The type of event is evident from the service or command invoked, such as an ETSI Get key request or a CLI tx_reset. The subject identity is always recorded, whether it is an administrator identified by Debian login or TLS certificate, or a user client identified by SAE and KME IDs in the ETSI QKD protocol. Each record also specifies the outcome of the operation, whether success or failure (FAU_GEN.2).

Audit records are stored locally (with oldest-first overwrite on full) and can be transmitted to an external audit IT entity over a trusted channel (FAU_STG_EXT.1, FTP_ITC.1). In practice, the product exposes configurable Syslog forwarding as part of the node configuration. In the TOE the locally stored audit records are handled through an automatic log rotation mechanism. The system writes logs to /var/log/tx.log, and every 24 hours the file is rotated. The previous day's log is compressed using gzip and renamed with the corresponding date.

Audit records include reliable timestamps provided by the TSF and, when configured, synchronized with NTP servers (FPT_STM_EXT.1). This configuration is done through a YAML file.

7.1.2 Cryptographic Support

The TOE generates asymmetric cryptographic keys using FIPS-approved mechanisms (FCS_CKM.1). RSA key pairs of 2048 and 3072 bits are supported for TLS and SSH operations, generated according to FIPS 186-4. ECC key pairs on NIST curves P-256 and P-384 are supported for ECDSA signatures and ECDHE key exchange. These keys are generated internally by the OpenSSL library wrapped by the TOE and tested during start-up using known-answer tests (`txfips`) before being used.

For key establishment (FCS_CKM.2), the TOE implements RSA-based key establishment for TLS handshake and elliptic-curve key agreement using ECDHE on P-256/P-384 curves. TLS conforms to RFC 5246 for TLS 1.2 and RFC 8446 for TLS 1.3. Keys are exchanged securely during TLS and SSH sessions, using ephemeral values that provide forward secrecy.

In addition, the TOE supports key distribution between KMEs using post-quantum ML-KEM encapsulation (FCS_CKM.2/QKD). This function is provided as part of the Phio Hive architecture, where ML-KEM key encapsulation and decapsulation, standardized in FIPS 203, protect symmetric keys exchanged between KMEs. Configuration of PQC key pairs is performed via the CLI configuration tools before the `tx` daemon is started, ensuring that only authorized KMEs can establish these secure channels.

The TOE provides internal access to cryptographic keys but does not support external archival or recovery mechanisms (FCS_CKM.3). Key access is restricted to internal processes, with no user-accessible interfaces to export or recover key material.

For key destruction (FCS_CKM.6), the TOE destroys SSH and TLS private keys when an administrator issues a reset or zeroization command (`tx_reset`), and ephemeral KEM-generated keys are automatically destroyed after a single use. Destruction is implemented by overwriting key material in memory with zeroes, followed by release of memory, in accordance with FIPS 140-3 requirements.

The TOE performs AES encryption and decryption (FCS_COP.1/DataEncryption) in CBC, CTR, and GCM modes with key sizes of 128, 192, and 256 bits. These algorithms are provided through OpenSSL 1.1.1, validated under CAVP testing, and used to protect TLS and SSH sessions as well as storage of integrity references.

For digital signatures (FCS_COP.1/SigGen), the TOE supports RSA signature generation and verification with 2048- and 3072-bit keys, and ECDSA using P-256 and P-384 curves. Both PKCS#1 v2.1 (RSASSA-PSS) and ECDSA signature algorithms are supported, as required by FIPS 186-4. These are used for TLS authentication, SSH authentication, and certificate verification.

The TOE implements cryptographic hashing (FCS_COP.1/Hash) using SHA-256, SHA-384, and SHA-512, with digest sizes of 160, 256, 384, and 512 bits respectively. These functions are used in signature verification, TLS and SSH key derivation, and software integrity checks.

Keyed hash authentication is implemented through HMAC with SHA-256, SHA-384, and SHA-512 (FCS_COP.1/KeyedHash). Key sizes of 256, 384, and 512 bits are supported, and HMAC is used for TLS record protection, SSH MACs, and firmware integrity verification (`txscan`).

Random numbers are generated using a CTR_DRBG with AES (FCS_RBG_EXT.1). The DRBG is seeded from entropy sources including the Linux kernel's getrandom() noise source and platform-based noise from the Protectli FW6A hardware, ensuring at least 256 bits of entropy. This seeding meets the security strength requirements defined in ISO/IEC 18031 Table C.1. The DRBG is used for all key generation, IVs, and nonces in TLS and SSH.

The TOE implements SSH in accordance with RFCs 4251–4254 and related extensions (FCS_SSHS_EXT.1). It supports password-based and public key-based authentication using ECDSA on P-256 and P-384. The SSH

transport layer uses AES in GCM mode for encryption, with HMAC-SHA-256 or HMAC-SHA-512 for integrity. Only ecdh-sha2-nistp256, ecdh-sha2-nistp384 and *ecdh-sha2-nistp521* are permitted for key exchange. The TOE enforces rekeying at least once per hour or per gigabyte of protected data, whichever occurs first. Packets larger than the allowed maximum are dropped as required by RFC 4253

For TLS client functionality (FCS_TLSC_EXT.1), the TOE implements TLS 1.2 and 1.3 only, rejecting SSLv2, SSLv3, TLS 1.0, and TLS 1.1. It supports only FIPS-approved cipher suites including RSA and ECDHE with AES-256 in CBC and GCM modes, and PSK suites defined in RFCs 4279 and 5487. Certificate validation is strict: the identifier must match the reference identifier per RFC 6125, and invalid server certificates are rejected with no administrator override. During the ClientHello, the TOE offers only secp256r1 and secp384r1 as supported groups (FCS_TLSC_EXT.1.4).

The TOE also supports TLS mutual authentication as a client (FCS_TLSC_EXT.2), using X.509v3 certificates to establish trust with servers.

As a TLS server (FCS_TLSS_EXT.1), the TOE accepts only TLS 1.2 and TLS 1.3 with the same restricted cipher suites as the client. Connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, or TLS 1.1 are denied. Key establishment is limited to secp256r1/secp384r1 curves. Session resumption and tickets are not supported, ensuring that each TLS session performs a fresh handshake.

The TOE supports TLS mutual authentication for clients (FCS_TLSS_EXT.2). X.509v3 client certificates are required, and the TOE will not establish a channel if the client certificate is invalid or the subject identifier does not match the expected identity. There is no administrator override of certificate validation.

7.1.3 Identification and Authentication

The TOE enforces authentication failure management by monitoring login attempts for administrators connecting via SSH. The Debian underlying operating environment is configured with PAM settings that enforce a minimum password length and count failed attempts. After a defined number of failed logins, the account is locked, and further attempts are blocked (FIA_AFL.1). The system requires explicit administrative action from another user via CLI utilities to re-enable the locked account, preventing the offending administrator from establishing remote sessions using password authentication until this unlock command is executed (FIA_AFL.1).

Password management is implemented using the native Linux PAM framework. Passwords may contain any printable ASCII character, including letters, digits, and special symbols (FIA_PMG_EXT.1). The TOE enforces a minimum of eight characters by default, as configured in `/etc/pam.d/common-password`, while supporting configurable values up to 256 characters through system policies (FIA_PMG_EXT.1). This ensures administrators are required to use strong, complex credentials resistant to brute-force attacks.

All administrator actions are mediated by identification and authentication. When connecting, the TOE requires the administrator to present valid credentials before granting access to TSF-mediated services (FIA_UIA_EXT.1). Before this process, the TOE displays a mandatory banner notice to the operator, ensuring legal notification and user acknowledgment prior to authentication (FIA_UIA_EXT.1). This is implemented through the configured Debian login process and enforced before access to CLI services. In the evaluated configuration, all administrative access is performed using a single administrative account (qxc). As a result, once identification and authentication have been successfully completed, all administrative actions performed on the TOE are attributable to the qxc account.

Multiple authentication mechanisms are available to administrators. For remote sessions, the TOE provides password-based authentication as well as SSH public key-based authentication (FIA_UAU_EXT.2). Locally at the console, the password input is masked so that feedback is obscured, preventing cleartext exposure during authentication (FIA_UAU.7). This obscured feedback mechanism is inherent in the PAM login and SSH client handling, where typed characters are not echoed on the screen.

The TOE requires administrator identification and authentication before any TSF-mediated action (local console or remote) and obscures authentication feedback (FIA_UIA_EXT.1, FIA_UAU_EXT.2, FIA_UAU.7). The TOE specifies password policy controls (composition/length) and failure handling thresholds/lockout (FIA_PMG_EXT.1, FIA_AFL.1).

The TOE validates X.509 certificates strictly according to RFC 5280 (FIA_X509_EXT.1). The ETSI GS QKD 014 specification governs REST-based interactions between Secure Application Entities (SAEs) and Key Management Entities (KMEs), requiring mutual TLS authentication. Each certificate presented must pass full path validation, terminating at a trusted CA designated as a trust anchor. Intermediate CA certificates must contain the basicConstraints extension with the CA flag set to TRUE (FIA_X509_EXT.1). Revocation checking is performed using CRLs, ensuring that expired or revoked certificates cannot be accepted (FIA_X509_EXT.1). The extendedKeyUsage field is verified to ensure the certificate is used only for approved purposes, such as TLS client or server authentication (FIA_X509_EXT.1).When client verification is enabled for a service, the TOE authenticates TLS clients by validating the presented client certificate chain to a trusted CA (per FIA_X509_EXT.1) and checking the client's identifier (e.g., CN/reference identifier) as applicable for the protocol, consistent with RFC 6125.

For authentication of entities, the TOE uses X.509v3 certificates to establish trust in TLS for ETSI REST services and in SSH for administrative sessions (FIA_X509_EXT.2). The TOE rejects connections if certificate validity cannot be established, either because the trust chain is incomplete or because revocation status cannot be determined (FIA_X509_EXT.2). This strict enforcement ensures that only authenticated and trusted endpoints can establish sessions with the TOE. At the management level, certificates are installed, generated, and removed through dedicated CLI utilities such as tx_conf commands. These allow an administrator to load PKI material including root CA files, service certificates, and private keys that are required for secure communications.

The TOE exposes only network services listed in its configuration (tx.yaml). For each listed service, optional source IP/subnet filters restrict which client origins may connect. If a service is configured with PKI client verification, the TOE requires client certificate authentication before granting access and validates the client identity (e.g., CN/reference identifier) per policy.

7.1.4 Security Management

The TOE restricts the ability to perform manual updates to its security functions exclusively to Security Administrators (FMT_MOF.1/ManualUpdate). Updates are executed through authenticated CLI commands, such as `tx_install_cf` for installing configuration files or `tx_install_license` for loading license data. These operations require administrator credentials and are validated by the TOE before changes are applied. The process includes integrity verification, for example through `txscan`, which performs HMAC-based checks over firmware and software components to confirm authenticity prior to activation.

The TOE enforces that installation of updates is executed with root privileges. Operationally, the Security Administrator performs downloads and checksum verification as an administrator, and then executes the installation using a controlled privilege-escalation mechanism (e.g., sudo) to run the installer as root.

The TOE provides two supported installation paths:

- Local package installation via phiotx_<version>_install_deb_x86_64
- Automated secure download and installation via tx_upgrade -url <url> -sha256 <hash>

Both methods require root for the installation step. In the evaluated configuration, TLS peer and host verification are enforced for tx_upgrade. The administrator must supply an expected SHA-256 hash for the downloaded package.

This design satisfies FMT_MOF.1 by restricting update installation to authorized administrators and leveraging the TOE's enforcement of root-only execution for the critical install step. The separation between non-privileged preparation (download, checksum) and privileged installation preserves security while maintaining administrative control.The management of TSF data is similarly limited to Security Administrators (FMT_MTD.1/CoreData). Administrative utilities within the tx_conf suite control system-level configuration, network parameters, and access control data. These include functions such as `tx_get_cf`, `tx_yank_nic`, and `tx_list_nics`, all of which require authenticated administrator access before they can be executed. No unprivileged user or unauthenticated service has the ability to manage this data.

The management of cryptographic keys is strictly enforced (FMT_MTD.1/CryptoKeys). Keys are generated and distributed using the ETSI GS QKD 014 REST API or internal KEM-based services, and each key is assigned a unique identifier (key_ID). Only Security Administrators are able to install, update, or remove certificates and keys through the TOE's certificate management services. Sensitive Security Parameters such as TLS private keys, session keys, and key exchange material are never accessible in plaintext form, and are generated, stored, and zeroized in accordance with FIPS 140-3 requirements.

Security Administrators can administer the TOE both locally and remotely using the CLI via SSH or console access. They can configure the access banner shown at login, enforce session inactivity timeouts that lead to automatic termination or lockout, and update the TOE software while verifying updates through cryptographic hash checks prior to installation. They are also able to configure authentication failure thresholds that support the lockout mechanism described under (FIA_AFL.1). Beyond these baseline functions, the TOE allows administrators to start and stop services, configure the system time zone (`tx_set_tz`), set NTP synchronization, manage the trust store, import X.509 certificates as trust anchors, re-enable locked administrator accounts, and manage cryptographic keys and certificate chains (FMT_SMF.1).

The TOE maintains a single defined role, the Security Administrator. All services, including configuration management, key and certificate handling, and update operations, are tied to the Security Administrator role. No other role exists, and unprivileged or anonymous access is not permitted to manage any aspect of the TOE (FMT_SMR.2).

7.1.5 Protection of the TSF

The TOE prevents the reading of all pre-shared keys, symmetric keys, and private keys (FPT_SKP_EXT.1). Sensitive Security Parameters (SSPs) such as TLS session keys, SSH keys, and DRBG states are generated and stored only inside the cryptographic boundary of the FIPS 140-3 validated module. Keys are never output in plaintext, and their use is restricted to internal cryptographic operations. When keys are zeroized, commands such as `tx_reset` ensure that all values are securely destroyed. Certificate private keys, TLS session keys, and ephemeral KEM values are all handled according to the module's FIPS 140-3 design, which prohibits plaintext extraction.

Administrative passwords are stored in non-plaintext form (FPT_APW_EXT.1). Within the Debian environment, passwords are hashed using SHA-512 via PAM configuration in /etc/pam.d/common-password. These hashed values are the only stored representation of administrator passwords, and plaintext recovery is impossible. When administrators authenticate, their inputs are compared to the stored hash values without

revealing the original password. Plaintext passwords are never written to disk or logs, preventing reading of cleartext administrative credentials (FPT_APW_EXT.1).

he TOE implements a comprehensive self-test mechanism to verify the correct functioning of cryptographic algorithms and software integrity (FPT_TST_EXT.1). At every start-up, the `txfips` process executes algorithmic known-answer tests (CASTs) using NIST ACVP vectors for AES, SHA, HMAC, and other primitives. Following this, the `txscan` process checks firmware integrity by computing and verifying HMAC values against a trusted reference database. These self-tests are mandatory at power-on and can also be initiated by an authorized administrator on demand. If any test fails, the TOE transitions into an error state where no cryptographic services are provided until the fault is resolved.

The TOE enforces strict trusted update procedures (FPT_TUD_EXT.1). Administrators can query the running version of the software using the tx_status command, which outputs the current module identifier and version information. No other version data is exposed (FPT_TUD_EXT.1). Updates can only be initiated manually by Security Administrators using CLI services such as tx_install_cf and txscan, which verify package integrity before activation. No automatic or background update mechanisms exist (FPT_TUD_EXT.1). To ensure authenticity, firmware and software updates are validated by comparing a published HMAC or hash signature before installation, ensuring that only trusted and unmodified updates are deployed (FPT_TUD_EXT.1). The vendor publishes a cryptographic hash of each update package that the Administrator, prior to invoking the installer, may verify out-of-band as a procedural integrity check. This published hash is not provided to, nor consumed by, the installer itself.

The TOE provides reliable time stamps for its own use (FPT_STM_EXT.1). The Debian operating environment maintains system time through the standard Linux time services, which are configured during installation. Administrators can set the local timezone via the tx_set_tz command, and the TOE supports synchronization with external NTP servers to maintain accuracy (FPT_STM_EXT.1). This ensures that all logs, audit records, and cryptographic operations requiring time references are associated with reliable and synchronized timestamps

7.1.6 TOE Access

The TOE enforces session locking for local interactive sessions (FTA_SSL_EXT.1). Local administrator access is handled through the Debian console login process. Session inactivity is managed by administrator-configurable timeout parameters that terminate the session after the defined idle period. This configuration is controlled through PAM and systemd settings in the operating environment, which are set by the Security Administrator during TOE installation and configuration. By terminating the console session after the timeout, the TOE prevents unattended local consoles from remaining open.

For remote interactive sessions, the TOE applies inactivity termination (FTA_SSL.3). Administrative access over SSH is enforced with OpenSSH. The `ClientAliveInterval` and `ClientAliveCountMax` directives are available to Security Administrators to configure a precise idle timeout for SSH sessions. After the interval elapses without activity, the TOE automatically closes the remote session. This ensures that inactive remote sessions cannot be exploited by attackers and maintains strict control over SSH administrative access.

Administrators also retain the ability to terminate their own sessions at any time (FTA_SSL.4). In practice, this is done by issuing the `exit` command in an SSH or console shell. The TOE accepts the termination request immediately, and the session is closed. This ensures administrators remain in full control of their sessions whenever they wish to end them.

Before establishing an administrative session, the TOE displays an advisory banner (FTA_TAB.1). The banner text is defined by the Security Administrator and provides a consent and usage warning that must be displayed

before login. This ensures that legal and organizational policy requirements for notice and consent are satisfied before any authentication credentials are entered.

7.1.7 Trusted Path/Channels

The TOE establishes trusted channels with Secure Application Entities (SAEs) using TLS (FTP_ITC.1). The ETSI GS QKD 014 REST API defines the interactions between the TOE, acting as a Key Management Entity (KME), and its connected SAEs. All such communications occur over TLS v1.2 or v1.3 with mutually authenticated X.509v3 certificates. This ensures that both endpoints are positively identified and that the data exchanged—specifically, symmetric key material—is protected against disclosure and modification during transmission. The trusted TLS channel is logically separated from other communication paths, isolating key distribution from other system traffic. In this communication the TOE acts as server and the SAEs are the clients, and so communication initiators. Administrators configure the trust anchors by loading CA certificates into the TOE's trust store, and connections are only permitted if the presented certificate chains terminate in a trusted CA. Configuration files defining authorized SAEs are YAML-based and must be installed via `tx_install_cf`, which validates them before service activation.

In addition to SAE communication, the TOE also establishes trusted inter-TSF channels with peer KMEs (FTP_ITC.1/QKD). These channels use post-quantum Key Encapsulation Mechanisms (ML-KEM, standardized as FIPS 203) to secure the exchange of symmetric keys between KMEs. Each KME connection is uniquely identified, and the cryptographic encapsulation ensures confidentiality and integrity of exchanged key material even in the presence of quantum-capable adversaries. Either the TOE or a peer KME may initiate a channel (FTP_ITC.1.2/QKD), and the TOE initiates trusted channels specifically to exchange symmetric keys with other KMEs (FTP_ITC.1.3/QKD). This dual use of TLS for SAE communications and ML-KEM for KME-to-KME exchanges ensures that all trusted key delivery paths are cryptographically robust, logically distinct, and tamper-resistant.

Administrators must install the required post-quantum public/private key pairs on each participating KME, using the CLI configuration utilities (`tx_conf` for key material and configuration management). The YAML configuration files specify which peer KMEs are authorized to connect, and these are validated before the `tx` daemon is allowed to start. Once the configuration and PQC key material are installed, either the TOE or a peer KME can initiate the encapsulation exchange (FTP_ITC.1.2/QKD).

For administration, the TOE provides a dedicated trusted path using SSH (FTP_TRP.1/Admin). Remote administrators authenticate and manage the TOE exclusively over SSH, which establishes a secure channel ensuring confidentiality and integrity of administrative traffic. The trusted path is logically distinct from operational key distribution and REST interfaces. Remote administrators may initiate these connections (FTP_TRP.1.2/Admin), and the TOE mandates the use of SSH as the only valid channel for remote administration. Initial authentication, as well as all subsequent administrative actions, are performed through this secure channel (FTP_TRP.1.3/Admin). Administrator credentials are never transmitted in plaintext, and session traffic is fully protected against eavesdropping or modification.

Certificate-based SSH keys are installed into the TOE by Security Administrators through configuration files.

8 REFERENCES AND ABBREVIATIONS

8.1 References

[CC1] Common Criteria for Information Technology Security Evaluation. Part 1: Introduction to General Model, CC:2022 Revision 1

[CC2] Common Criteria for Information Technology Security Evaluation. Part 2: Security functional components, CC:2022 Revision 1

[CC3] Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components, CC:2022 Revision 1

[CC4] Common Criteria for Information Technology Security Evaluation. Part 4: Framework for the specification of evaluation methods and activities, CC:2022 Revision 1

[CC5] Common Criteria for Information Technology Security Evaluation. Part 5: Pre-defined packages of security requirements, CC:2022 Revision 1

[ETSIQKD] ETSI GS QKD 014 - Quantum Key Distribution (QKD); Protocol and data format of REST-based key delivery API, V1.1.1, 2019-02

[PPND] collaborative Protection Profile for Network Devices, Version 2.2e

8.2 Abbreviations

ACVP	Automated Cryptographic Validation Protocol
AES	Advanced Encryption Standard
API	Application Programming Interface
ASCII	American Standard Code for Information Interchange
CA	Certification Authority
CASTs	Known Answer Tests
CBC	Cipher Block Chaining
CC	Common Criteria
CC2	Common Criteria Part 2
CC3	Common Criteria Part 3
CLI	Command Line Interface

CN	Common Name
cPP	collaborative Protection Profile
CRL	Certificate Revocation List
CSR	Certificate Signing Request
CTR	Counter
CTR_DRBG	Counter Deterministic Random Bit Generator
DH	Diffie-Hellman
DHE	Diffie Hellman Ephemeral
DRBG	Deterministic Random Bit Generator
DSS	Digital Signature Standard
DTLS	Datagram Transport Layer Security
EAL	Evaluation Assurance Level
EAL2	Evaluation Assurance Level 2
ECC	Elliptic Curve Cryptography
ECD	Extended Component Definition
ECDH	Elliptic Curve Diffie Hellman
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
EncapDecap	Encapsulation Decapsulation
ESD	Electrostatic Discharge
ETS	European Telecommunications Standards Institute
ETSI	European Telecommunications Standards Institute
FQDN	Fully Qualified Domain Name
GCM	Galois Counter Mode

GS	Group Specification
GUI	Graphical User Interface
HMAC	Hash-based Message Authentication Code
HQC	Hamming Quasi-Cyclic
HSMs	Hardware Security Modules
HTTPS	Hypertext Transfer Protocol Secure
ID	Identifier
IEC	International Electrotechnical Commission
IP	Internet Protocol
IPsec	Internet Protocol Security
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
ISO	International Organization for Standardization
IT	Information Technology
IVs	Initialization Vectors
KB	Kilobyte
KEM	Key Encapsulation Mechanism
KEX	Key Exchange
KeyGen	Key Generation
KME	Key Management Entity
KMEs	Key Management Entities
LCD	Liquid Crystal Display
LED	Light Emitting Diode
MAC	Message Authentication Code

MACs	Message Authentication Codes
ML-KEM	Module Lattice-based Key Encapsulation Mechanism
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
OE	Organizational Environment
OSP	Organizational Security Policies
PAM	Pluggable Authentication Module
PINs	Personal Identification Numbers
PKCS	Public-Key Cryptography Standards
PKI	Public Key Infrastructure
PP	Protection Profile
PPND	Protection Profile for Network Devices
PQ	Post Quantum
PQC	Post-Quantum Cryptography
PSK	Pre Shared Key
PUB	Publication
QKD	Quantum Key Distribution
RBG	Random Bit Generator
REST	Representational State Transfer
RFC	Request For Comments
RJ-45	Registered Jack 45
RSA	Rivest Shamir Adleman
RSAES	RSA Encryption Scheme

RSASSA-PKCS1v1_5	RSASSA PKCS1 version 1 5
RSASSA-PSS	RSASSA Probabilistic Signature Scheme
SA	Security Association
SAE	Secure Application Entity
SAEs	Security Association Entities
SAN	Subject Alternative Name
SAR	Security Assurance Requirements
SARs	Security Assurance Requirements
SFP	Small Form-factor Pluggable
SFR	Security Functional Requirement
SFRs	Security Functional Requirements
SHA	Secure Hash Algorithm
SigGen	Signature Generation
SKIP	Simple Key-management for Internet Protocols
SSH	Secure Shell
SSL	Secure Sockets Layer
SSPs	Sensitive Security Parameters
ST	Security Target
TLS	Transport Layer Security
TLSC	Transport Layer Security Client
TLSS	Transport Layer Security Server
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSS	TOE Summary Specification

TX	Trusted Xchange
UIA_EXT	User Identification and Authentication Extension
VPNs	Virtual Private Networks
X.509	X dot 509
X.509v3	X Version 3
X509	X Internet Engineering Task Force Recommendation 509
YAML	YAML Ain't Markup Language