qCrypt-xStream R1.1 Security Target

Common Criteria: EAL2

Document Revision: 1.0

Document ID: 0001363

Document Date: 16-Mar-15

Document Information

Document identification

Document title	qCrypt-xStream R1.1 Security Target
Document id	0001363
Document date	16-Mar-15
Author	Muzamir Mohamad
Reviewed and approved by	John Leiseboer, QuintessenceLabs CTO
Release Authority	Jason Chapman

Document history

Revision	Date	Description
0.1	28-AUG-2014	Released for internal review.
0.2	23-SEP-2014	Released to evaluator
0.3	29-DEC-2014	Updated to address EOR-ASE v1.0
0.4	19-FEB-15	Updated to address issues in SFRs and TSS.
1.0	16-Mar-15	Released as FINAL

Table of Contents

1 Security Target introduction (ASE_INT.1)

1.1 Background

Cryptography services are used to protect the confidentiality, integrity, authenticity and non-repudiation of data residing and traversing across information systems. The trust and integrity lies within the security of the underlying signing and encryption keys. Thus, protection of these keys is critical to ensure the overall trust and integrity of the information system itself.

Cryptographic key material can be stored and protected in a variety of ways and on a variety of media including software, smart cards and USB tokens. However, where protection is critical and potentially of national security importance, the use of these types of key storage devices is considered inadequate.

QuintessenceLabs has designed the qCrypt-xStream™, which provides the following features:

a) Secure replication of policies and managed cryptographic objects between nodes and support for both centralized and distributed deployment topologies to suit any availability and performance requirements

b) Hierarchical replication allowing for both logical and physical segregation of managed cryptographic objects into separate domains to meet multi-level, regulatory, and operational security needs

c) Granular, hierarchical and auditable access control

d) KMIP (Key Management Interoperability Protocol) 1.0, and 1.1 conformant

e) Compliance with the Key Management recommendations from NIST SP800-57 Part 1

f) For qCrypt-xStream™, a true random quantum entropy source, guaranteeing the strongest cryptographic keys

g) Support for one-time-pad cryptography with QRNG seeding to facilitate timeless data protection, today

h) High key generation rate and management of high key volumes for demanding applications, rapid rotation requirements and/or large-scale enterprise projects

The qCrypt-xStream™, is ideally suited to businesses deploying a cryptographic system where the protection of cryptographic keys is a priority, for example, in organizations requiring certificate signing, code or document signing, bulk generation or ciphering of keys or data.

1.2 ST reference

ST Title	qCrypt-xStream Security Target
ST Version	1.0
ST Date	16-Mar-15
TOE Reference	qCrypt-xStream R1.1, consisting of the two main components: • Appliance qCrypt-xStream R1.1 – Part number 1362 • Quantum Random Number Generator (QRNG) card - Part number 1108.
CC Identification	Common Criteria for Information Technology (IT) Security Evaluation, Version 3.1 (REV 4) September 2012, incorporating: • Part One – Introduction and General Model (Ref. [1]), • Part Two – Security Functional Components (Ref. [2]), and • Part Three – Security Assurance Components (Ref. [3]).

1.3 Document organization

This document is organized into the following major sections:

- Section 1 provides the introductory material for the ST as well as the TOE description (ASE_INT.1).
- Section 2 provides the conformance claims for the evaluation (ASE_CCL.1).
- Section 3 provides the definition of the security problem that the TOE has been designed to address (ASE_SPD.1).
- Section 4 defines the security objectives for the TOE and the environment (ASE_OBJ.2).
- Section 5 contains the security functional and assurance requirements derived from the Common Criteria, Part 2 and 3 respectively, which are in turn satisfied by the TOE and the development lifecycle (ASE_REQ.2).
- Section 6 provides a summary of the TOE specification, identifying the IT security functions provided by the TOE (ASE_TSS.1).

1.4 References

[1] Common Criteria for Information Technology Security Evaluation: Part 1: Introduction and general model, September 2012, Version 3.1 Revision 4,

[2] Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, September 2012, Version 3.1, Revision 4,

[3] Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, September 2012, Version 3.1, Revision 4.

1.5 Defined terms

The following table defines all subjects, objects, operations, security attributes, external entities and other key terms that are used within the statements of security functional and assurance requirements. It also describes the acronym used in this documentation.

Table 1 – Defined terms

Term/Acronym	Definition
Administrator	A role that performs TOE initialisation and general TOE administrative functions.
Authentication	The process used to verify the claimed identity of a user.
FIPS 140-2	FIPS-140-2 is the National Institute of Standards and Technology (NIST) technical publication defining cryptography modules standards.
FIPS 180-2	FIPS-180-2 is the National Institute of Standards and Technology (NIST) technical publication defining Secure Hash standards.
FIPS 186-4	FIPS 186-4 defines the Digital Signature Standard (DSS)
FIPS 197	FIPS 197 defines the Advanced Encryption Standard (AES)
FIPS 198-1	FIPS 198-1 defines the Keyed Hash Message Authentication Code (HMAC)
NIST SP 800-67	NIST Special Publication 800-67 defines the Triple Data Encryption Algorithm (TDES)
NIST SP 800-90	NIST Special Publication 800-90 describes the NIST Recommendation for Random Number Generation Using Deterministic Random Bit Generators.
NIST SP 800-133	NIST Special Publication 800-133 defines the NIST Recommendation for Cryptographic Key Generation
Root	The administrator (super user) of the underlying Operating Systems, who has unrestricted access to all the TOE resources and functions.
RSA	An algorithm for public-key cryptography published by RSA Laboratories
RSA PKCS #1	PKCS #1 defines the first Public Key Cryptography Standard published by RSA.
RFC 4346	RFC 4346 defines version 1.1 of the Transport Layer Security (TLS) protocol.

Term/Acronym	Definition
RFC 5246	RFC 5246 defines version 1.2 of the Transport Layer Security (TLS) protocol.
Service	A specified function or set of functions implemented by the TOE and accessible to authorised users.
SSH	Secure Shell – network protocol used for cryptographically securing data transmitted between the TOE and external clients.
TLS	Transport Layer Security, used for securing data in transmission between the TOE and external clients.
TSF data	Data created by and for the TOE, that might affect the operation of the TOE
User	Any entity (human user or external IT entity) outside the TOE that interacts with the TOE.
User data	Data created by and for the user that does not affect the operation of the TOE.
Web UI	The administrative user interface provided by the TOE for the management of TOE resources and functions.

1.6 TOE overview

1.6.1 TOE usage and major security features

The TOE is a cryptographic key management appliance, designed to centrally manage enterprise digital keys and certificates for enterprise applications, users and devices throughout their full life cycle, including key generation, distribution, usage, automated rotation and renewal in line with TOE-defined policy. The TOE can be deployed as part of any cryptographic system that uses digital keys. The TOE is intended to provide a high-level of assurance in protection of the digital keys, especially keys that are of high-value, to avoid negative impact on the system if the keys were to be compromised.

The powerful synthesis of the TOE with its key management functionality delivers significant cost-effective benefits and efficiencies in the operational, incident and change management processes.

In the context of this ST the TOE is expected to provide the following major security features:

a) Secure generation, distribution and destruction of cryptographic keys;

b) On-board cryptographic functions to secure traffic sent between the TOE and external users;

c) Secure storage and management of keys throughout their lifecycle;

d) Role-based authentication and access control mechanisms to facilitate controlled access to cryptographic key management and TOE management functions by trusted personnel only;

e) Functionality to detect errors in received traffic or replay attacks;

f) Auditing of security relevant events to provide suitable accountability;

g) Protection of stored audit data to prevent modification or accidental deletion; and

h) Self-test of the core cryptographic functions and algorithms of the TOE.

1.6.2 TOE Type

The TOE centrally manages enterprise digital keys and certificates in line with TOE defined policy. The TOE can be categorised as a ***key management system*** in accordance with the categories identified on the Common Criteria Portal that lists all certified products.

1.6.3 Supporting Hardware, software and/or firmware

Minimum System Requirements	
Appliance	
Software	qCrypt-xStream R1.1
Hardware	Appliance qCrypt-xStream R1.1 – Part number 1362
	Quantum Random Number Generator (QRNG) card - Part number 1108
Web UI User	
Web Browser	Internet Explorer 11 Firefox 30 Chrome 35

1.7 TOE description

1.7.1 Physical scope of the TOE

The TOE is a self-contained appliance (Figure 1) consisting of two main components, namely the QuintessenceLabs Key Manager (QKM) and the Quantum Random Number Generator (QRNG), as a single product in the form of an appliance.

Figure 1: qCrypt xStream

The QuintessenceLabs Key Manager (QKM) is a software based component that is managed via a web management interface. The QRNG is a hardware based component (Figure 2) with an optics core for laser processing, which serves as a quantum based entropy source and is able to produce true random data at a rate of 1Gb/s. The TOE has four (4) Ethernet ports which can be individually associated to one or more of the following networks: management, replication or client.

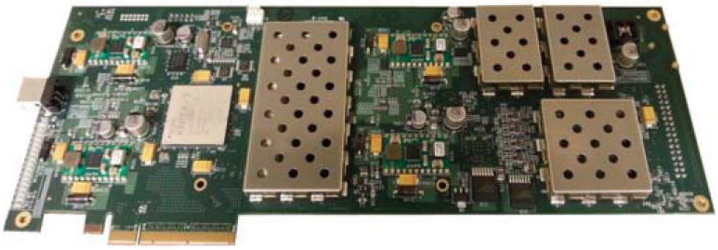

Figure 2: Quantum Random Number Generator PCIe card.

Internally, the TOE is comprised of a number of different components that combine to deliver the core security functionality and capabilities of the device. Figure 3 below illustrates the high level functionality on the scope of the TOE.

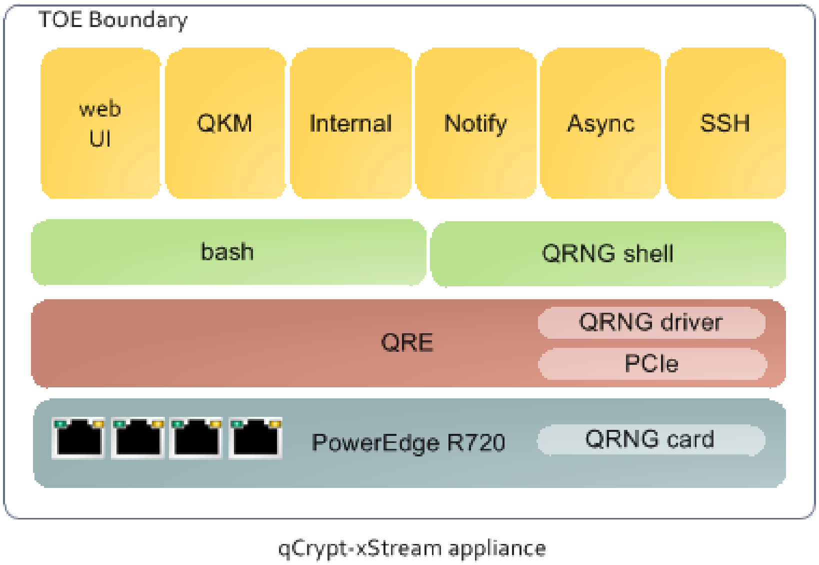

Figure 3 – TOE components

The key components of the TOE are described in the following table.

Table 2 – TOE components

Component	Description
Web user interface (Web UI)	Provides the main mechanism for administering the TOE.
QuintessenceLabs Key Manager (QKM)	Externally-accessible KMIP server

Component	Description
Internal	Supports the Web UI (only accessible via local sockets).
Notify	Provides server-generated notifications to clients
Async	Supports the QKM by handling asynchronous KMIP operations.
SSH	Allows remote login to the TOE for administrative purposes.
Bash shell	Allows remote access to the TOE for general administration of the device.
QRNG shell	Allows remote access to the TOE for configuration and maintenance of the QRNG.
QuintessenceLabs run-time environment (QRE)	QRE is a purpose-built distribution of the Linux operating system

1.7.2 Logical scope of the TOE

Table below provides a brief overview of each of the security functions provided by the TOE that are included within the scope of this evaluation.

Table 3 – TOE security function overview

Security function	Description
Access control	The TOE implements administrative roles that are used for segmenting access control. Each role has pre-defined access to certain functions.
Audit	The TOE logs significant events to an internal audit log with at minimum a timestamp.
Cryptographic operations	The TOE implements several cryptographic algorithms in hardware and software. These algorithms are used internally by the TOE and are also provided to users by the QuintessenceLabs Key Manager (QKM). The TOE implements asymmetric and symmetric encryption algorithms, key generation algorithms, signing, cryptographic checksum and random number generation algorithms.
Data protection	The TOE implements mechanisms to secure TOE data when it is both at rest and when it is exported from the TOE.
Secure key management	The TOE provides the means to generate and manage cryptographic keys as part of the KMIP service offered to clients, as well as for use with its various cryptographic functions.
Security management	The TOE implements a set of functions and mechanisms to securely manage the TSF and TSF data.
Self-test	The TOE implements a set of self-test that verifies the TOE's cryptographic algorithms and random number generator (QRNG).

Security function	Description
User authentication	The TOE provides a mechanism for secure authentication.

2 Conformance Claim (ASE_CCL.1)

The ST and TOE are conformant to version 3.1 (REV 4) of the Common Criteria for Information Technology Security Evaluation.

The following conformance claims are made for the TOE and ST:

- **Part 2 extended.** Conformant with Common Criteria for Information Technology Security Evaluation Part 2: Security functional requirements, version 3.1 (REV 4), September 2012 with extended declaration of the FCS_RNG_EXT component.
- **Part 3 conformant, EAL2.** Conformant with Common Criteria for Information Technology Security Evaluation Part 3: Security assurance requirements, version 3.1 (REV 4). Evaluation is EAL2, September 2012.

3 Security problem definition (ASE_SPD.1)

3.1 Overview

This section describes the nature of the security problem that the TOE is designed to address. The security problem is described through:

a) a set of ***threats*** that the TOE has been designed to mitigate,

b) specific ***assumptions*** about the security aspects of the environment (both IT related and non-IT related elements) in which the TOE will operate, and

c) relevant ***organisational security policies*** statements that are made in terms of rules or guidelines that must be followed by the TOE and/or the operational environment.

3.2 Threats

In the context of this ST, the TOE has the following threat agents:

a) Individuals that have not been granted access to the application who attempt to gain access to information or functions provided by the TOE. This threat agent is considered an ***unauthorised individual***.

b) Individuals that are registered and have been explicitly granted access to the application who may attempt to access information or functions that they are not permitted to access. This threat agent is considered an ***authorised user***.

The following table lists the identified security threats relevant to the TOE.

Table 4 - Threats

Identifier	Threat statement
T.AUDREC	Persons may not be accountable for the actions that they conduct because the audit records are not reviewed or created, thus allowing an attacker to modify the behaviour of TSF data without being detected.
T.COMMSEC	An unauthorised user may attempt to bypass the security mechanism and compromise the integrity and confidentiality of the data collected, processed and transmitted between a) TOE and KMIP client (user data), b) TOEs (for backup and replication), and c) TOE and remote administrators via SSH or the Web UI

Identifier	Threat statement
T.MALFUNC	Internal malfunction of TOE functions may result in the modification or misuse of TOE services. This includes hardware failures which prevent the TOE from performing its services. Technical failure may result in an insecure operational state violating the integrity and availability of the TOE services. The correct operation of the TOE also depends on the correct operation of critical hardware (QRNG) and software (QKM) components. Critical services include: a) cryptographic operations; and b) random number generation
T.UNAUTHORISED	A user may gain unauthorized access to the TOE and residing data.
T.TUSAGE	The TOE may be inadvertently configured, used and administered in an insecure manner by either authorized or unauthorized persons.

3.3 Organisational security policies

No organisational security policies have been defined regarding the use of the TOE.

3.4 Assumptions

The following assumptions provide the foundation for security objectives for the operational environment for the TOE.

Table 5 - Assumptions

Identifier	Assumption statement
A.ADMIN	The Administrator is not careless, wilfully negligent, or hostile, and will follow and abide by the instructions provided by administrator documentation.
A.CLIENT	It is assumed that KMIP clients protect the keys and other security sensitive data that are used to communicate with the TOE.
A.UPDATE	The underlying platform on which the TOE operates will be updated when needed with the latest security patches and fixes to ensure data stored on the platform remains protected and secure.

4 Security objectives (ASE_OBJ.2)

4.1 Overview

The security objectives are a concise statement of the intended response to the security problem defined in Section 3. They are security objectives for the TOE to address and additional objectives that provide specific direction for the intended environment in which the TOE is to operate.

4.2 Security objectives for the TOE

Table 6 – Security objectives for the TOE

Identifier	Objective statements
O.AUDREC	The TOE must provide a means to record a readable audit trail of security-related events, with accurate dates and times, and a means to search the audit trail based on relevant attributes.
O.CRYPT	The TOE implements cryptographic functions compliant to the relevant industry standards.
O.COMMSEC	The TOE shall utilize cryptographic functions and industry standards (such as SSL and TLS) to ensure that data transmitted between the TOEs and KMIP client is secure and protected from tampering or modification.
O.CONTROL	The TOE shall restrict TOE security and management functions to authorised roles.
O.PROTECT	The TOE shall ensure that all TSF data stored within the TOE are protected sufficiently to prevent their disclosure to a malicious entity.
O.TEST	The TOE shall perform tests to verify that its components operate correctly. This includes testing of TOE's random number generator and cryptographic module during operation.

4.3 Security objectives for the environment

Table 7 – Security objectives for the environment

Identifier	Objective statements
OE.ADMIN	The administrators assigned to oversee the TOE are trusted by the organisation and are trained in the use of the TOE.
OE.CLIENT	The KMIP client must protect the key and must not disclose their password to any other user.

Identifier	Objective statements
OE.GUIDAN	The TOE must be delivered, installed, administered, and operated in a manner that maintains security.
OE.UPDATE	The developer shall provide updates or new release of the TOE as needed.

4.4 Security objectives rationale

The following table demonstrates that all security objectives for the TOE trace back to the threats in the security problem definition.

Table 8 – Security objectives rationale

Threats	Objectives	Rationale
T.AUDREC	O.AUDREC	T.AUDREC concerns that the persons may not be accountable for the actions that they conduct because the audit records are not reviewed or created, thus allowing an attacker to modify the behaviour of TSF data without being detected. O.AUDREC which ensures The TOE provides a means to record a readable audit trail of security-related events, with accurate dates and times, and a means to search the audit trail based on relevant attributes
T.COMMSEC	O.CRYPT O.PROTECT O.COMMSEC	T. COMMSEC concerns the integrity of the data collected processed and transmitted between TOE and KMIP client and the risk of this data being modified by an attacker by bypassing the security mechanism. O.CRYPT will ensure that the data collected and transmitted by TOE using cryptographic algorithms is done so in compliance to standards and protected from attacks that attempt to bypass the security mechanisms of TOE. O.PROTECT ensures that TSF data stored within the TOE are protected sufficiently to prevent their disclosure to a malicious entity. O.COMMSEC ensures that data sent between the two TOE components is secure and protected from modification or tampering.
T.MALFUNC	O.TEST	T. MALFUNC concerns that a failure may prohibit the TOE to operate correctly. O.TEST ensures the TOE perform tests to verify that its components operate correctly.

Threats	Objectives	Rationale
T.UNAUTHORISED	O.CONTROL	T.UNAUTHORISED_ACCESS concerns an unknown or unauthorised user gaining access to the TOE, its functions, or its data. O.CONTROL ensures that there is a robust access control system in place to restrict TOE data and management functions to authorised users only.
T.USAGE	OE.ADMIN OE.GUIDAN	T.USAGE concerns that the TOE may be inadvertently configured, used and administered in an insecure manner by either authorized or unauthorized persons OE.ADMIN which ensures the operational environment provides well-trained administrators to appropriately install, configure, and maintain the TOE within its evaluated configuration according to the installation and guidance documents for the TOE. OE.GUIDAN which ensures the operational environment provides a secure manner of TOE delivery, installation, administration, and operation

4.5 Environment security objectives rationale

The following table demonstrates that all security objectives for the operational environment all trace back to assumptions or OSPs in the security problem definition.

Table 9 – Environmental security objectives rationale

Assumptions	Objectives	Rationale
A.ADMIN	OE.ADMIN	This non-IT security objective is necessary to counter the threat T.TUSAGE and support the assumption A.ADMIN because it ensures that authorized administrators receive the proper training in the correct configuration, installation and usage of the TOE.
A.CLIENT	OE.CLIENT	This security objective is necessary to counter the threat T.UNAUTHORISED and support assumption A.CLIENT because it ensures that KMIP clients protect their keys from unknown or unauthorised user gaining access to user data.
A.UPDATE	OE.UPDATE	This security objective is necessary to counter the threat T.UNAUTHORISED and support assumption A.UPDATE because it ensure the TOE is protected with the latest patch or update to prevent disclosure to a malicious entity.

5 Extended components definition (ASE_ECD.1)

To define the IT security functional requirements of the TOE, an additional family (FCS_RNG) of the Class FCS (cryptographic support) is defined here. The FCS_RNG family describes an SFR for random number generation used for cryptographic purposes.

Random Number Generation (FCS_RNG_EXT)

Generation of random numbers requires that the random number generator implements defined security capabilities and the random numbers meet a defined quality metric.

Management: FCS_RNG_EXT.1

There are no management activities foreseen.

Audit: FCS_RNG_EXT.1

There are no actions defined to be auditable.

FCS_RNG_EXT.1	Random number generation
Hierarchical to:	No other components
Dependencies:	No dependencies
FCS_RNG_EXT.1.1	The TSF shall provide a [***selection: physical, non-physical true, deterministic, physical hybrid, deterministic hybrid***] random number generator, which implements: [**assignment: list of security capabilities**].
FCS_RNG_EXT.1.2	The TSF shall provide random numbers that meet [***assignment: a defined quality metric***].

Application Note: A physical random number generator (RNG) produces the random number by a noise source based on physical random processes. A non-physical true RNG uses a noise source based on non-physical random processes like human interaction (e.g. key strokes, mouse movement). A deterministic RNG uses a random seed to produce a pseudorandom output. A hybrid RNG combines the principles of physical and deterministic RNGs. The list of security capabilities may include tests of the internal noise source in case of physical RNG.

6 Security requirements (ASE_REQ.2)

6.1 Overview

This section defines the security requirements satisfied by the TOE. Each requirement has been extracted from version 3.1 (REV 4) of the Common Criteria, part 2.

Part 2 of the Common Criteria defines an approved set of operations that may be applied to security functional requirements. Following are the approved operations and the document conventions used within this ST to depict their application:

- **Assignment.** The assignment operation provides the ability to specify an identified parameter within a requirement. Assignments are depicted using bolded text and are surrounded by square brackets as follows [**assignment**].

- **Selection.** The selection operation allows the specification of one or more items from a list. Selections are depicted using bold italics text and are surrounded by square brackets as follows [***selection***].

- **Refinement.** The refinement operation allows the addition of extra detail to a requirement. Refinements are indicated using bolded text, for **additions**, and strike-through, for ~~deletions~~.

- **Iteration.** The iteration operation allows a component to be used more than once with varying operations. Iterations are depicted by placing a letter at the end of the component identifier as follows FDP_IFF.1a and FDP_IFF.1b.

The security functional requirements are expressed using the notation stated in Section 6.1 above and are itemised in the table below.

Table 10 – Security functional requirements (SFRs)

Identifier	Title
Security audit (FAU)	
FAU_GEN.1	Audit data generation
FAU_SAR.1	Audit Review
FAU_STG.1	Protected audit trail storage
FAU_STG.3	Action in case of possible audit loss
Cryptographic support (FCS)	
FCS_CKM.1	Cryptographic key generation

Identifier	Title
FCS_CKM.2	Cryptographic key distribution
FCS_CKM.4	Cryptographic key destruction
FCS_COP.1	Cryptographic operation
FCS_RNG_EXT.1	Random number generation
User data protection (FDP)	
FDP_ACC.1	Subset access control
FDP_ACF.1	Security attribute based access control
FDP_ETC.1	Export of user data without security attributes
FDP_ITC.1	Import of user data without security attributes
FDP_RIP.1	Subset residual information protection
FDP_UCT.1	Basic data exchange confidentiality
FDP_UIT	Data exchange integrity
Identification and authentication (FIA)	
FIA_ATD.1	User attribute definition
FIA_UAU.2	User authentication before any action
FIA_UAU.7	Protected authentication feedback
FIA_UID.2	User identification before any action
Security management (FMT)	
FMT_REV.1	Revocation
FMT_SMF.1	Specification of management functions
FMT_SMR.1	Security roles
FMT_MSA.3	Static attribute initialisation
FMT_MTD.1	Management of TSF data
FMT_MOF.1	Management of security functions behaviour
Protection of the TOE security functions (FPT)	
FPT_FLS.1	Failure with preservation of secure state
FPT_ITC.1	Inter-TSF confidentiality during transmission
FPT_ITI.1	Inter-TSF detection of modification
FPT_RPL.1	Replay detection
FPT_STM.1	Reliable time stamps
FPT_TST.1	TSF testing

Identifier	Title
TOE Access (FTA)	
FTA_SSL.3	TSF-initiated termination
FTA_SSL4	User-initiated termination
Trusted Channel (FTP)	
FTP_ITC.1.1	Inter-TSF trusted channel
FTP_TRP.1.1	Trusted path

6.2 Security audit (FAU)

6.2.1 FAU_GEN.1 Audit data generation

Hierarchical to:	No other components
Dependencies:	FPT_STM.1 Reliable time stamps
FAU_GEN.1.1	The TSF shall be able to generate an audit record 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) [**Specifically defined auditable events listed below**]
FAU_GEN.1.2	The TSF shall record within each audit record at least the following information: a) Date and time of the 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/ST, [**none**].
Notes:	Auditable events within the TOE are: a) All KMIP Interaction with QKM; b) All use of the user identification and authentication mechanism; c) Modifications to the group of users that are part of a role; d) Backup and duplication status; e) Server connection status (Stopped/started); f) SSL errors; g) Start-up and shutdown of the random number generation service; h) QRNG test result (failure/success).

6.2.2 FAU_SAR.1 Audit Review

Hierarchical to:	No other components

Dependencies:	FAU_GEN.1 Audit data generation
FAU_SAR.1.1	The TSF shall provide [**root, users with the System Management privilege**] with the capability to read [**all audit information**] from the audit records.
FAU_SAR.1.2	The TSF shall provide the audit records in a manner suitable for the user to interpret the information.
Notes:	Web UI users with appropriate privileges are able to export logs.

6.2.3 FAU_STG.1 Protected audit trail storage

Hierarchical to:	No other components
Dependencies:	FAU_GEN.1 Audit data generation
FAU_STG.1.1	The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.
FAU_STG.1.2	The TSF shall be able to [***prevent***] unauthorised modifications to the stored audit records in the audit trail.
Notes:	The TOE prevents unauthorised modification of logs. Only the root user is able to delete logs.

6.2.4 FAU_STG.3 Action in case of possible audit data loss

Hierarchical to:	No other components
Dependencies:	FAU_STG.1 Protected audit trail storage
FAU_STG.3.1	The TSF shall [**overwrite the oldest stored audit records**] if the audit trail exceeds [**User Specified limits**].
Notes:	The TOE implements a log rotation regime to protect against audit storage failure.

6.3 Cryptographic Support (FCS)

6.3.1 FCS_CKM.1 Cryptographic key generation

Hierarchical to:	No other components
Dependencies:	[FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic operation] FCS_CKM.4 Cryptographic key destruction
FCS_CKM.1.1	The TSF shall generate cryptographic keys in accordance with a specified cryptographic key generation algorithm [**the cryptographic generation algorithms specified in Table 11**] and specified cryptographic key sizes [**cryptographic key sizes specified in Table 11**] that meet the following: [**applicable standards as specified in Table 11**]**.**

<table>
<tr><td>Notes:</td><td>
<b>Table 11 – Implemented key generation methods</b>
<table>
<tr><th>Algorithm</th><th>Key size (in bits)</th><th>Standard</th></tr>
<tr><td>AES</td><td>128, 192, 256</td><td>SP 800-133</td></tr>
<tr><td>Triple-DES</td><td>112, 168</td><td>SP 800-133</td></tr>
<tr><td>RSA</td><td>2048</td><td>RSA PKCS#1</td></tr>
</table>
</td></tr>
</table>

6.3.2 FCS_CKM.2 Cryptographic key distribution

<table>
<tr><td>Hierarchical to:</td><td>No other components.</td></tr>
<tr><td>Dependencies:</td><td>[FDP_ITC.1 Import of user data without security attributes, or
FDP_ITC.2 Import of user data with security attributes, or
FCS_CKM.1 Cryptographic key generation]
FCS_CKM.4 Cryptographic key destruction</td></tr>
<tr><td>FCS_CKM.2.1</td><td>The TSF shall distribute cryptographic keys in accordance with a specified cryptographic key distribution method [<b>cryptographic key distribution methods described in Table 12 – Implemented key distribution methods below</b>] that meets the following: [<b>multiple standards described in Table 12 – Implemented key distribution methods</b>].</td></tr>
<tr><td>Notes:</td><td>
<b>Table 12 – Implemented key distribution methods</b>
<table>
<tr><th>Key distribution method</th><th>Standard</th></tr>
<tr><td>Distribution of session keys using TLS</td><td>RFC 5246 (TLS v1.1)
RFC 4346 (TLS v1.2)</td></tr>
<tr><td>Distribution of key pairs for SSL user credentials.</td><td>None.
Export (and import) of SSL credentials is via secure (HTTPS) download (or upload) using the web admin interface</td></tr>
</table>
QKM controlled keys which are accessed via the KMIP are considered user data (not cryptographic keys) for the purpose of this SFR.
</td></tr>
</table>

6.3.3 FCS_CKM.4 Cryptographic key destruction

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]

FCS_CKM.4.1	The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [**key zeroisation**] that meets the following: [**none**].
Notes:	None

6.3.4 FCS_COP.1 Cryptographic operation

Hierarchical to:	No other components.
Dependencies:	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction
FCS_COP.1.1	The TSF shall perform [**cryptographic operations specified in** Error! Reference source not found. **below**] in accordance with a specified cryptographic algorithm [**cryptographic algorithm specified in** Error! Reference source not found. **below**] and cryptographic key sizes [**cryptographic key sizes specified in** Error! Reference source not found. **below**] that meet the following: [**standards specified in** Error! Reference source not found. **below**].
Notes:	**Table 13 – Cryptographic functions**

Operation	Algorithm and mode	Key size (in bits)	Applicable standard
Encryption and decryption	AES (CBC/GCM)	128 192 256	FIPS 197
	TDES	112 168	SP 800-67
Hashing	SHA-1 SHA-256 SHA-384	160 256 384	FIPS 180-2
Keyed-hash message authentication	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-384	160 256 384	FIPS 198-1
Digital Signatures	RSA	2048	RSA PKCS#1
	DSA	2048	FIPS 186-4
	ECDSA	256 384 521	FIPS 186-4

6.3.5 FCS_RNG_EXT.1 Random number generation

Hierarchical to:	No other components.
Dependencies:	No dependencies
FCS_RNG_EXT.1.1	The TSF shall provide a [***physical***] random number generator, which implements: [**online test of the raw random number sequence**].
FCS_RNG_EXT.1.2	The TSF shall provide random numbers that meet [**full entropy**].
Notes:	A physical random number generator (RNG) produces the random number by a noise source based on physical random processes. A non-physical true RNG uses a noise source based on non-physical random processes like human interaction (e.g. key strokes, mouse movement). A deterministic RNG uses a random seed to produce a pseudorandom output. A hybrid RNG combines the principles of physical and deterministic RNGs. The list of security capabilities may include tests of the internal noise source in case of physical RNG.

6.4 User Data Protection (FDP)

6.4.1 FDP_ACC.1 Subset Access control

Hierarchical to:	No other components.
FDP_ACC.1.1	The TSF shall enforce the [**access control SFP**] on [• **Subjects:** ○ **Root** ○ **Users** • **Objects:** ○ **Secret objects (symmetric keys, private keys, split keys, secret data and KMIP opaque objects)** ○ **Public keys** ○ **Public and private KMIP templates** • **Operations:** ○ **Locate** ○ **Check** ○ **Get** ○ **Get Attributes** ○ **Get Attribute List** ○ **Add/Modify/Delete Attribute** ○ **Obtain Lease** ○ **Activate** ○ **Revoke** ○ **Destroy**

	○ **Re-key** ○ **Re-key key pair** ○ **Derive key** ○ **Get usage allocation**].
Dependencies:	FDP_ACF.1 Security attribute based access control
Notes:	A full verbose list of the CRUD operations available can be found in the KMIP standard (version 1.1).

6.4.2 FDP_ACF.1 Security attribute based access control

Hierarchical to:	No other components.
Dependencies:	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation
FDP_ACF.1.1	The TSF shall enforce the [**access control SFP**] to objects based on the following: [**assigned user privileges**].
FDP_ACF.1.2	The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed: [***Users are permitted access to objects if they have the Key Management privilege assigned to their accounts***].
FDP_ACF.1.3	The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: [**Root users can access all objects**].
FDP_ACF.1.4	The TSF shall explicitly deny access of subjects to objects based on the following additional rules: [**Users have no privileges attached to their accounts**].
Notes:	None.

6.4.3 FDP_ETC.1 Export of user data without security attributes

Hierarchical to:	No other components.
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]
FDP_ETC.1.1	The TSF shall enforce the [**access control SFP**] when exporting user data, controlled under the SFP(s), outside of the TOE.
FDP_ETC.1.2	The TSF shall export the user data without the user data's associated security attributes
Notes:	Keys and associated attributes can be exported from the TOE through two mechanisms: a) KMIP 'Get' and 'GetAttributes', which is implicitly controlled via the rules specified by FDP_ACF.1.3; and b) Database duplication/back-up export, which relies on a configured peer authentication data and the TLS protocol.

	Attributes can be uniquely associated to keys via the digest attribute.

6.4.4 FDP_ITC.1 Import of user data without security attributes

Hierarchical to:	No other components.
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FMT_MSA.3 Static attribute initialisation
FDP_ITC.1.1	The TSF shall enforce the [**access control SFP**] when importing user data, controlled under the SFP, from outside of the TOE.
FDP_ITC.1.2	The TSF shall ignore any security attributes associated with the user data when imported from outside the TOE.
FDP_ITC.1.3	The TSF shall enforce the following rules when importing user data controlled under the SFP from outside the TOE: [**TLS connection shall be established before importing user data**].
Notes:	Users are able to import keys created outside the TOE via KMIP Register operation. Users are authenticated via TLS.

6.4.5 FDP_RIP.1 Subset residual information protection

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FDP_RIP.1.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the [***deallocation of the resource from***] the following objects: [**user data**].
Notes:	None.

6.4.6 FDP_UCT.1 Basic data exchange confidentiality

Hierarchical to:	No other components.
Dependencies:	[FTP_ITC.1 Inter-TSF trusted channel, or FTP_TRP.1 Trusted path] [FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]
FDP_UCT.1.1	The TSF shall enforce ~~the~~ [**access control SFP**] to [***transmit and receive***] user data in a manner protected from unauthorised disclosure.
Notes:	Defines confidentiality requirements for data transferred between the TOE and another trusted IT product. In this case, • the confidentiality of the database duplication process is protected by TLS; and • the confidentiality of the KMIP messages is protected using TLS

6.4.7 FDP_UIT.1 Data exchange integrity

Hierarchical to:	No other components.
Dependencies:	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] [FTP_ITC.1 Inter-TSF trusted channel, or FTP_TRP.1 Trusted path]
FDP_UIT.1.1	The TSF shall enforce the [**access control SFP**] to [***transmit, receive***] user data in a manner protected from [***modification, deletion, insertion, replay***] errors.
FDP_UIT.1.2	The TSF shall be able to determine on receipt of user data, whether **[*modification, deletion, insertion, replay*]** has occurred.
Notes:	The integrity of the database duplication process is protected by TLS; The confidentiality of the KMIP messages is protected using TLS.

6.5 Identification and authentication (FIA)

6.5.1 FIA_ATD.1 User attribute definition

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FIA_ATD.1.1	The TSF shall maintain the following list of security attributes belonging to individual users: [**Web UI username and password, SSH username and password, KMIP client certificate and asymmetric key pairs**].
Notes:	None.

6.5.2 FIA_UAU.2 User authentication before any action

Hierarchical to:	FIA_UAU.1 Timing of authentication
Dependencies:	FIA_UID.1 Timing of identification
FIA_UAU.2.1	The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.
Notes:	Before any action, users are authenticated either by a password (administrators and root user) or using TLS public-key based credentials in addition to proxy passwords, if applicable.

6.5.3 FIA_UAU.7 Protected authentication feedback

Hierarchical to:	No other components.
Dependencies:	FIA_UAU.1 Timing of authentication
FIA_UAU.7.1	The TSF shall provide only [**obscured feedback**] to the user while the authentication is in progress.

Notes:	None.

6.5.4 FIA_UID.2 User identification before any action

Hierarchical to:	FIA_UID.1 Timing of identification
Dependencies:	No dependencies.
FIA_UID.2.1	The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.
Notes:	User authentication is enforced before any action performed by the TOE on behalf of the user. Authentication is password based for administrators and public-key based using TLS for clients.

6.6 Security management (FMT)

6.6.1 FMT_REV.1 Revocation

Hierarchical to:	No other components.
Dependencies:	FMT_SMR.1 Security roles
FMT_REV.1.1	The TSF shall restrict the ability to revoke [**security attributes**] associated with the [***User***] under the control of the TSF to [**Root**].
FMT_REV.1.2	The TSF shall enforce the rules [**authorised users shall be able to revoke security-relevant authorizations by completely deleting user security attributes, or by modifying the user name, group, or by setting a new Password. Such revocation is to take effect when the user next authenticates to the Web UI**].
Notes:	None.

6.6.2 FMT_SMF.1 Specification of management functions

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FMT_SMF.1.1	The TSF shall be capable of performing the following management functions:[• **Configuration of authentication parameters;** • **System back-up/duplication;** • **User/group creation;** • **Policy setting;** • **User certificate and key creation;** • **Network configuration;** • **Power off/reboot;** • **Set time;** • **Log rotation; and** • **Configuration of remote log storage**

	].
Notes:	None.

6.6.3 FMT_SMR.1 Security roles

Hierarchical to:	No other components.
Dependencies:	FIA_UID.1 Timing of identification
FMT_SMR.1.1	The TSF shall maintain the roles **[User, Root].**
FMT_SMR.1.2	The TSF shall be able to associate users with roles.
Notes:	None.

6.6.4 FMT_MSA.3 Static attribute initialization

Hierarchical to:	No other components.
Dependencies:	FMT_MSA.1 Management of security attributes FMT_SMR.1 Security roles
FMT_MSA.3.1	The TSF shall enforce the [**access control SFP**] to provide [***restrictive***] default values for security attributes that are used to enforce the SFP.
FMT_MSA.3.2	The TSF shall allow the [**users with the privileges defined in Table 14, root**] to specify alternative initial values to override the default values when an object or information is created.
Notes:	None

6.6.5 FMT_MTD.1 Management of TSF data

Hierarchical to:	No other components.
Dependencies:	FMT_SMR.1 Security roles FMT_SMF.1 Specification of Management Functions
FMT_MTD.1.1	The TSF shall restrict the ability to [**query, modify, delete,** [***view, create***]] the [• ***User data (username, password, assigned roles/privileges, session timeout)*** • ***Role data (role name, description, assigned users, assigned privileges)*** • ***Policy data (policy name, group name, assigned operations)*** • ***Key data (name, private key password, public key password, algorithm, permitted uses, operational policy)*** • ***Template data (name, algorithm, usage limit, usage policy, permitted uses)*** • ***Certificate data (name, password, signing password, country code, state, locality, OU, common name, email, validity)*** • ***Certificate authority data (name, CA certificate, private key)***] to [***root, users with privileges defined in Table 14***].

<table>
<tr><td>Notes:</td><td><b>Table 14 – Required user privileges</b>
<table>
<tr><th>Data type</th><th>Required privilege set</th></tr>
<tr><td>User data</td><td>Account Management</td></tr>
<tr><td>Role data</td><td>Account Management</td></tr>
<tr><td>Policy data</td><td>Client Management</td></tr>
<tr><td>Key data</td><td>Key Management</td></tr>
<tr><td>Template data</td><td>Key Management</td></tr>
<tr><td>Certificate data</td><td>System Management</td></tr>
<tr><td>Certificate authority data</td><td>System Management</td></tr>
</table>
</td></tr>
</table>

6.6.6 FMT_MOF.1 Management of security functions behaviour

Hierarchical to:	No other components.
Dependencies:	FMT_SMR.1 Security roles FMT_SMF.1 Specification of Management Functions
FMT_MOF.1.1	The TSF shall restrict the ability to [**disable, enable, modify the behaviour of**] the functions [***Key Management (KM) service, NTP service, log rotation service***] to [***root, users with the System Management privilege***].
Notes:	None

6.7 Protection of the TOE security functions (FPT)

6.7.1 FPT_FLS.1 Failure with preservation of secure state

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_FLS.1.1	The TSF shall preserve a secure state when the following types of failures occur: [**any failure resulting from power-on self-tests or error state that requires manual intervention**].
Notes:	The TOE implements the following self-test mechanisms: The QRNG provides a set of power-on known-answer tests, which upon failure force the QRNG into an error state that requires manual intervention.

6.7.2 FPT_ITC.1 Inter-TSF confidentiality during transmission

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_ITC.1.1	The TSF shall protect all TSF data transmitted from the TSF to another trusted IT product from unauthorised disclosure during transmission.

Notes:	None

6.7.3 FPT_ITI.1 Inter-TSF detection of modification

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_ITI.1.1	The TSF shall provide the capability to detect modification of all TSF data during transmission between the TSF and another trusted IT product within the following metric: [**a TLS integrity error**]
FPT_ITI.1.2	The TSF shall provide the capability to verify the integrity of all TSF data transmitted between the TSF and another trusted IT product and perform [**retransmission in case of a protocol error or session termination in case of malicious tampering**] if modifications are detected.
Notes:	In this case, the security relevant TOE data that is exported consists of logs and user account information, which are transferred via TLS.

6.7.4 FPT_RPL.1 Replay detection

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_RPL.1.1	The TSF shall detect replay for the following entities: **[SSH communications segments, TLS communications segments]**.
FPT_RPL.1.2	The TSF shall perform **[drop segments]** when replay is detected.
Notes:	None.

6.7.5 FPT_STM.1 Reliable time stamps

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_STM.1.1	The TSF shall be able to provide reliable time stamps.
Notes:	The TOE supports time synchronisation with time servers using NTP.

6.7.6 FPT_TST.1 TSF testing

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FPT_TST.1.1	The TSF shall run a suite of self tests **[*during initial start-up, at the request of the authorised user*]** to demonstrate the correct operation of **[*the TSF*].**
FPT_TST.1.2	The TSF shall provide authorised users with the capability to verify the integrity of [***TSF data***].
FPT_TST.1.3	The TSF shall provide authorised users with the capability to verify the integrity of

	[***TSF***].
Notes:	None

6.8 TOE Access (FTA)

6.8.1 FTA_SSL.3 TSF-initiated termination

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FTA_SSL.3.1	The TSF shall terminate an interactive session after a [**a logout or a specified time interval of user inactivity set by a user with account management privileges. The default session timeout is 300 seconds**].
Notes:	When a user has not been active for a period of time the session is terminated and subsequent attempts to perform an action or navigate away from the current page automatically redirects to the login page.

6.8.2 FTA_SSL.4 User-initiated termination

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FTA_SSL.4.1	The TSF shall allow user-initiated termination of the user's own interactive session.
Notes:	None.

6.9 Trusted Channel (FTP)

6.9.1 FTP_ITC.1 Inter-TSF trusted channel

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FTP_ITC.1.1	The TSF shall provide a communication channel between itself and another trusted IT product 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.
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 **[database replication].**
Notes:	TLS channels are implemented for communications to other TOES (for database replication) and to clients.

6.9.2 FTP_TRP.1 Trusted path

Hierarchical to:	No other components.
Dependencies:	No dependencies.
FTP_TRP.1.1	The TSF shall provide a communication path between itself and [***remote***] users that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from [***modification, disclosure***]**.**
FTP_TRP.1.2	The TSF shall permit **[*the TSF, local users, remote users*]** to initiate communication via the trusted path.
FTP_TRP.1.3	The TSF shall require the use of the trusted path for **[*initial user authentication and all further actions performed via the interface*]**.
Notes:	Communications between users and the Web UI are secured using TLS. Remote login access to the TOE is protected using SSH.

6.10 Dependency rationale

The table below demonstrates the mutual supportiveness of the SFRs for the TOE by demonstrating how the SFR dependencies are fulfilled by the TOE and by justifying those dependencies that are not fulfilled.

The SARs relevant to the TOE constitute an evaluation assurance level of EAL2, as defined in the Common Criteria and include no extensions or augmentations. Therefore, as a complete evaluation assurance level, they are a mutually supportive set and require no further justification.

Table 15 – SFR dependency rationale

SFR	Dependency	Rationale
Security audit (FAU)		
FAU_GEN.1	FPT_STM.1 Reliable time stamps	Included
FAU_SAR.1	FAU_GEN.1 Audit data generation	Included
FAU_STG.1	FAU_GEN.1 Audit data generation	Included
FAU_STG.3	FAU_STG.1 Protected audit trail storage	Included
Security audit (FAU)		
FCS_CKM.1	[FCS_CKM.2 Cryptographic key distribution, or FCS_COP.1 Cryptographic operation] FCS_CKM.4 Cryptographic key destruction	Included

SFR	Dependency	Rationale
FCS_CKM.2	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction	Included
FCS_CKM.4	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]	Included
FCS_COP.1	[FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation] FCS_CKM.4 Cryptographic key destruction	Included
FCS_RNG_EXT.1	No dependencies	N/A
FDP (User data protection)		
FDP_ACC.1	FDP_ACF.1 Security attribute based access control	Included
FDP_ACF.1	FDP_ACC.1 Subset access control FMT_MSA.3 Static attribute initialisation	Included
FDP_ETC.1	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]	Included
FDP_ITC.1	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] FMT_MSA.3 Static attribute initialisation	Included
FDP_RIP.1	No dependencies	N/A
FDP_UCT.1	[FTP_ITC.1 Inter-TSF trusted channel, or FTP_TRP.1 Trusted path] [FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]	Included
FDP_UIT	[FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control] [FTP_ITC.1 Inter-TSF trusted channel, or FTP_TRP.1 Trusted path]	Included
FIA (Identification and authentication)		
FIA_ATD.1	No dependencies	N/A
FIA_UAU.2	FIA_UID.1 Timing of identification	Included
FIA_UAU.7	FIA_UAU.1 Timing of authentication	Included
FIA_UID.2	No dependencies	N/A

SFR	Dependency	Rationale
FMT (Security management)		
FMT_REV.1	FMT_SMR.1 Security roles	Included
FMT_SMF.1	No dependencies	N/A
FMT_SMR.1	FIA_UID.1 Timing of identification	Included
FMT_MSA.3	FMT_MSA.1 Management of security attributes FMT_SMR.1 Security roles	Included
FMT_MTD.1	FMT_SMR.1 Security roles FMT_SMF.1 Specification of Management Functions	Included
FMT_MOF.1	FMT_SMR.1 Security roles FMT_SMF.1 Specification of Management Functions	Included
FPT (Protection of the TSF)		
FPT_FLS.1	No dependencies	N/A
FPT_ITC.1	No dependencies	N/A
FPT_ITI.1	No dependencies	N/A
FPT_RPL.1	No dependencies	N/A
FPT_STM.1	No dependencies	N/A
FPT_TST.1	No dependencies	N/A
FTA (TOE access)		
FTA_SSL.3	No dependencies	N/A
FTA_SSL.4	No dependencies	N/A
FTP (Trusted path/channels)		
FTP_ITC.1.1	No dependencies	N/A
FTP_TRP.1.1	No dependencies	N/A

6.11 Mapping of SFRs to security objectives for the TOE

Table 16 – Mapping of SFRs to TOE security objectives

Security objective	Mapped SFRs	Rationale
O.AUDREC	FAU_GEN.1	The TOE allows set of rules to be applied to indicate authorised and unauthorised access of every user.
	FAU_STG.1	The requirement helps meet the objective by protecting audit trail storage.
	FAU_STG.3	The requirement helps meet the objective by performing log rotation regime to protect against audit storage failure.
	FAU_SAR.1	The TOE maintain a profile of system usage and suspicion rating to each profile along with threshold condition to indicate possible security violation.
O.CRYPT	FCS_CKM.1	Generate cryptographic keys in accordance with a specified cryptographic key generation algorithm.
	FCS_CKM.2	Distribute cryptographic keys in accordance with a specified cryptographic key distribution algorithm.
	FCS_CKM.4	The requirement helps to meet the objective by destroying cryptographic keys in accordance with a specified cryptographic key destruction method.
	FCS_COP.1	Perform cryptographic operation in accordance with a specified cryptographic algorithm.
	FCS_RNG_EXT .1	Perform quantum random number generation for use in key generation.
O.COMMSEC	FPT_ITC.1	The requirement helps to meet the objective by protecting imported data from modification or disclosure.
	FTP_ITC.1	The requirement helps meet the objective by providing TLS channels for communications to other TOES (for database duplication) and to clients.
	FTP_TRP.1	The requirement helps meet the objective by providing a secured communication path between users and the Web UI using TLS and remote login using SSH.
O.CONTROL	FTA_SSL.3	The requirement helps meet the objective by terminating an interactive session after user inactivity for a period of interval time.
	FTA_SSL.4	The requirement helps meet the objective by allowing user-initiated termination of the user's own interactive session.
	FMT_REV.1	The requirement helps meet the objective by restricting the ability to revoke security attributes.

Security objective	Mapped SFRs	Rationale
	FMT_MSA.3	The requirement helps meet the objective by restricting access to provide default values for security attributes that are used to enforce the SFP.
	FMT_SMR.1	The requirement helps meet the objective by defining the security roles used within the TOE.
	FDP_UCT.1	The requirement helps meet the objective by protecting the confidentiality of data exchange between TOEs and KMIP client using TLS.
	FDP_ETC.1	The requirement helps meet the objective by protecting export user data without security attributes.
	FDP_ITC.1	The requirement helps meet the objective by protecting import user data without security attributes.
	FDP_UIT.1	The requirement helps meet the objective by protecting data exchange integrity using TLS.
	FPT_STM.1	The requirement helps meet the objective by providing reliable time stamps.
	FMT_MOF.1	The requirement helps meet the objective by restricting the configuration of the management functions.
	FMT_MTD.1	The requirement helps meet the objective by restricting the ability to modify the TSF data.
	FDP_ACC.1	The requirement helps meet the objective by providing an access control functionality to ensure that access to security functionality is controlled.
	FPT_ITI.1	The requirement provides inter-TSF detection of modification that is exported consists of logs and user account information, which are transferred via TLS.
	FDP_ACF.1	The requirement provides access control functionality to ensure that access to security functionality is controlled.
O.PROTECT	FIA_UID.2	The requirement helps meet the objective by identifying user before any TSF mediated actions.
	FIA_UAU.2	The requirement helps meet the objective by authenticating user before any TSF mediated actions.
	FIA_UAU.7	The requirement helps meet the objective by providing only Obfuscated feedback to the user while the authentication is in progress.
	FIA_ATD.1	The requirement helps meet the objective by ensuring user security attributes are maintained.

Security objective	Mapped SFRs	Rationale
	FDP_RIP.1	The requirement helps meet the objective by ensuring that any previous information within a resource is made unavailable before any TSF mediated actions can be performed.
	FMT_SMF.1	The requirement helps meet the objective by providing management functions of the TOE for authenticated user.
	FPT_RPL.1	The requirement helps meet the objective by performing drop segments when replay is detected.
	FMT_SMR.1	The requirement helps meet the objective by providing user timing of identification.
O.TEST	FPT_TST.1	The requirement helps meet the objective by performing self-test functions.
	FPT_FLS.1.1	The requirement helps meet the objective by preserving a secure state upon failure of self-tests.

7 Security requirements rationale

7.1 TOE Security assurance requirements

This section defines the security assurance requirements satisfied by the TOE. Each requirement has been extracted from version 3.1 (REV 4) of the Common Criteria, part 3. The assurance components are summarized in the following table

Table 17 – Security assurance requirements

Assurance class	Assurance components
ADV: Development	ADV_ARC.1 Security architecture description
	ADV_FSP.2 Security-enforcing functional specification
	ADV_TDS.1 Basic design
AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
ALC: Life cycle support	ALC_CMS.2 Parts of the TOE CM coverage
	ALC_CMC.2 Use of a CM system
	ALC_DEL.1 Delivery procedures
ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST Introduction
	ASE_OBJ.2 Security objectives
	ASE_REQ.2 Derived security requirements
	ASE_SPD.1 Security Problem Definition
	ASE_TSS.1 TOE summary specification
ATE: Tests	ATE_IND.2 Independent testing - sample
	ATE_FUN.1 Functional testing
	ATE_COV.1 Evidence of coverage
AVA: Vulnerability assessment	AVA_VAN.2 Vulnerability analysis

8 TOE summary specification (ASE_TSS.1)

8.1 Overview

This section provides the TOE summary specification, a high-level description of how the TOE implements the claimed security functional requirements.

The TOE implements the following security functions:

- Access control
- Audit
- Cryptographic operations
- Data protection
- Secure key management
- Security management
- Self-test
- User authentication

8.2 Access Control

The TOE implements a role based access control to ensure that only users authenticated with their credentials are permitted to perform allocated functions. The TOE implements the following roles that are used to restrict access to the TOE's management tasks (FMT_SMR.1):

- **Users**. Web UI users can be granted access to differing levels of functionality, depending on the set of privileges allocated to their role; and
- **Root**. A root user role is defined within the TOE to carry out administrative tasks on the underlying OS (QRE) via SSH.

Users cannot access TOE data or functionality unless they a) have an account to authenticate with the TOE and b) have the relevant privilege for each function assigned to their account. Only the root role may access remotely via SSH to perform functions provided by the TOE.

As demonstrated, the ***access control*** function implements the following SFRs: **FMT_SMR.1, *FDP_ACC.1 and FDP_ACF.1***

8.3 Audit

Event data is captured in the audit log. Each audit log entry includes an event code, a time/date stamp, a number indicating the event type, and any other relevant parameters. The TOE will generate audit records for the following events:

- All KMIP Interaction with QKM;
- All use of the user identification and authentication mechanism;
- Modifications to the group of users that are part of a role;
- Backup and duplication status;
- Server connection status (Stopped/started);
- SSL errors;
- Start-up and shutdown of the random number generation service;
- QRNG test result (failure/success).

Users with the “System Management” privilege (or root users) may view the audit records stored on the TOE. The TOE implements protections to prevent users from modifying audit data once it has been generated and stored.

If the audit store is full, the TOE will automatically delete the oldest stored audit records to ensure that new audit logs are retained.

Timestamps are generated for audit logs by utilising the underlying operating system.

As demonstrated, the ***audit*** function implements the following SFRs: **FAU_GEN.1, FAU_SAR.1, FAU_STG.1, FAU_STG.3** and **FPT_STM.1**.

8.4 Cryptographic operations

The TOE implements the following key generation methods:

Table 18 – Implemented key generation methods

Algorithm	Key size (in bits)	Standard
AES	128, 192, 256	SP 800-133
Triple-DES	112, 168	SP 800-133
RSA	2048	RSA PKCS#1

Key generation of keys *managed* by the TOE is accomplished by:

- Constructing AES keys by taking random bits directly from the QRNG up to the required key length.
- Generating all other keys via the OpenSSL library, which is configured to take its random input from the QRNG

Key generation of keys *used* by the TOE is accomplished by:

- TLS session keys for communicating with the TOE via KMIP are generated by OpenSSL, which is configured to take its random input from the QRNG
- SSH keys are generated and used by the OpenSSH daemon, version 5.9p1.
- TLS session keys that secure web UI sessions are generated by the Apache server, version 2.4.3
- Generation of client authentication certificates and signing of same is performed by PHP code which calls into OpenSSL to perform the cryptographic functions.

AES keys are used by the TOE for SSH connections between itself and users connecting to the TOE via the administrative interface or users connecting directly to the QRNG card to perform diagnostics. AES is also used for TLS connections between the TOE and users accessing the TOE via the web interface.

The Triple-DES and RSA keys generated by the TOE are used as part of the certificate generation and signing process.

The TOE implements key distribution methods in accordance with RFCs 5246 (TLS v1.1) and 4346 (TLS v1.2) for TLS keys. SSL keys are manually imported/exported by administrative users via the WebUI interface.

The TOE includes a hardware based entropy source designed to meet the requirements of NIST Draft Special Publication 800-90B (SP 800-90B). The entropy is derived from measuring a quantum state, specifically: the quadraturesof quantum vacuum states of light, which are measured using a balanced optical homodyne detector. The measurements are digitised and post-processed using a CBC-MAC-AES-128 conditioning function as per SP 800-90B. The resulting entropy source construction (SP 800-90B) – the QRNG – is a true random bit generator providing full-entropy.

The raw entropy data coming from the noise source is tested continuously for any abnormalities. The following health tests specified in SP800-90B are implemented in the hardware: repetition count test and adaptive proportion test. These tests are performed on the entire raw output sequence (i.e. prior to conditioning). On detection of a health test failure, the QRNG ceases outputting data.

In addition to the continuous health test, the QRNG performs power-on and on demand self-testing. When the QRNG starts, a known-answer test is executed against the conditioning function. Similarly, the health tests themselves are tested when the QRNG starts by running them against predefined sequences and checking the health test output against expected results.

The TOE implements the following cryptographic functions:

Table 19 – Cryptographic functions

Operation	Algorithm and mode	Key size (in bits)	Applicable standard
Encryption and decryption	AES (CBC/GCM)	128 192 256	FIPS 197
	TDES	112 168	SP 800-67
Hashing	SHA-1 SHA-256 SHA-384	160 256 384	FIPS 180-2
Keyed-hash message authentication	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-384	160 256 384	FIPS 198-1
Digital Signatures	RSA	2048	RSA PKCS#1
	DSA	2048	FIPS 186-4
	ECDSA	256 384 521	FIPS 186-4

All of the methods listed above are leveraged from the OpenSSL cryptographic library, version 1.0.1i.

The TOE will zeroise cryptographic keys and other related material when they are no longer required.

As demonstrated, the ***cryptographic operations*** function implements the following SFRs: ***FCS_CKM.1, FCS_CKM.2, FCS_CKM.4, FCS_COP.1*** and ***FCS_RNG_EXT.1.***

8.5 Data protection

The TOE ensures that no identifiable/sensitive information is imported to or exported from the TOE and requires the use of a TLS tunnel to prevent the transmitted data from interception or modification.

The TOE verifies the integrity of incoming TLS traffic to determine whether the traffic has been subject to unauthorised modification, deletion, insertion or has been sent as part of a potential replay attack.

Once data is no longer required, the TOE de-allocates the information to render it unusable by any other TOE function.

The TOE permits the following actions (both on the TOE and via the KMIP protocol) to be performed on public keys, public and private KMIP templates and secret objects (symmetric keys, private keys, split keys, secret data and KMIP opaque objects).

- Locate
- Check
- Get
- Get Attributes
- Get Attribute List
- Add/Modify/Delete Attribute
- Obtain Lease
- Activate
- Revoke
- Destroy
- Re-key
- Re-key key pair
- Derive key
- Get usage allocation

As demonstrated, the data protection function implements the following SFRs: **FDP_ACC.1, FDP_ACF.1, FDP_ITC.1, FDP_ETC.1, FDP_RIP.1, FDP_UCT.1** and **FDP_UIT.1**.

8.6 Secure key management

The TOE provides a range of key management functions that are available to authenticated users. Authorized roles are capable of generating both symmetric and asymmetric keys, securely destroying keys and other cryptographic data when it is no longer required and the import and export of cryptographic keys.

The confidentiality of the keys is managed throughout their lifecycle, including when being exported.

As demonstrated, the secure key management function implements the following SFRs: **FCS_CKM.2, FCS_CKM.4, FPT_ITC.1 and FPT_ITI.1.**

8.7 Security Management

The TOE contains various management functions to ensure efficient and secure management of the TOE (FMT_SMF.1).

Users with the appropriate permissions (User Management privileges) can modify the access control list and mapping of users to roles. The TOE provides the following suite of management functions (FMT_MOF.1):

- Configuration of authentication parameters (including session timeout limits);
- System backup/duplication;
- User/group management;
- Policy configuration;
- Certificate and key management;
- Network configuration;
- Time/NTP configuration;
- Log rotation and configuration of remote log storage; and
- System power off/reboot

Users with the “User Management” privilege (or root users) may assign and adjust the functions available to users; users may assign and adjust the functions based on organization’s requirement(s).

The TOE ensures that the residual data and any other sensitive data are sufficiently cleared when no longer needed in accordance with the following policies:

- PostgresSQL zeroisation policy;
- QKM code zeroisation of memory on allocation; and
- Key and cryptographic sensitive parameter zeroisation by OpenSSL as per FIPS 140-2.

As demonstrated, the ***security management*** function implements the following SFRs*:* ***FMT_SMF.1, FMT_MSA.3, FMT_SMR.1, FMT_MTD.1, FMT_REV.1, FTA_SSL.3, FTA_SSL.4, FPT_RPL.1, FPT_STM.1, FTP_ITC.1*** and ***FTP_TRP.1.***

8.8 Self-test

Self-tests are conducted on security relevant functions of the TOE each time TOE is powered on or reset. Self-tests are carried out on QRNG and the OpenSSL cryptographic module provide a set of self-testing functions.

As demonstrated, the ***self-test*** function implements the following SFRs: ***FPT_TST.1*** and ***FPT_FLS.1***

8.9 User Authentication

a) KMIP Authentication

KMIP packets contain a header that carries additional information that can be used for identification of the user. This will allows the KMIP client to act as an agent on behalf of other users. This information can come in one of two formats:

- A username and an optional password. These credentials are checked against a database in the QKM: the username must exist in the database, and if the database entry contains a password, the client-supplied password must match it.
- A set of device credentials - possible credentials are:
  - a) Device serial number
  - b) Password
  - c) Device Identifier
  - d) Network Identifier
  - e) Machine Identifier
  - f) Media Identifier

These credentials are checked against a database in the QKM.

b) WebUI Authentication

When a user issues a request to the TOE to access a protected resource (methods or HTML pages), the TOE requires that the user identify and authenticate themselves before performing any TSF mediated action. The TOE compares the credentials by checking the information presented by the user at the Web UI login page against the authentication information stored in the TOE.

Obfuscated feedback is provided while user authentication is in progress.

All user presented passwords are hashed before being used to authenticate to the TOE, or when users change their passwords to be written to the database. This is all done by the TOE.

c) SSH Authentication

SSH is used for command-line connections to the appliance. These connections are secured by password-based authentication, authentication by RSA keypair is not provided out of the box.

SSH connections are established for two purposes:

- To perform administrative functions not possible through the web interface - these will only be performed by qualified service personnel.
- To manage the qStream QRNG card or to perform detailed diagnostics.

The encryption algorithms employed by SSH are set by client negotiation during session establishment.

As demonstrated, the ***user authentication*** function implements the following SFRs: ***FIA_UID.2, FIA_UAU.2, FIA_ATD.1, FIA_UAU.7, FMT_MTD.1, and FCS_COP.1.***