ASI-HSM AHX5 kNET Cryptographic Module Security Target

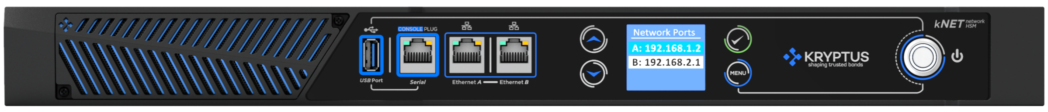


Table of Contents

1 Revision History

Table 1.1: Revision History

Version	Date (YY-MM-DD)	Creator/Modifier	Reviewed by	Description
Up to 1.3.0	2023-08-09	Sinara Pamplona	Leonardo Cerqueira	Initial version, basic structures
1.4.0	2023-08-30	Sinara Pamplona	Leonardo Cerqueira	First official version

1.5.0	2024-03-04	Leonardo Cerqueira	Sinara Pamplona	Modified version after first ROA.
				Added:
				Subsection Licensing System;
				Subsection Secure Execution of Applications;
				Edited:
				Subsection Multi User Capabilities;
				Table Cryptographic module interfaces.;
				Subsection Self-Tests;
				Subsection Operator management;
				Subsection Data storage;
				Subsection Quorum authentication;
				Subsection Audit;
				Section TOE Overview;
				Subsection Security Features;
				Subsection TOE Delivery;
				Chapter Security Requirements;
				Table Key Generation Table;
				Subsection FCS_CKM.4 Cryptographic key destruction;
				Table Cryptographic Operations Table;
				Subsection FIA_AFL.1 Authentication failure handling;
				Subsection FDP_IFF.1/KeyBasics Simple security attributes;
				Subsection FTP_TRP.1/Local Trusted Path;
				Subsection FTP_TRP.1/External Trusted Path;
				Subsection FPT_STM.1 Reliable time stamps;
				Subsection FPT_TST_EXT.1 Basic TSF Self Testing;
				Subsection FMT_SMR.1 Security roles;
				Subsection FMT_SMF.1 Security management functions;
				Subsection FMT_MTD.1/AuditLog Management of TSF data;
				Subsection FAU_STG.2 Guarantees of audit data availability;
				Table TOE Summary Specification: Briefing Table;
				Table TOE Summary Specification: Mapping

1.6.0	2024-06-13	Leonardo Cerqueira	Sinara Pamplona	Modified version after second ROA.
				Added:
				Glossary;
				Edited:
				Cover Image;
				Section Document Organization;
				Section Identification;
				Subsection TOE Delivery;
				Introduction of Chapter Security Requirements;
				Subsection Physical Scope;
				Table Key Generation Table;
				Subsection FCS_CKM.4 Cryptographic key destruction;
				Subsection FCS_COP.1 Cryptographic operation;
				Subsection FAU_GEN.1 Audit data generation;

1.6.1	2024-09-06	Leonardo Cerqueira	Sinara Pamplona	Modified version after third ROA.
				Edited:
				Subsection Self-Tests;
				Section Document Organization;
				Section Identification;
				Subsection TOE Delivery;
				Introduction of Chapter Security Objectives;
				Subsection FCS_CKM.1 Cryptographic key generation;
				Subsection FCS_CKM.4 Cryptographic key destruction;
				Table Cryptographic Operations Table;
				Application Note 14;
				Table TOE Summary Specification: Mapping;

2 Glossary

Item	**Description**
HSM	Hardware Security Module
PHSM	Physical HSM A physical instance of an HSM
VHSM	Virtual HSM A logical instance of an HSM, running inside a physical HSM
Operator	Operator A generic term for a user of the TOE, regardless of their category
PCO	Physical Crypto Officer An administrator category with privileged access to the PHSM, responsible for managing the entire device
VCO	Virtual Crypto Officer An administrator category responsible for managing a single VHSM and the Key Users within it
Key User	A category of operator responsible for the management and use of cryptographic objects. In our public documents (References folder) it is referred to as just User, but in the Common Criteria documentation we decided to call it Key User to avoid ambiguity

3 Introduction

3.1 Document Organization

This document is part of Common Criteria Documentation [2] [3] and defines a specific Target of Evaluation (TOE), ASI-HSM AHX5 kNET Cryptographic Module. The Security Target (ST) provides the TOE overview and a set of security objectives and requirements. Furthermore, based on the aspects of the environment, the ST defines a list of threats the device intends to counter and the security functions provided by the TOE.

Section 1, Revision History contains a table detailing the history of the document's revisions.

Section 2, Glossary contains the definitions of some of the terms used in the document.

Section 3, Introduction provides the TOE identification, the TOE overview and the TOE description.

Section 4, Conformance Claims provides the conformance claims.

Section 5, Security Problem Definition presents the assets, subjects, threats, organisational security policies and assumptions related to the TOE.

Section 6, Security Objectives defines the security objectives for both the TOE and the TOE environment.

Section 7, Extended Components Definition defines the extended components.

Section 8, Security Requirements contains the functional requirements and specifies the assurance requirements that must be satisfied by the TOE and the TOE environment.

Section 9, Rationales presents all the Security Objectives Rationales and the Security Requirements Rationales.

Section 10, TOE Summary Specification describes the security functions that are included in the TOE.

Section 11, References contains the listing of all references.

3.2 Identification

Document Identifier:	ASI-HSM AHX5 kNET Cryptographic Module Security Target
Document Version:	1.6.1
TOE Name:	ASI-HSM AHX5 kNET Cryptographic Module
TOE Hardware Version:	1.0.1
TOE Firmware Version:	1.1.0
TOE Software Version:	1.48.1
Created by:	Kryptus S.A.
Publication Date:	13th of September 2024

KeyWords: Cryptographic module, General Purpose HSM, Hardware security module, Virtual HSM, Digital signature, Random number generation, ECDSA, RSA, AES, SHA, HMAC, Hash, Encryption, Decryption, Key generation, Message digest generation, Key management, Object management, Tamper protection, KMIP, PKCS#11, OpenSSL Engine, Java JCA Provider

3.3 TOE Overview

The ASI-HSM AHX5 kNET Cryptographic Module is a Hardware Security Module that provides cryptographic services with native implementation of the Key Management Interoperability Protocol [15]. This protocol defines the structure and execution of cryptographic operations, such as generation of asymmetric key pairs, issuing of certificates, generation of random data and others. The communication between the user and the module itself occurs through standard Ethernet interface using TCP protocols.

A fragment of its functional capabilities are the following:

- Symmetric encryption with AES;
- Asymmetric encryption with RSA, ECIES;
- Message authentication codes with HMAC, CBC-MAC;
- Digital signatures with RSA, DSA, ECDSA;
- Generate Random Data;
- Export Objects;
- Import Objects;
- Quorum Authentication (M of N);
- Backup;
- Multiple Virtual HSMs;
- Clustering of multiple HSMs.

Although its requests must abide to KMIP specifications, the module comes along with libraries for the mainstream application programming languages:

- kkmip-cpp: C++ library;
- kkmip-py: Python library;
- kkmip-java: Java library;
- kkmip-js: JavaScript library.

Also, the module is provided with the most common cryptographic APIs for prompt integration with already implemented systems, which are the following:

- PKCS#11;
- OpenSSL Engine;
- Java JCA Provider.

By having multi-tenant potential, the module is able to reproduce its capabilities into independent Virtual HSMs (VHSM), where its administrator doesn't have permission to interfere in another VHSM's management, this allows varied cryptographic service roles using the same equipment.

3.3.1 Security Features

- **Key Security**: The TOE is a cryptographic module that executes cryptographic operations with appropriate parameters in conformance with SOG-IS Agreement [6], which includes but are not restricted to: generation and management of AES Symmetric keys and of Asymmetric keys with RSA and ECDSA algorithms. All of these operations use cryptographic objects, which keys are part of, contained within the TOE and only require a safe environment that provides a power supply and User access to the TOE through a Ethernet connection.
- **Operators Roles and Authentication**: All TOE's operators are stored and maintained in isolation, each one with a different scope: PCOs manage the entire product (physical environment); VCOs oversee an independent virtual environment called VHSM (isolated from each other) and the Users create objects and execute cryptographic operations. All these TOE's operators can authenticate by username and password, OTP, certificate, token or session. Additionally, by activating the Quorum Authentication (M of N), it is possible to protect chosen objects by only allowing the execution of Critical Operations on them with the agreement of a defined group of Users.
- **Data Protection**: The TOE protects the keys against logical attacks that would result in disclosure, compromise and unauthorised modification, and for ensuring that the TOE's services are available only if authentication is verified. Also the TOE provides physical mechanisms to detect

and protect data against physical attacks, such as an opaque epoxy resin applied over components and sensors, power input and environment monitoring.

- **External Key Usage**: The TOE is capable of storing external keys and using them internally, or momentarily using them without storing.
- **Backup and Recovery**: The TOE provides two types of backup, each one related to the correspondent environment (PHSM or VHSM). When creating a backup, you can specify any number of target HSMs, with a minimum of one, by supplying their device certificates and, after authentication has been confirmed, it can be stored in removable media or downloaded through the network to the operator's device. In case of failure, the same options are supported by the restore operation.
- **Communication Protection**: The TOE provides a trusted communication path between external client applications and itself, using TLS over TCP protocol or pure TLS.
- **Audit**: The TOE generates audit records of all processes, such as: start-up, shutdown, key generation and destruction, import and export keys and others.
- **Secure Execution of Applications**: The TOE allows the execution of Python scripts internally, extending its inherent security features to the applications executing those scripts and providing further security by encapsulating them into their own independent environments, guaranteeing their execution and safety.

3.4 TOE Description

The TOE is the board containing the entire ASI-HSM AHX5 kNET Cryptographic Module, enclosed by the red rectangle in Figure 3.2. The ASI-HSM AHX5 kNET Cryptographic Module (Figure 3.1) is a multi-user, multi-chip embedded crypto-module, it is referred to in the remainder of this document as the module. It is embedded within a stand-alone network appliance, which is a Non-TOE part, that gives easy access to its Ethernet, RS232 and Frontal Board interfaces and includes a dual power supply. That appliance is typically used in large-scale cloud infrastructures, where ease of remote configuration and operation is required.

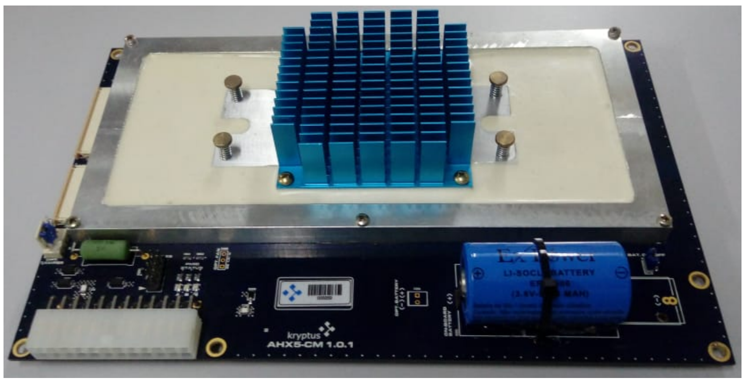


Figure 3.1: ASI-HSM AHX5 kNET Cryptographic Module.

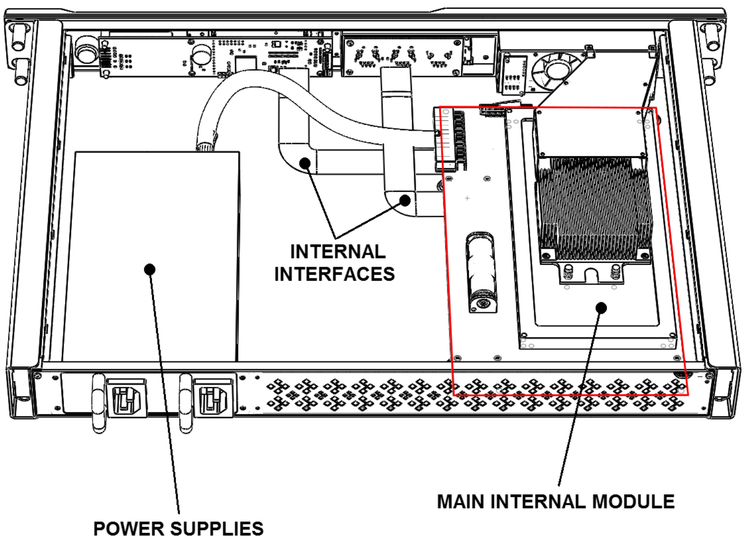


Figure 3.2: Architecture of ASI-HSM AHX5 kNET

The module exists to provide cryptographic services to applications running on behalf of its Users who communicate with it via a standard Ethernet interface using IP protocols. In order to provide these services, the module also requires a power supply.

3.4.1 TOE Components

The component that manages the cryptographic services and system's assurances, called Software, and the one which calls the low hardware-level cryptographic capabilities, called Firmware, are both responsible for enforcing all SFRs and SARs. All other parts of the TOE, which are contained within the boundary shown in red in Figure 3.2, support but do not interfere with enforcement e.g., OS, DB, logging system, ethernet interface driver and CPU.

3.4.2 Multi User Capabilities

The module is divided into two logical parts: physical and virtual. The physical part is named Physical HSM (PHSM) with its administrator Physical Crypto Officer (PCO), its virtual counterpart is the Virtual HSM (VHSM) with its administrator Virtual Crypto Officer (VCO). The module provides cryptographic services only from VHSMs logical security modules and they are created and managed by the PHSM.

The administrators do not have access to the TOE's cryptographic services, those are solely reserved to the Key User role and only this role can manage and use cryptographic objects. The Key Users are created and managed by the VCO in their respective VHSM. Multiple VHSMs can be created in the PHSM by the PCO, and each VHSM has its own Key Users and data, which cannot be accessed by other VHSMs, nor by the PHSM.

For more details on PCOs and VCOs capabilities, see sections 3.4.5.3 and 3.4.5.8

3.4.3 Authentication

The module provides management and cryptographic operations through the KMIP protocol [15], a network based communication protocol. KMIP specifies that compliant Servers and Clients shall establish and maintain channel confidentiality and integrity. Therefore, every message, including any accompanying authentication data, is encrypted using the TLS protocol.

The module enforces identity-based authentication and each identity is mapped to a single role. The module supports the following authentication schemes:

- **Password-based authentication**: username and password. The password is composed of 6 or more alphanumeric characters, which may include both upper and lower case letters, punctuation marks, and symbols (such as @, &, and *).

- **Certificate-based authentication**: private key and certificate. Both are used to enable client authentication according to the TLS protocol, in which a handshake message is digitally signed using the private key and the signature is sent to the module.
- **OTP-based authentication**: one time password. Both time-based and counter-based one time passwords may be used, but this authentication can only be used as a second authentication factor.
- **Token-based authentication**: username and signed challenge. The signed challenge is generated by signing a random 100-byte string with a previously associated token. The random string is requested to the module and valid for a single use.
- **Session-based authentication**: session token. The session token is composed of 20 alphanumeric characters, which may include both upper and lower case letters, punctuation marks, and symbols (such as @, &, and *).

The PCOs and VCOs of a physical and logical module may configure the authentication policy, i.e. which authentication schemes must be used. Those roles may also configure the password policy, making it as restrict as necessary.

The module also implements a mechanism of Quorum authentication, used to restrict access to the module only when multiple identities agree to operate the module. See section 3.4.5.6 for further details.

3.4.4 Physical Scope

The module has two main protection mechanisms: an opaque epoxy resin is applied over its PCB components (see 3.4.4.1), to prevent physical access to the components of the module, and environmental sensors, to prevent attacks and tampering (see 3.4.4.2).

3.4.4.1 Physical Security Mechanisms

To prevent physical access to the components of the module, an opaque epoxy resin is applied over its PCB components. The epoxy coating completely conceals the internal components of the cryptographic boundary. Any attempt to physically access the components leads to the destruction of the module. An aluminum frame is used to delimit the resin coating over the PCB, which can be seen on Figure 3.3. Figure 3.4 shows how the resin is spread over the PCB of the module.

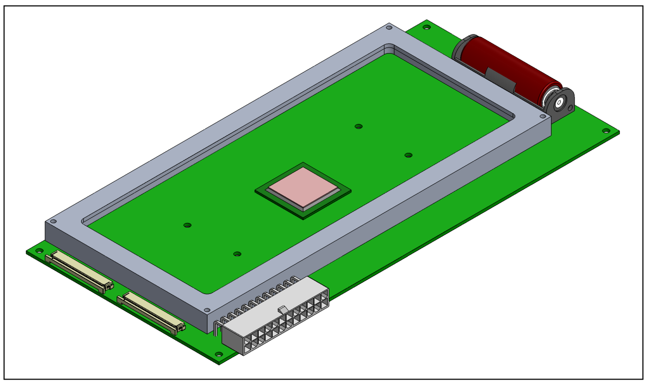


Figure 3.3: Module board with aluminum potting frame.

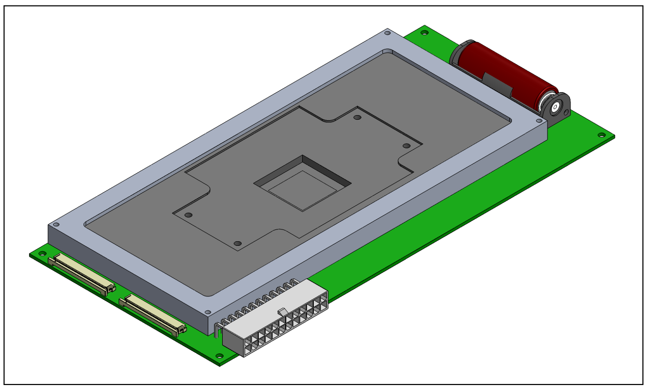


Figure 3.4: Epoxy resin potting of the module.

To allow heat dissipation of the main processor, an aluminum heatsink adapter is placed on top of it and is locked on the resin. This adapter can be seen within the complete module assembly on 3.5 and alone on Figure 3.6.

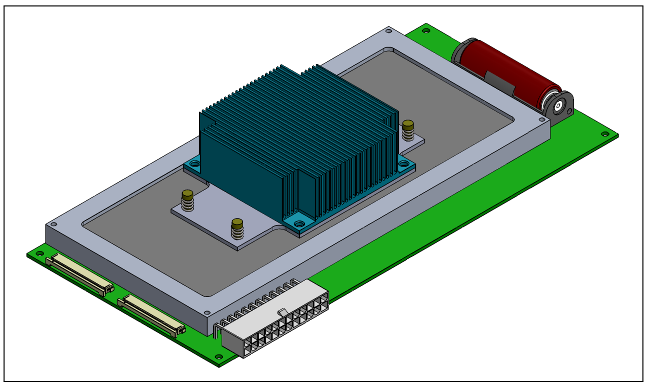


Figure 3.5: Complete assembly of the module.

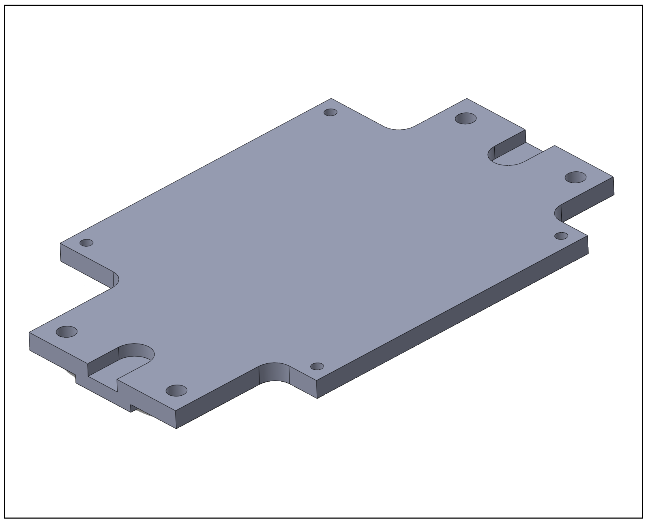


Figure 3.6: Aluminum heatsink adapter.

3.4.4.2 Physical Sensoring Mechanisms

Besides the use of the epoxy potting, the module monitors environmental parameters to prevent some attacks.

An ultra-low-power microcontroller is used to control, sample and analyze the data from the sensors around the PCB. It is also capable of, optionally, holding critical system parameters for the CPU, as it is able to exchange data with it through an internal bus. It is also connected to the real time clock and a dedicated Random Number Generator. A non-removable battery, along with voltage conditioning and a supercapacitor, is used to ensure that all the sensors and the sensors monitor have non-interruptible power at all times.

If the module determines that an attempt has been made to compromise its physical security, it reboots and initializes in an error state, effectively erasing all its sensitive data from memory. To remove the module from this error state, either the Reset HSM or the Remove from Error State services must be used.

3.4.4.3 Hardware Appliance Boundary and Interfaces

All the hardware components that will execute security functions are contained in the Cryptographic Boundary, which is completely covered by the epoxy resin potting explained on the Physical Security Mechanisms section. Figure 3.7 illustrates the module's cryptographic boundary and the external interfaces. The external interfaces are connected to all Non-TOE elements contained in the chassis: dual power supply, RS232, dual Ethernet connections, USB port and the interactive Frontal Board.

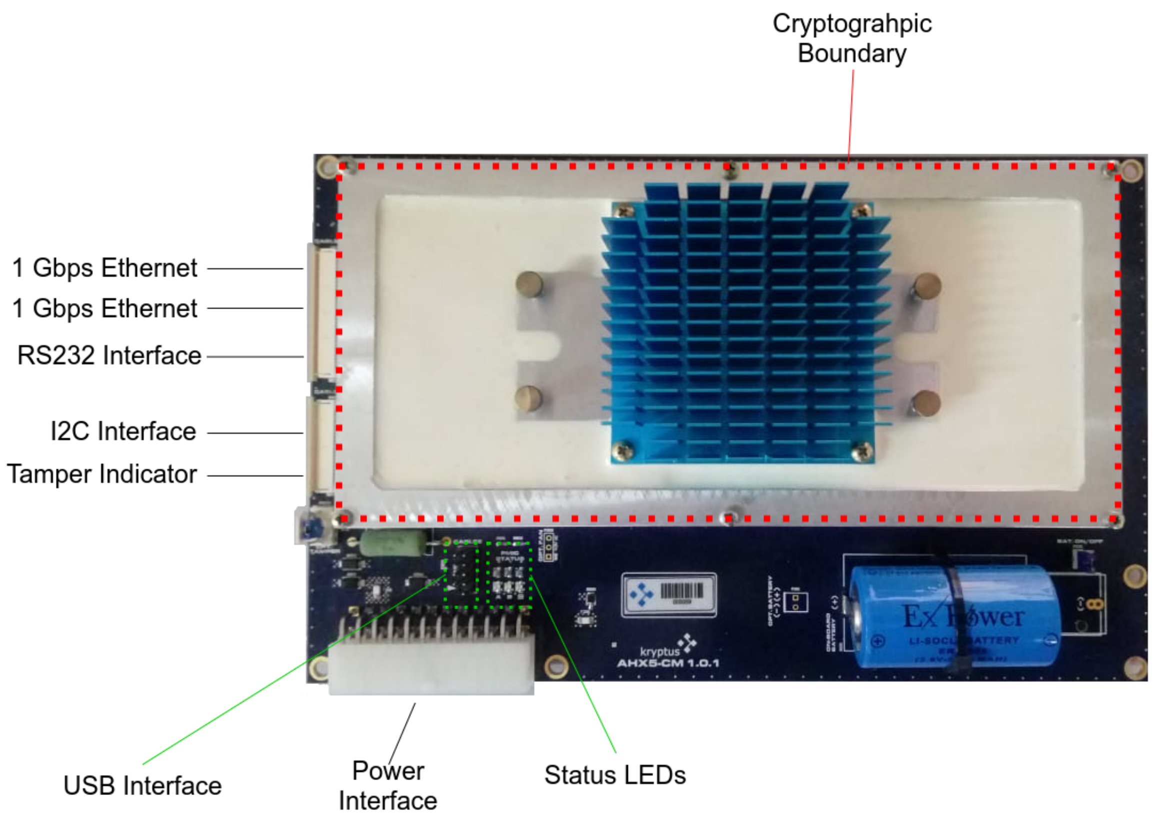


Figure 3.7: Cryptographic Boundary and Interfaces.

Table 3.1 lists the module's external connection interfaces.

Table 3.1: Cryptographic module interfaces.

Interface	**Description**	**Logical Interface**
Power Interface	Powers the module.	Power
Status LEDs	Set of pins used to output status.	Status Output
USB Interface	Optionally used to export and import backups. Since backups are encrypted, the sensitive backup information is protected.	Data Input and Output
Tamper Indicator	Logical part of the I2C interface, indicates if tampering has been detected (either by the voltage or temperature monitoring)	Status Output
I2C Interface	Used to connect to external devices for module control (e.g. turn off the module). No sensitive information is transmitted through this channel. Also used to output module status to the frontal board.	Control Input, Status Output
RS232 Interface	Serial interface used for module control (e.g. configure network interface). No sensitive information is transmitted through this channel. Also used to output module status.	Control Input, Status Output

Table 3.1: Cryptographic module interfaces (continued).

1 Gbps Ethernet Interfaces	Two sets of pins for Ethernet connection. These interfaces are the main communication channel for the cryptographic module. Sensitive data is always encrypted when transmitted through these pins.	Control Input, Data Input and Output

When the module is initialized, it outputs the temporary PIN of the crypto officer through the RS232 and I2C interfaces. However, at this moment the module does not contain sensitive user data and once the crypto officer's password is changed, it cannot be obtained via those interfaces anymore.

3.4.5 Logical Scope

The module's protection mechanisms from a logical scope intent to prevent unauthorized access, ensure and verify that the module is working correctly.

To prevent unauthorized access, the module:

- Implements Secure boot (see 3.4.5.1), to ensure that the module will only execute trusted firmware.
- Store sensitive data encrypted/obfuscated (see 3.4.5.4) at all times.
- Restrict access to objects (see 3.4.5.5).
- Restrict usage of objects (see 3.4.5.7).
- Quorum authentication (see 3.4.5.6).

To ensure and verify that the module is working correctly, the module run self-tests (see 3.4.5.2) and allows auditing its operations through logs (see 3.4.5.8).

3.4.5.1 Secure boot

Secure boot is a verification mechanism for ensuring that the firmware is trusted. This mechanism verifies each binary loaded at boot against known keys. If the verification fails for any of the required binaries, the boot process halts and puts the module into an error state, protecting the module against unauthorized software execution.

The module's secure boot mechanism requires four RSA Public Keys (with at least 2048 bits), which are entered into a one-time-programmable memory on the module during manufacturing.

3.4.5.2 Self-Tests

The module performs self-tests according to FIPS140-2 [25]. Although this version of FIPS has been superseded in 2019 by FIPS-140-3, the TOE is only certified in the older version.

The tests are divided into power-on self-tests and conditional self-tests, explained below. If any of power-on self-tests fail, the module will not initialize and will enter an error state. Similarly, if any of the conditional self-tests fail, every virtual module become blocked and the module will enter an error state.

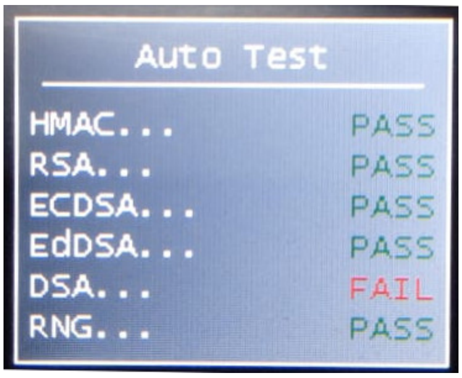
The power-on self-test are executed when the module initializes, with no operator intervention. The power-on self tests are listed in Table 3.2.

Table 3.2: Power-on Self Tests

Test	Description
AES	Encryption and decryption in CBC, ECB* and CTR modes. Key size: 128 bits.
AES GCM	Encryption and decryption. Key size: 128 bits.
SHA	SHA-1 and SHA-2 message digest generation with SHA-1*, SHA-224, SHA-256, SHA-384, SHA-512.
HMAC	Generation and verification with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512.
RSA	Key Pair Generation, Digital Signature Generation and Digital Signature Verification. Key size: 2048 bit.

ECDSA	Key Pair Generation, Digital Signature Generation and Digital Signature Verification. Curve: NIST P-521.
EdDSA*	Key Pair Generation, Digital Signature Generation and Digital Signature Verification. Curve: Ed521.
DSA	Key Pair Generation, Digital Signature Generation and Digital Signature Verification. Key size: 256 bits.
DRBG	Known answer test and health tests (instantiate, generate and reseed).
KAS	Key Agreement Scheme (Primitive "Z" computation using ECDH).
TLS 1.2 KDF	SP800-135 Rev 1 TLS 1.2 KDF.
Firmware Integrity Test	Digital Signature Verification with RSA 2048 bits and SHA-256.

[1] When the power-on self-tests are run, the frontal board shows the results of each test, as illustrated in Figure 3.8. The same results can be viewed after boot through the frontal board menu.



```
Auto Test

HMAC...       PASS
RSA...        PASS
ECDSA...      PASS
EdDSA...      PASS
DSA...        FAIL
RNG...        PASS
```

Figure 3.8: Power-on self-test results in frontal board.

The module performs conditional self-tests during its operation, in response to some events. The self tests are listed in Table 3.3.

[1]*When in CC mode, only Power-On Self-Tests for the Approved cryptographic algorithms are executed.

Table 3.3: Conditional Self Tests

Test	Description
Pair-wise Consistency Tests	RSA, ECDSA and DSA Key Pair Generation
Continuous RNG test	Continuous DRBG test
Firmware Load Test	Digital Signature Verification with RSA 2048 bits and SHA-256

3.4.5.3 Operator management

As mentioned earlier in section 3.4.2, the module supports three roles:

- Physical Crypto Officer (PCO): manages the module at the physical scope (Physical HSM, PHSM) creates logical modules (VHSMs) and doesn't have access to cryptographic operations.
- Virtual Crypto Officer (VCO): manages the logical module (Virtual HSM, VHSM), although there could be many VHSMs the VCO only exists in and interacts with their VHSM and also doesn't have access to cryptographic operations.
- User: Only exists within a VHSM and is created by the VCO, is able to manage and use cryptographic objects and has access to cryptographic operations through the Ethernet interface. They are also capable of creating Secure Applications to execute operations on their behalf.

After the first boot, while the module is in factory state, a default PCO must be used to initialize the PHSM. At this moment, the PCO must specify any policy that restricts the module's operation. After that, the PCO will be able to create, delete and list VHSMs, export and import backups, create new PCOs, export the module log, among other operations.

Similarly, after a new VHSM is created, the first VCO must be used to initialize the VHSM. After that, the VCO will be able to create, delete and list other VCOs and Key Users, and export the VHSM's log, among other operations. The effects of deleting Key Users that have keys will be explained in section 3.4.5.5.

Different from Users, who may own cryptographic objects, PCOs and VCOs own the entire PHSM and VHSM, respectively. Therefore, any PCO is able to delete any other PCO, and similarly any VCO is able to delete any other VCO, but strictly on their own VHSM as they can't interact with VHSMs other than their own. However, PCOs and VCOs never interact with each other.

Roles are created alongside a temporary password in the module, which may only be used to change the role's password. After that, the role will be able to execute any operation permitted to its type.

All roles may communicate with the module using KMIP through the Ethernet interface without requiring any driver installation or local clients, although applications may implement such protocol in order to create interactive platforms or services.

3.4.5.4 Data storage

The module stores data in a persistent local storage. Sensitive data, such as user keys and password, is stored obfuscated and can only be unobfuscated in the module it's stored. Sensitive data is never stored unobfuscated in persistent storage, and only keys may be temporarily unobfuscated, in some situations when being loaded for use.

Individual keys may be exported from the module, either in plaintext or wrapped by another key. However, the PCO may set a policy to force sensitive keys to never leave the module as plaintext. It is also possible to specify during key creation that the key is sensitive or not exportable. Non-exportable keys can only be moved between HSMs using the backup mechanism or clustering.

Backups can be exported either for a single virtual module or for the module as a whole. In both cases, the target of the backup must be specified by its device certificate, a Digital Certificate generated during manufacturing that identifies the module. The module may export a single backup that targets multiple other modules in a single operation. It's worth noting that when a VHSM is imported, it starts out in the "Pending Administrator Approval" state, and a VCO must approve it to transition it to the "Operational" state and start using it. Therefore, any and all backups require dual authentication to be approved, which satisfies the FDP_ACF.1.2/Backup security requirement.

A module may replicate the storage from another if it joins a Cluster with the latter. All modules contained in a Cluster share the same storage data, while maintaining all policies, allowing provision of cryptographic operations by all member modules.

The following list summarizes all key types and the correspondent storage.

- Disk: PHSM Module Certificate, Kryptus kNET CA Certificate, PHSM/VHSM Server CA Certificate, PHSM/VHSM Server Certificate, PHSM/VHSM Client CA Certificate, PCOs'/VCOs'/Users' Obfuscated Passwords and PCOs'/VCOs'/Users' Certificate Fingerprints;
- Obfuscated in Disk: PHSM Module Key, PHSM/VHSM Server CA Key, PHSM/VHSM Server Key, PHSM/VHSM Client CA Key, PCO's/VCO's/User's OTP Key and Users' Objects;
- Volatile Memory: PCO's/VCO's/User's First Password;
- Temporarily in secure processor memory: Job Descriptor Key Encryption Key (generated internally on boot, encrypts keys in memory while they are being used and it never exits the TOE);
- Plaintext in secure processor register: ECDH Private and Public Components, TLS Keys (Peer Public Key, Pre-Master Secret, Master Secret, Session and Authentication keys) and DRBG C and V values;
- One-time programmable memory: Super Root Key Hash (entered during manufacturing and it never exits).

3.4.5.5 Object access restriction

The module defines two permissions to associate objects with Users: Owner and Usage. Users with Owner permission to an object may delete the object and add or remove permissions from other Users to the object. Users with Usage permission may only use the object. A User may have both permissions, only one permission or no permission to objects, and only Users with any permission to an object may list it.

Objects may only be created by the User role, and only on VHSMs. Initially, the User that created the object will have Owner and Usage permissions over the object. When a User is deleted, its permissions will be removed from every object. If that causes an object to lose its last Owner, the object will be deleted.

3.4.5.6 Quorum authentication

In addition to the authentication mechanisms described in Section 3.4.3, the module provides an additional layer of security for Critical Operations. These operations, which involve modifying the system environment (such as creating a VHSM or deleting roles) or managing valuable resources (such as cryptographic objects), require the consensus of a group of operators. Specifically, a minimum number (M) of operators from a designated group (totaling N operators, where M is less than or equal to N) must agree on the execution of these Critical Operations.

In particular, when protecting cryptographic objects, the group must be formed by object Owners, and they must allow operation before any User may use the object for cryptographic operations.

3.4.5.7 Objects usage

When an object is created or registered in the module, additional flags must be supplied to indicate for which operations the object may be used. Some of the available flags are:

- Sign
- Verify
- Encrypt
- Decrypt
- Wrap Key
- Unwrap Key

3.4.5.8 Audit

Only the Administrator roles are able to export the log of operations executed by the module. However, the log is filtered so only entries relevant to the given role may be exported:

- **VCO**: export every operation executed in the VCO's VHSM.
- **PCO**: export every operation executed in the module. A PCO may also request to delete the logs alongside the export operations.

3.4.5.9 Licensing System

The TOE is equipped with a licensing system that allows for the restriction of the number of Virtual HSMs (VHSMs) that can be created within the module, based on the customer's requirements. However, it's important to note that this restriction does not apply when the TOE is operating in Common Criteria mode, as the module will have its full capabilities.

3.4.6 Secure Execution of Applications

The kNET HSM allows Key Users to run Python applications, accessing cryptographic objects stored within the device. These applications, uploaded and managed through custom KMIP requests sent to the kNET HSM KMIP server, which accepts two formats: a single script file or a directory tree filled with scripts and resource files.

Once uploaded, a unique link in the form of a Unique Identifier (UID) is provided for each application, allowing multiple instances to run concurrently on the kNET HSM. These Secure Execution Applications primarily utilize cryptographic operations, connecting to the Virtual HSM through an internal UNIX socket to fulfill their functions effectively.

The security of this functionality is guaranteed by the following factors:

1. The application is executed completely within the Cryptographic boundary of the TOE;
2. To import an application, the signatures of a PCO and a VCO are required by default;

3. The application developer is restricted to use the KMIP API;
4. The integrity of the application is always checked when initialised.

3.5 Delivery

3.5.1 TOE Delivery

- **Hardware**: Kryptus kNET Network Hardware Security Module
  - Unique Identifier: Each device can be uniquely identified by its serial number, located on its back, and the Hardware Version v1.0.1.
- **Firmware**:
  - Yocto Board Support Package v1.48.1;
    - Unique Identifier: Gitlab tag *v1.48.1*
    - Format: System Image
  - PMIC Firmware v1.0.0;
    - Unique Identifier: Gitlab tag *v1.0.0*
    - Format: Binary
  - MONITOR Firmware v1.0.0.
    - Unique Identifier: Gitlab tag *v1.0.0*
    - Format: Binary
- **Software**:
  - **Crypto APIs**:
    - jca-provider v1.3.0: Kryptus' JCA Provider for Java 8;
      - Unique Identifier: Gitlab tag *v1.3.0*
      - Format: JAR file (.jar)
    - jca-provider v2.0.3: Kryptus' JCA Provider for Java 11;
      - Unique Identifier: Gitlab tag *v2.0.3*
      - Format: JAR file (.jar)

- pkcs11 v1.17.0: Kryptus' PKCS#11 Module;
  - Unique Identifier: Gitlab tag *v1.17.0*
  - Format: Dynamic Library, .dll for Windows and .so for Linux
- engine openssl v1.0.16: Custom OpenSSL engine
  - Unique Identifier: Gitlab tag *v1.0.16*
  - Format: Dynamic Library, .dll for Windows and .so for Linux

– **Examples**:
- jca_provider_examples v1.3.0: JCA Provider examples for Java 8;
  - Unique Identifier: Gitlab tag *v1.3.0*
  - Format: ZIP file (.zip) of several JCA source code examples (.java)
- jca_provider_examples v2.0.3: JCA Provider examples for Java 11;
  - Unique Identifier: Gitlab tag *v2.0.3*
  - Format: ZIP file (.zip) of several JCA source code examples (.java)
- kkmip_java_examples v1.14.0: kkmip-java client examples for Java 8;
  - Unique Identifier: Gitlab tag *v1.14.0*
  - Format: ZIP file (.zip) of several kkmip-java source code examples (.java)
- kkmip_java_examples v4.2.0: kkmip-java client examples for Java 11;
  - Unique Identifier: Gitlab tag *v4.2.0*
  - Format: ZIP file (.zip) of several kkmip-java source code examples (.java)
- pkcs11_module_examples: PKCS#11 Module examples;
  - Unique Identifier: Not version controlled
  - Format: ZIP file (.zip) of PKCS#11 Module code examples (.c or .cpp)
- secure_execution_examples: Secure execution apps examples;

- Unique Identifier: Not version controlled
- Format: ZIP file (.zip) of example SEApps (.py)

* xml_examples: XML examples.
  - Unique Identifier: Not version controlled
  - Format: ZIP file (.zip) of example XML requests (.xml)

– **KMIP Clients**:
  * kkmip_cpp v2.3.0: Kryptus' KMIP C++ Client. Includes the headers and libraries for Windows and Linux (Debian based);
    - Unique Identifier: Gitlab tag *v2.3.0*
    - Format: Dynamic Library, .dll for Windows and .so for Linux; Headers (hpp) and Documentation (.html)
  * kkmip-py v2.16.0: Kryptus' KMIP Python Client;
    - Unique Identifier: Gitlab tag *v2.16.0*
    - Format: Python source code (.py)
  * kkmip-java v1.14.0: Kryptus' KMIP Java Client for Java 8;
    - Unique Identifier: Gitlab tag *v1.14.0*
    - Format: JAR file (.jar)
  * kkmip-java v4.2.0: Kryptus' KMIP Java Client for Java 11;
    - Unique Identifier: Gitlab tag *v4.2.0*
    - Format: JAR file (.jar)
  * kkmip-js v2.3.0: Kryptus' KMIP JavaScript Client.
    - Unique Identifier: Gitlab tag *v2.3.0*
    - Format: JavaScript source code (.js)

– **Management interfaces**:
  * command_line_client v2.16.0: Python application for managing the kNET HSM;
    - Unique Identifier: Gitlab tag *v2.16.0*
    - Format: Python source code (.py)
  * kNET Manager v1.13.0: Graphical interface for managing the kNET HSM.

    · Unique Identifier: Gitlab tag *v1.13.0*
    · Format: Installer, .deb for Linux, .exe for Windows and .tar.gz for RedHat systems

- **Support documentation**:
  - kkmip-java-1.14.0-doc: The kkmip-java client documentation for Java 8;
    - ⋆ Unique Identifier: Gitlab tag *v1.14.0*
    - ⋆ Format: ZIP file of documentation files in HTML format
  - kkmip-java-4.2.0-doc: The kkmip-java client documentation for Java 11;
    - ⋆ Unique Identifier: Gitlab tag *v4.2.0*
    - ⋆ Format: ZIP file of documentation files in HTML format
  - kkmip-py-2.16.0-doc: The kkmip-py client documentation;
    - ⋆ Unique Identifier: Gitlab tag *v2.16.0*
    - ⋆ Format: ZIP file of documentation files in HTML format
  - KMIP_Protocol_Overview - v1.4.1: A brief description about the KMIP protocol;
    - ⋆ Unique Identifier: Gitlab tag *kmip_v1.4.1*
    - ⋆ Format: PDF file (.pdf)
  - kNET_HSM_CLI_English_Manual - v1.12.2: The Command Line Interface (CLI) documentation in English;
    - ⋆ Unique Identifier: Gitlab tag *v1.12.2*
    - ⋆ Format: PDF file (.pdf)
  - kNET_HSM_Cluster_Guide_English - v1.1.2: A brief description about clusters in English;
    - ⋆ Unique Identifier: Gitlab tag *cluster_guide_v1.1.2*
    - ⋆ Format: PDF file (.pdf)
  - kNET_HSM_Frontal_Board_Manual - v1.1.0: The kNET's frontal board documentation;

  - ⋆ Unique Identifier: Gitlab tag *fb_v1.1.0*
  - ⋆ Format: PDF file (.pdf)

- kNET_HSM_Integration_With_APIs_English - v1.4.0: The documentation about how to integrate APIs with the HSM in English;
  - ⋆ Unique Identifier: Gitlab tag *apis_v1.4.0*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_Integration_With_EJBCA - v1.0: A brief description about how to integrate EJBCA with the HSM;
  - ⋆ Unique Identifier: Gitlab commit hash *d3ff71df*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_KMIP_Operations - v1.12.2: The kNET's operations documentation;
  - ⋆ Unique Identifier: Gitlab tag *kmip_operations_v1.12.2*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_Manager_English_Manual - v1.8.1: The kNET Manager (GUI) documentation in English;
  - ⋆ Unique Identifier: Gitlab tag *gui_v1.8.1*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_Operators_English - v1.1.1: A brief description about kNET's operators in English;
  - ⋆ Unique Identifier: Gitlab tag *operators_v1.1.1*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_Quick_Start_Guide - v1.4.0: A brief description about how to use and manage the HSM;
  - ⋆ Unique Identifier: Gitlab tag *quick_start_guide_v1.4.0*
  - ⋆ Format: PDF file (.pdf)
- kNET_HSM_Secure_Execution - v1.1.2: The documentation about Secure Execution Apps.
  - ⋆ Unique Identifier: Gitlab tag *secure_execution_v1.1.2*

    * Format: PDF file (.pdf)

  - kNET_HSM_Rack_Install_English_Manual - v1.0.0: A brief description in English on how to install the kNET on a server rack;
    * Unique Identifier: Gitlab commit hash *61903296*
    * Format: PDF file (.pdf)
  - kNET_HSM_Printable_Leaflet_English_v1.0: A physical leaflet sent inside the kNET HSM's box.
    * Unique Identifier: Gitlab commit hash *6f1a04ac*
    * Format: Physical paper
  - Preparative_Procedures - v1.2.1 - A brief document that indicates the correct way to unpack and verify the delivered equipment.
    * Unique Identifier: Gitlab tag *preparative_procedures_v1.2.1*
    * Format: PDF file (.pdf)
  - Common_Criteria_Mode_Manual - v1.1.2 - A guide to help the kNET HSM's administrator (PCO) configure the equipment to operate in the Common Criteria compliant mode.
    * Unique Identifier: Gitlab tag *cc_mode_manual_v1.1.2*
    * Format: PDF file (.pdf)

During the final steps of manufacturing, the TOE is assembled with its required Non-TOE components, properly packaged with some interface cables that may be useful and shipped to the buyer. Besides the server rack installation and connection of the power supplies to the grid and Ethernet interface to the network, no assembly on site is required. It also includes smartcards for use with client-token based authentication.

3.5.2 Manuals and APIs

The Manuals and APIs may be requested and delivered over through SFTP or email.

3.5.3 Update Packages

The update packages are created and sent on demand through an SFTP account, hosted at the company's server, in a secure and confidential manner. In addition, every package is specifically targeted at the devices to be updated, in such a way that it is not possible to reuse them on devices with different serial numbers. Since it is possible for a customer to purchase one or more devices, the same update package can take those devices to the same final version, regardless of their current version.

3.6 Non-TOE Components

3.6.1 Non-TOE Hardware

In order to function, the TOE requires two redundant power supplies with automatic interchange in case of failure and the Frontal Board, all assembled inside a chassis. All of these items are provided on purchase and assembled before the TOE's delivery.

The Frontal Board is a separate device and isn't considered an integral part of the TOE. It consists of: a frontal LCD display, 4 touch sensitive buttons (UP, DOWN, CONFIRM and MENU), power button with a LED ring and its circuit board. The Frontal Board monitors the TOE status and obtains some system information, such as temperature, battery level, fans status, intrusion state and others. Also, some operations can be executed in case of loss of the administrator password or device's inaccessibility to the network, such as factory reset and network configuration, respectively.

At last, smartcards, which are non-TOE components, are provided on purchase for client-token based authentication with the TOE, as mentioned in 3.5.1.

3.6.2 Non-TOE Software

The monitoring firmware executed in the Frontal Board is a non-modifiable Non-TOE software.

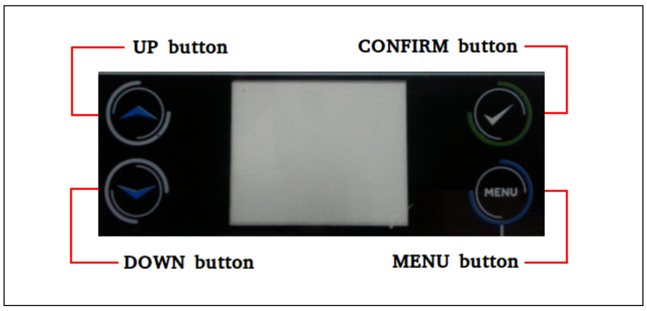


Figure 3.9: Frontal LCD display and the 4 touch sensitive buttons

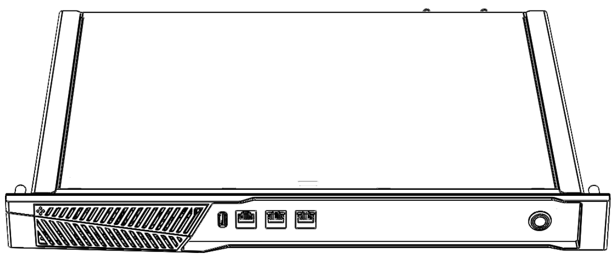


Figure 3.10: Frontal view of ASI-HSM AHX5 kNET

4 Conformance Claims

4.1 CC Conformance Claim

This Security Target is conformant to Common Criteria version 3.1 revision 5.
More precisely, this ST is conformant to:

- Common Criteria for Information Technology Security Evaluation. Part 2: Security functional components. April 2017. Version 3.1, Revision 5. CCMB-2017-04-002 [3], extended only with FCS_RNG.1 (Generation of Random Numbers) and FPT_TST_EXT.1 (Basic TSF Self Testing).
- Common Criteria for Information Technology Security Evaluation. Part 3: Security assurance components. April 2017. Version 3.1, Revision 5. CCMB-2017-04-003 [3], of which the assurance components are the only basis for the SARs in this ST.

The assurance requirement of this Security Target is EAL4 augmented with:

- AVA_VAN.5 Advanced methodical vulnerability analysis
- ALC_FLR.3 Systematic flaw remediation

The Common Methodology for Information Technology Security Evaluation [2], Version 3.1, Revision 5, April 2017 has to be taken into account.

4.2 PP Conformance Claim

This Security Target claims strict conformance with PP 419221-5 (Protection Profile for TSP Cryptographic Modules - Part 5) [12].

4.3 PP Conformance Claim Rationale

This ST conforms with PP 419221-5 [12] given this PP's relevance to Hardware Security Modules (HSMs) and its requirement by Electronic Identification and Trust Services (eIDAS).

4.4 Conformance Statement

This Security Target follows strict conformance with Protection Profile 419221-5 [12] as required by the Protection Profile.

5 Security Problem Definition

5.1 Assets

The assets that need to be protected by the TOE are identified in PP 419221-5 [12], section 6.1.

5.2 Subjects

The subjects identified in this Security Target are described in PP 419221-5 [12], section 6.2, with a single modification bold underlined at S.Admin described below. The two types of administrator mentioned, VCO and PCO, are detailed in Section 3.4.5.3.

Table 5.1: Subjects - Extension

Subjects	Description
S.Admin	An administrator of the TOE. Administrators are responsible for performing the TOE initialisation, TOE configuration and other TOE administrative functions. **There are two types of administrator: VCO and PCO. The difference between them is that they manage different scopes: VCO manages an environment called "VHSM" and the PCO manages an environment called "PHSM"**.

5.3 Threats

The threats addressed by the TOE are listed in PP 419221-5 [12], section 6.3.

5.4 Organisational Policies

The applicable organisational policies are listed in PP 419221-5 [12], section 6.4.

5.5 Assumptions

The assumptions are listed in PP 419221-5 [12], section 6.5.

6 Security Objectives

Common Criteria identifies two categories of security objectives and the purpose of this chapter is to show which security concerns are addressed by the TOE, and which security concerns are addressed by the TOE environment, or both.

6.1 Security Objectives for the TOE

The security objectives that are to be addressed by the TOE are defined in PP 419221-5 [12], section 7.2.

6.2 Security Objectives for the Environment

The security objectives that are to be addressed by the operational environment are defined in PP 419221-5 [12], section 7.3.

7 Extended Components Definition

7.1 Security Functional Requirements

All Extended Components for SFRs used in this ST are defined in PP 419221-5, section 8 [12] and described in this ST, chapter 8.

7.2 Security Assurance Requirements

There are no Extended Components for SARs.

8 Security Requirements

This chapter provides SFRs and SARs that must be satisfied by the TOE and the environment.

The SFRs components were taken from Protection Profile 419221-5, section 6.3 [12].

The SARs references assurance components from CC Part 3 [3].

Formatting choices from PP 419221-5 [12] were not modified, **bold underlined** text indicates the operations performed on components by this ST. Additionally, if the text has a **~~strike-through~~**, it means that it's not applicable to the TOE.

8.1 Security Functional Requirements

8.1.1 Cryptographic Support (FCS)

8.1.1.1 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 **list of key generation algorithm specified in Key Generation Table** [8.1] and specified cryptographic key sizes **specified in Key Generation Table** [8.1] that meet the following: **list of standards specified in Key Generation Table** [8.1].

Table 8.1: Key Generation Table

Key generation algorithm	Key size(s)	Standard
Asymmetric Key Generation	See FCS_COP.1	FIPS 186-5[23] SP 800-90A
Symmetric Key Generation	See FCS_COP.1	Direct generation using FCS_RNG.1

Application Note 12

- Key generation is linked to the setting of its security attributes (including the link to a subject who owns the key, via the setting of authorisation data) as in FMT_MSA.1/GenKeys and FMT_MSA.1/AKeys.
- The internal RNG of the TOE (see FCS_RNG.1) is used in the key generation process.

8.1.1.2 FCS_CKM.4 Cryptographic key destruction

- Hierarchical to: no other components.
- Dependencies:
  - 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 zeroisation that meets the following: **FIPS-140-2 Level 3** [25].

Application Note 13
Although the referenced FIPS-140-2 version has been superseded in 2019 by FIPS-140-3, the TOE is only certified in the older version.

8.1.1.3 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 **list of cryptographic operations specified in Cryptographic Operations Table** [8.2] in accordance with a specified cryptographic algorithm **specified in Cryptographic Operations Table** [8.2] and cryptographic key sizes **specified in Cryptographic Operations Table** [8.2] that meet the following: **list of standards specified in Cryptographic Operations Table** [8.2].

Table 8.2: Cryptographic Operations Table

Cryptographic Operation	Cryptographic Algorithm	Key Size(s)	Standard
Sign	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	Digital signature schemes PSS and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	DSA	2048 and 3072 bits	FIPS-186-5 [23]
	ECDSA	256, 384, 512 and 521 bits	FIPS-186-5 [23], RFC5639 [10, 26]
Verify	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	digital signature schemes PSS and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	DSA	2048 and 3072 bits	FIPS-186-5 [23]
	ECDSA	256, 384, 512 and 521 bits	FIPS-186-5 [23], RFC5639 [10, 26]
Encrypt	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	Encryption schemes OAEP and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	ECIES	256, 384, 512 and 521 bits	Key establishment scheme ECIES-KEM defined in ISO/IEC 18033-2:2006 [13]
	AES	128, 192 and 256 bits	AES: FIPS-197 [16] Block Ciphers having support for Block Modes CBC and CTR defined in NIST SP 800-38A [18] and GCM defined in NIST SP 800-38D [17]
Decrypt	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	Encryption schemes OAEP and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	ECIES	256, 384, 512 and 521 bits	key establishment scheme ECIES-KEM defined in ISO/IEC 18033-2:2006 [13]
	AES	128, 192 and 256 bits	AES: FIPS-197 [16] Block Ciphers having support for Block Modes CBC and CTR defined in NIST SP 800-38A [18] and GCM defined in NIST SP 800-38D [17]
Derive Key	AES	128, 192 and 256 bits	AES: FIPS-197 [16] Block Ciphers having support for Block Modes CBC and CTR defined in NIST SP 800-38A [18] and GCM defined in NIST SP 800-38D [17]

Table 8.2: Cryptographic Operations Table

Cryptographic Operation	Cryptographic Algorithm	Key Size(s)	Standard
Hash	SHA-1 (legacy), SHA-256, SHA-384, SHA-512 and SHA-512/256	None	Secure Hash Standard (SHS) FIPS PUB 180-4, section 4 Functions and Constants [24], Shake-256 in SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, section 6 SHA-3 Function Specifications [21]
MAC Generate	CMAC using AES keys	For AES: 128, 192 and 256 bits	NIST SP800-38B [19]
	HMAC with hashing algorithms SHA-1 (legacy), SHA-224, SHA-256, SHA-384 and SHA-512	Any multiple of 8 bits	The Keyed-Hash Message Authentication Code (HMAC) [22]
MAC Verify	CMAC using AES keys	For AES: 128, 192 and 256 bits	NIST SP800-38B [19]
	HMAC with hashing algorithms SHA-1 (legacy), SHA-224, SHA-256, SHA-384 and SHA-512	Any multiple of 8 bits	The Keyed-Hash Message Authentication Code (HMAC) [22]
Validate (Certificate Chain)	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [4, 27, 9, 7] and digital signature schemes PSS and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	DSA	2048 and 3072 bits	Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [4, 27, 9, 7] and DSA scheme defined in FIPS-186-5 [23]
	ECDSA	256, 384, 512 and 521 bits	FIPS-186-5 [23], RFC5639 [10, 26] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile [4, 27, 9, 7] and digital signature scheme EC-DSA defined in FIPS-186-5 [23]
Sign XML	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	XML Signature Syntax and Processing Version 1.1 [1] and digital signature schemes PSS and PKCS1v1.5 (legacy) defined in RFC8017 [8]
	DSA	2048 and 3072 bits	XML Signature Syntax and Processing Version 1.1 [1] and DSA scheme defined in FIPS-186-5 [23]
	ECDSA	256, 384, 512 and 521 bits	FIPS-186-5 [23], RFC5639 [10, 26] XML Signature Syntax and Processing Version 1.1 [1] and digital signature scheme EC-DSA defined in FIPS-186-5 [23]
Generate CSR	RSA	Multiple of 16 bits with a minimum size of 3000 bits and maximum of 8192 bits	PKCS10: Certification Request Syntax Specification. Version 1.7 [11, 26] and FIPS PUB 186-5, section 5. The RSA Digital Signature Algorithm [23]
	ECDSA	256, 384, 512 and 521 bits	FIPS-186-5 [23], RFC5639 [10, 26] PKCS10: Certification Request Syntax Specification. Version 1.7 [11, 26] and digital signature scheme EC-DSA defined in FIPS-186-5 [23]

Application Note 14

The Verify operation from the Cryptographic Operations Table [8.2] covers the signature verification algorithm that is also used to verify the authenticity (including integrity) of the main cryptographic module firmware on power-on in support of FPT_TST_EXT.1.

This function is used to verify both the signature on the stored firmware ahead of execution alongside the validation of firmware update packages

and backup packages.
Although the referenced FIPS-140-2 version has been superseded in 2019 by FIPS-140-3, the TOE is only certified in the older version.

8.1.1.4 FCS_RNG.1 Generation of random numbers

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FCS_RNG.1.1: The TSF shall provide a **hybrid physical** random number generator that implements: **DRBG-HASH using SHA-256 [20] seeded with 384-bit value from hardware TRNG, with automatic reseeding every 10000000 requests**.
FCS_RNG.1.2: The TSF shall provide **octets of bits** that meet **256 bits of security per NIST-SP-800-90A [20]**.

Application Note 15
The TOE's TRNG is used to generate cryptographic keys (FCS_CKM.1), initialization vectors for encryption (FCS_COP.1) and PSS paddings for signatures (FCS_COP.1).

8.1.2 Identification and authentication (FIA)

8.1.2.1 FIA_UID.1 Timing of identification

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FIA_UID.1.1: The TSF shall allow

(1) Self test according to FPT_TST_EXT.1

(2) **The following list of unauthenticated functions:**

   (a) **Via network:**

      i. **Query**
      ii. **Discover Versions**

   (b) **Via physical access:**

      i. **Get Status**
      ii. **Get Temporary PIN**
      iii. **Configure Network**
      iv. **Get Network Configuration**
      v. **Reset HSM**
      vi. **Get Date Time**
      vii. **Get Serial Number**
      viii. **Get Intrusion Log**
      ix. **Force Cluster Recovery**
      x. **Shutdown**
      xi. **Restart**

on behalf of the user to be performed before the user is identified.

FIA_UID.1.2: The TSF shall require each user to be successfully identified before allowing any other TSF-mediated actions on behalf of that user.

Application Note 16
This SFR applies to the HSM explicit roles (i.e. PCO, VCO and User [3.4.5.3]).

8.1.2.2 FIA_UAU.1 Timing of authentication

- Hierarchical to: no other components.

- Dependencies:
  - FIA_UID.1 Timing of identification

FIA_UAU.1.1: The TSF shall allow

(1) Self-test according to FPT_TST_EXT.1,

(2) Identification of the user by means of TSF required by FIA_UID.1

(3) **List of unauthenticated functions: Query, Discover Versions via network; Get Status, Get Temporary PIN, Configure Network, Get Network Configuration, Reset HSM, Get Date Time, Get Serial Number, Get Intrusion Log, Force Cluster Recovery, Shutdown, Restart via physical access**

on behalf of the user to be performed before the user is authenticated.

FIA_UAU.1.2: The TSF shall require each user to be successfully authenticated before allowing any other TSF-mediated actions on behalf of that user.

Application Note 17
This SFR applies to the HSM explicit roles (i.e. PCO, VCO and User). Moreover, Identification and Authentication are done simultaneously, which is why the list of allowed actions is the same for FIA_UID.1 and FIA_UAU.1.

8.1.2.3 FIA_AFL.1 Authentication failure handling

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

FIA_AFL.1.1: The TSF shall detect when **1 or more** unsuccessful authentication or authorisation attempts occur related to consecutive failed authentication or authorisation attempts **using authentication methods described in Authentication Failure Handling Table** [8.3].

FIA_AFL.1.2: When the defined number of unsuccessful authentication or authorisation attempts has been **met**, the TSF shall block access to **all functions that require authentication for that operator** until **meeting an Unblock Condition from Authentication Failure Handling Table** [8.3].

Table 8.3: Authentication Failure Handling Table

Authentication Method	Unsuccessful Authentication or Authorisation Attempts Detection	Unblock Condition
Certificated-based	1 or more attempts	Successful authentication of the subject
Password-based, Quorum Authentication, Time-based OTP, HMAC-based OTP and Client Token	1 attempt	A time period of 1 second has elapsed
	2 attempts	A time period of 2 seconds has elapsed
	3 attempts	A time period of 4 seconds has elapsed
	4 attempts	A time period of 8 seconds has elapsed
	5 or more attempts	A time period of 16 seconds has elapsed

8.1.2.4 FIA_UAU.6/KeyAuth Re-authenticating

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FIA_UAU.6.1/KeyAuth: The TSF shall authorise and re-authorise the user for access to a secret key under the conditions

(1) Authorisation in order to be granted initial access to the key; and

(2) **Authorisation on every subsequent access to the key**.

8.1.3 User data protection (FDP)

8.1.3.1 FDP_IFC.1/KeyBasics Subset information flow control

- Hierarchical to: no other components.
- Dependencies:
  - FDP_IFF.1 Simple security attributes

FDP_IFC.1.1/KeyBasics: The TSF shall enforce the Key Basics SFP on

(1) subjects: all;

(2) information: keys;

(3) operations: all.

8.1.3.2 FDP_IFF.1/KeyBasics Simple security attributes

- Hierarchical to: no other components.
- Dependencies:
  - FDP_IFC.1 Subset information flow control
  - FMT_MSA.3 Static attribute initialisation

FDP_IFF.1.1/KeyBasics: The TSF shall enforce the Key Basics SFP based on the following types of subject and information security attributes:

(1) whether a key is a secret or a public key

(2) whether a secret key is an Assigned Key

(3) whether channels selected to export keys are secure

(4) the value of the Export Flag of a key.

FDP_IFF.1.2/KeyBasics: The TSF shall permit an information flow between a controlled subject and controlled information via a controlled operation if the following rules hold:

(1) Export of secret keys shall only be allowed provided that the secret key is not an Assigned Key, that the secret key is encrypted, and that a secure channel (providing authentication and integrity protection) is used for the export

(2) Public keys shall always be exported with integrity protection of their key value and attributes

(3) Keys shall only be imported over a secure channel (providing authentication and integrity protection)

(4) A secret key can only be imported if it is a non-Assigned key

(5) Secret keys shall only be imported in encrypted form or using split-knowledge procedures requiring at least two key components to reconstruct the key, with key components supplied by at least two separately authenticated users

Application Note 19

A secure channel for export of keys in FDP_IFF.1.2/KeyBasics (1) or for import of keys in FDP_IFF.1.2/KeyBasics (3) is one that meets the requirements of FTP_TRP.1/Local or FTP_TRP.1/External.
The encrypted form required for keys imported or exported over a secure channel requires encryption of the key itself, in addition to any encryption provided by the secure channel.
Since FMT_MTD.1/Unblock is not applicable, FDP_IFF.1.2/KeyBasics (6) from PP is not applicable as well.

Application Note 20

This SFR is supported by the following FCS_COP iterations:

- Symmetric encryption of keys for Key Export/Import: FCS_COP.1 (Encrypt-AES).
- Asymmetric encryption of keys of Key Export/Import: FCS_COP.1 (Encrypt-RSA and Encrypt-ECIES).

FDP_IFF.1.3/KeyBasics: The TSF shall enforce the following additional information flow control rules: none.

FDP_IFF.1.4/KeyBasics: The TSF shall explicitly authorise an information flow based on the following rules: none.

FDP_IFF.1.5/KeyBasics: The TSF shall explicitly deny an information flow based on the following rules:

(1) No subject shall be allowed to access the plaintext value of any secret key directly.

(2) No subject shall be allowed to export a secret key in plaintext.

(3) No subject shall be allowed to export an Assigned Key.

(4) No subject shall be allowed to export a secret key without submitting the correct authorisation data for the key

(5) No subject shall be allowed to access intermediate values in any operation that uses a secret key

(6) A key with an Export Flag value marking it as non-exportable shall not be exported.

Application Note 21

Object Importation: The following attributes are set when not supplied in an object importation:

- CryptographicUsageMask: Zero, if not supplied
- Sensitive: False, if not supplied
- Extractable: True, if not supplied
- OperationPolicyName: default, if not supplied
- AlwaysSensitive: False, if not supplied
- NeverExtractable: False, if not supplied
- Backupable: True, if not supplied

Object Exportation: The following attributes are represented when an object is exported:

- Object Type: Certificate, Symmetric Key, Public Key, Private Key, Split Key, Template, Secret Data, Opaque Object or PGP Key.
- Unique Identifier: The Unique Identifier of the object.
- Managed Object: The object being returned.

8.1.3.3 FDP_ACC.1/KeyUsage Subset access control

- Hierarchical to: no other components.
- Dependencies:
  - FDP_ACF.1 Security attribute based access control

FDP_ACC.1.1/KeyUsage: The TSF shall enforce the Key Usage SFP on

(1) subjects: all;

(2) objects: keys;

(3) operations: all.

8.1.3.4 FDP_ACF.1/KeyUsage 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/KeyUsage: The TSF shall enforce the Key Usage SFP to objects based on the following:

(1) whether the subject is currently authorised to use the secret key

(2) whether the subject is currently authorised to change the attributes of the secret key

(3) the cryptographic function that is attempting to use the secret key.

Application Note 22
Identification and Authentication of a subject are done simultaneously (FIA_-UID.1 and FIA_UAU.1). Whether a subject is currently authorised to change the attributes of a secret key is determined by the iterations of FMT_MSA.1 in Security management [8.1.6].

FDP_ACF.1.2/KeyUsage: The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

(1) Attributes of a key shall only be changed by an authorised subject, and only as permitted in the Key Attributes Modification Table [8.4]

(2) Only subjects with current authorisation for a specific secret key shall be allowed to carry out operations using the plaintext value of that key

(3) Only cryptographic functions permitted by the secret key's Key Usage attribute shall be carried out using the secret key.

Application Note 23 (from PP)
FDP_ACF.1.2/KeyUsage (1) refers to controls over changing attributes that are specified in more detail in the iterations of FMT_MSA.1.
FDP_ACF.1.2/KeyUsage (2) requires that a key can only be used when the relevant subject has been authorised either by presenting the correct authorisation data for the key as part of the request for the operation or else the authorisation has previously been presented by the subject and the current use of the key does not yet require re-authorisation according to FIA_UAU.6/KeyAuth (meaning that the current usage is therefore within the usage constraints for time and number of uses since the last authorisation of use of the key). The reference to use of the plaintext value of the key does not imply that a subject has access to that value, only that it can be used to carry out operations within the TOE – reference to operations of this sort are thus distinguished from operations that may use an encrypted form of a secret key (e.g. for external storage of keys) and that are not necessarily

restricted in this way.

FDP_ACF.1.3/KeyUsage: The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: none.

FDP_ACF.1.4/KeyUsage: The TSF shall explicitly deny access of subjects to objects based on the following additional rules: none.

Application Note 24

The requirements of FDP_ACF.1/KeyUsage apply regardless of how the key is stored by the TOE. Exception: External storage of keys.

8.1.3.5 FDP_ACC.1/Backup Subset access control

- Hierarchical to: no other components.
- Dependencies:
  - FDP_ACF.1 Security attribute based access control

FDP_ACC.1.1/Backup The TSF shall enforce the Backup SFP on

(1) subjects: all

(2) objects: keys

(3) operations: backup, restore.

8.1.3.6 FDP_ACF.1/Backup 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/Backup: The TSF shall enforce the Backup SFP to objects based on the following:

(1) whether the subject is an administrator.

FDP_ACF.1.2/Backup The TSF shall enforce the following rules to determine if an operation among controlled subjects and controlled objects is allowed:

(1) Only authorised administrators shall be able to perform any backup operation provided by the TSF to create backups of the TSF state or to restore the TSF state from a backup

(2) Any restore of the TSF shall only be possible under at least dual person control, with each person being an administrator

(3) Any backup and restore shall preserve the confidentiality and integrity of the secret keys, and the integrity of public keys

(4) Any backup and restore operations shall preserve the integrity of the key attributes, and the binding of each set of attributes to its key.

Application Note 25

Preserving the binding of a set of attributes to its key (in FDP_ACF.1.2/Backup (4)) means that it is not possible for the attributes to be changed during a backup operation, or by modification of the backup data while it is away from the TSF.

Backups contain keys whose export flag attribute marks them as 'non-exportable'. Keys marked as 'non-backupable' are not included in backups.

The cryptographic operations used to protect confidentiality and integrity of any supported backups are described in FCS_COP.1.

The backup operation can only be done by the PCO (operator described in Operator management [3.4.5.3]).

FDP_ACF.1.3/Backup: The TSF shall explicitly authorise access of subjects to objects based on the following additional rules: none.

FDP_ACF.1.4/Backup: The TSF shall explicitly deny access of subjects to objects based on the following additional rules: none.

8.1.3.7 FDP_SDI.2 Stored data integrity monitoring and action

- Hierarchical to:

- FDP_SDI.1 Stored data integrity monitoring

- Dependencies: no dependencies.

FDP_SDI.2.1: The TSF shall monitor user data stored in containers controlled by the TSF for integrity errors on all keys (including security attributes), based on the following attributes: integrity protection data.

FDP_SDI.2.2: Upon detection of a data integrity error, the TSF shall

(1) prohibit the use of the altered data

(2) notify the error to the user.

Application Note 27
The FCS_COP.1.1 (Hash) covers the cryptographic algorithm used to check the data integrity and the FCS_COP.1.1 (Encrypt-AES) covers the cryptographic algorithm used for protection against modification access to the value of a secret key.
The integrity protection data in FDP_SDI.2.1 is included in the list of attributes identified in FMT_MSA.1/GenKeys and FMT_MSA.1/AKeys, and protects the value of the key and of its other security attributes.

8.1.3.8 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:

- authorisation data
- secret keys.

Application Note 28
Authorisation data is not stored persistently in the TOE.

8.1.4 Trusted path/channels (FTP)

8.1.4.1 FTP_TRP.1/Local Trusted Path

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FTP_TRP.1/Local: Not applicable for this TSF.

Application Note 29

There is no way to directly insert keys into the TOE, only through the Ethernet interface (External Trusted Path)

8.1.4.2 FTP_TRP.1/External Trusted Path

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FTP_TRP.1.1/External: The TSF shall provide a communication path between itself and remote external client applications that is logically distinct from other communication paths and provides assured authentication of its end points and protection of the communicated data from modification and disclosure.

FTP_TRP.1.2/External: The TSF shall permit **remote external client applications** to initiate communication via the trusted path.

FTP_TRP.1.3/External: The TSF shall require the use of the trusted path for **all services**.

Application Note 30

Go library [5] provides the cryptographic functions that contribute to the implementation of the trusted path, such as: AES-256-GCM-SHA384, AES-128-CBC-SHA256 and AES-128-GCM-SHA256 for encryption; the Elliptic Curve

Diffie-Hellman (ECDH), for the key exchange. Since newer versions of the Go library can't be compiled for the architecture of the TOE, the version of the tls library was frozen at 1.10.3 and can't be updated. All the following vulnerabilites found in this old version don't affect the TOE:

1. GO-2021-0243: This vulnerability does not pose a significant risk as the application does not leak any data and is automatically restarted by a watchdog in the event of a crash;
2. GO-2022-0531: The TOE does not support resuming previous TLS connections, as each connection is closed after a request is completed;
3. GO-2023-1570: This vulnerability is not applicable to the TOE as it uses TLS 1.2 for server communication;
4. GO-2023-1987: The TOE enforces a maximum allowed size of 8192 bits for RSA keys in the Common Criteria mode;
5. GO-2023-2375: The TOE does not support RSA key exchange suites in the CC Mode.

Particularly in this TOE, the hash function used in the implementation of the trusted path is provided by the FCS_COP.1.1 (Hash).
The SFRs FIA_UID.1 and FIA_UAU.1 provide the authentication of the end points.

8.1.5 Protection of the TSF (FPT)

8.1.5.1 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.

Application Note 31
The TOE provides timestamps suitable for supporting the time in an audit record for FAU_GEN.1. The internal clock is managed by a RTC module with an accuracy of around 3 ppm (parts per million). For more details, see the PCF2127AT Datasheet [14].

8.1.5.2 FPT_TST_EXT.1 Basic TSF Self Testing

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FPT_TST_EXT.1.1: The TSF shall run a suite of the following self-tests during initial start-up (or power-on) and **at the conditions when a key pair is created** to demonstrate the correct operation of the TSF:

- At initial start-up (or power-on):
  - Software/firmware integrity test
  - Cryptographic algorithm tests
  - Random number generator tests
- **When a key pair is created:**
  - **Pair-wise consistency test**

8.1.5.3 FPT_PHP.1 Passive detection of physical attack

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FPT_PHP.1.1: The TSF shall provide unambiguous detection of physical tampering that might compromise the TSF.

FPT_PHP.1.2: The TSF shall provide the capability to determine whether physical tampering with the TSF's devices or TSF's elements has occurred.

8.1.5.4 FPT_PHP.3 Resistance to physical attack

- Hierarchical to: no other components.
- Dependencies: no dependencies.

FPT_PHP.3.1: The TSF shall resist **tampering with voltage and temperature** to the **TOE board** by responding automatically such that the SFRs are always enforced.

Application Note 34
This SFR is linked to the requirements for passive detection of physical attacks in FPT_PHP.1, and the relevant responses of the TOE are listed below:

- Emits an audible alert (continuously, at regular intervals)
- Changes the frontal board screen to red color and displays “Intrusion attempts were detected”
- Blocks operations from remote external client applications
- Zeroises the cache

8.1.5.5 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:

(1) Self-test according to FPT_TST_EXT.1 fails

(2) Environmental conditions are outside normal operating range (including temperature and power)

(3) Failures of critical TOE hardware components (including the RNG) occur

(4) Corruption of TOE software occurs

(5) **None**.

8.1.6 Security management (FMT)

8.1.6.1 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 Administrator, **External Client Application**, Key User, **Virtual Crypto Officer (VCO)**.

FMT_SMR.1.2: The TSF shall be able to associate users with roles.

Application Note 36
Since the FTP_TRP.1/Local Trusted Path is not applicable, the Local Client Application is consequently not supported by the TOE .
The TOE supports External Client Application (FTP_TRP.1/External Trusted Path), that may sometimes represent an Administrator (PCO), Key User and VCO.

8.1.6.2 FMT_SMF.1 Security 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:

(1) Unblock of access due to authentication or authorisation failures

(2) Modifying attributes of keys;

(3) Export and deletion of the audit data, which can take place only under the control of the Administrator role

(4) **Backup and restore functions**;

(5) **Key import function**;

(6) **Key export function**.

Application Note 37
The requirement "Unblock of access due to authentication or authorisation failures" is managed by the TOE itself, which automatically unblocks access after a certain amount of time has passed, as described in FIA_AFL.1.2. Only the PCO administrator can export the full audit data, the VCO administrator can only export the parts that are relevant to its scope but can't delete any of it.

8.1.6.3 FMT_MTD.1/Unblock 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/Unblock: The TSF shall restrict the ability to unblock the **none** to **no one**.

Application Note 38
The TOE does not block as a result of failed attempts of authentication or authorisations, but imposes conditions to new attempts described in FIA_-AFL.1 [8.3]. Only the functions listed in FIA_UID.1.1 (2) and FIA_UAU.1.1 (3) do not require authentication. Due to the physically protected environment in which the TOE operates (as expressed in OE.Env), it is unlikely that critical operations available through physical access (that don't require authentication) can be performed without records.

8.1.6.4 FMT_MTD.1/AuditLog 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/AuditLog: The TSF shall restrict the ability to control export and deletion of the audit log records to the Administrator role.

Application Note 39

Only the PCO administrator can delete and export the full audit data, the VCO administrator can only export the parts that are relevant to its scope but can't delete any of it.

8.1.6.5 FMT_MSA.1/GenKeys Management of security attributes

- Hierarchical to: no other components.
- Dependencies:
  - [FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]
  - FMT_SMR.1 Security roles
  - FMT_SMF.1 Specification of Management Functions

FMT_MSA.1.1/GenKeys: The TSF shall enforce the Key Usage SFP to restrict the ability to modify the security attributes **listed in the Key Attributes Modification Table** [8.4] to **the users listed in Key Attributes Modification Table** [8.4].

8.1.6.6 FMT_MSA.1/AKeys Management of security attributes

- Hierarchical to: no other components.
- Dependencies:
  - [FDP_ACC.1 Subset access control, or FDP_IFC.1 Subset information flow control]
  - FMT_SMR.1 Security roles
  - FMT_SMF.1 Specification of Management Functions

FMT_MSA.1.1/AKeys: The TSF shall enforce the Key Usage SFP to restrict the ability to modify the security attributes **listed in the Key Attributes Modification Table** [8.4] to **the users listed in Key Attributes Modification Table** [8.4].

Application Note 40

The Key Attributes Modification Table [8.4] is referenced from FMT_-MSA.1/GenKeys, and FMT_MSA.1/AKeys. The required constraints on security attribute modification specified in this ST are shown in Table [8.4].
Re-authorisation data does not apply, since every cryptographic operation described in FCS_COP.1 requires identification and authentication (FIA_-UID.1 and FIA_UAU.1). Table [8.4] summarizes the key attributes and its correspondent values.

Table 8.4: Key Attributes Modification Table

Key Attribute (MSA.1)	Assigned Key	General Key
Key ID	Cannot be modified	Cannot be modified
Key Type	Cannot be modified	Cannot be modified
Authorisation Data	Modified only when modification operation includes successful validation of current (pre-modification) authorisation data	Modified only when modification operation includes successful validation of current (pre-modification) authorisation data
Re-authorisation conditions	N/A	N/A
Key Usage	Cannot be modified	Cannot be modified
Export Flag	Always False	Can only be changed from exportable to non-exportable, by the Key User (owner)
Assigned Flag	Cannot be modified	Cannot be modified
Integrity Protection Data	Cannot be modified by users (maintained automatically by TSF)	Cannot be modified by users (maintained automatically by TSF)

8.1.6.7 FMT_MSA.3/Keys Static attribute initialisation

- Hierarchical to: no other components.
- Dependencies:
  - FMT_MSA.1 Management of security attributes
  - FMT_SMR.1 Security roles

FMT_MSA.3.1/Keys: The TSF shall enforce the Key Usage SFP to provide **restrictive** default values for security attributes that are used to enforce the SFP.

FMT_MSA.3.2/Keys: The TSF shall allow the **User, referred to as "Creator" in the table, according to the constraints in the Key Attributes Initialisation Table** [8.5] to specify alternative initial values to override the default values when an object or information is created.

Table 8.5: Key Attributes Initialisation Table

Key Attribute (MSA.1)	Assigned Key	Other Key
Key ID	Initialised by generation process	Initialised by generation process
Key Type	Initialised by generation process	Initialised by generation process
Authorisation Data	Initialised by creator during generation	Initialised by creator during generation
Re-authorisation conditions	Fixed conditions	Fixed conditions
Key Usage	Initialised by creator during generation	Initialised by creator during generation/import
Export Flag	False (i.e. export is not allowed)	Initialised by creator during generation/import
Assigned Flag	Initialised by generation process	Non-assigned
Integrity Protection Data	Initialised automatically by TSF)	Initialised automatically by TSF

Application Note 41

The Key Attributes Initialisation Table [8.5] is referenced from FMT_-MSA.3/Keys and matches the attributes covered by the separate iterations of FMT_MSA.1 above. The required constraints on security attribute initialisation specified in this ST are shown in Table [8.5].

8.1.7 Security audit data generation (FAU)

8.1.7.1 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) Startup of the TOE;
(d) Shutdown of the TOE;
(e) Cryptographic key generation (FCS_CKM.1);
(f) Cryptographic key destruction (FCS_CKM.4);
(g) Failure of the random number generator (FCS_RND.1);
(h) Authentication and authorisation failure handling (FIA_AFL.1): **all unsuccessful authentication or authorisation attempts;**
(i) All attempts to import or export keys (FDP_IFF.1/KeyBasics);
(j) All modifications to attributes of keys (FDP_ACF.1/KeyUsage, FMT_-MSA.1/GenKeys and FMT_MSA.1/AKeys);
(k) Backup and restore (FDP_ACF.1/Backup): use of any backup function, use of any restore function, unsuccessful restore because of detection of modification of the backup data;
(l) Integrity errors detected for keys (FDP_SDI.2);
(m) Failures to establish secure channels (~~**FTP_TRP.1/Local,**~~FTP_TRP.1/External);

(n) Self-test completion (FPT_TST_EXT.1);
(o) Failures detected by the TOE (FPT_FLS.1);
(p) All administrative actions (FMT_SMF.1, FMT_MSA.1 (all iterations), FMT_-MSA.3/Keys);
(q) Modifications to audit parameters (affecting the content of the audit log) (FAU_GEN.1);
(r) **None**.

Application Note 42
SFR FTP_TRP.1/Local is not applicable to the TOE.

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**.

8.1.7.2 FAU_GEN.2 User identity association

- Hierarchical to: no other components.
- Dependencies:
  - FAU_GEN.1 Audit data generation
  - FIA_UID.1 Timing of identification

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.

8.1.7.3 FAU_STG.2 Guarantees of audit data availability

- Hierarchical to:
  - FAU_STG.1 Protected audit trail storage

- Dependencies:
  - FAU_GEN.1 Audit data generation

FAU_STG.2.1: The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.2.2: The TSF shall be able to **prevent** unauthorised modifications to the stored audit records in the audit trail.

FAU_STG.2.3: The TSF shall ensure that all stored audit records will be maintained when the following conditions occur: audit storage exhaustion.

8.2 Security Assurance Requirements

The security assurance requirement level is **EAL4** augmented with **AVA_VAN.5** and **ALC_FLR.3**. The assurance components, besides **ALC_FLR.3**, are identified in PP 419221-5 [12], section 9.5. **ALC_FLR.3** is identified without refinements in CC Part 3 [3].

9 Rationales

9.1 Security Objectives Rationale

9.1.1 Security Objectives Coverage

The security objectives coverage is defined in PP 419221-5 [12], section 10.1.1.

9.1.2 Security Objectives Sufficiency

The security objectives sufficiency follows the definition in PP 419221-5 [12], section 10.1.2.

9.2 Security Requirements Rationales

9.2.1 Security Requirements Coverage

The security functional requirements coverage is defined in PP 419221-5 [12], section 10.2.1. Exception: FTP_TRP.1/Local that is not applicable.

9.2.2 Security Functional Requirements Dependencies

The security functional requirements dependencies are defined in PP 419221-5 [12], section 10.2.2.

9.2.3 Rationale for Security Assurance Requirements

The security assurance requirements rationale is defined in PP 419221-5 [12], section 10.2.3.

9.2.4 AVA_VAN.5 Advanced methodical vulnerability analysis

The TOE generates, uses and manages the highly sensitive data in the form of secret keys, at least some of which may be used as signature creation data. The protection of these keys and associated security of their attributes and use in cryptographic operations can only be ensured by the TOE itself. While the TOE environment is intended to protect against physical attacks, a high level of protection against logical attacks (especially those that might be carried out remotely) is also necessary, and is therefore addressed by augmenting vulnerability analysis to deal with High attack potential.

9.2.5 ALC_FLR.3 Systematic flaw remediation

The TOE has been designed, implemented, and maintained in a consistent, thorough, and secure manner. By adding systematic flaw remediation to the evaluation, the security development lifecycle and processes are further analysed to guarantee the development and testing processes of security functions within the TOE were properly executed and all the flaws were addressed. Moreover, this addition also assures that the developed security flaws response plans are sufficient to deal with any unaccounted security flaw.

10 TOE Summary Specification

This chapter provides a description of the security functions and assurance measures of the TOE that meet the TOE security requirements.

10.1 TOE Security Functions

The security functions performed by the TOE are as follows:

- Cryptographic Support
- Identification and Authentication
- User Data Protection
- Trusted path/channels
- Protection of the TSF
- Security Management
- Security Audit Data Generation

10.1.1 Overview

In the following table 10.1 will be presented a briefing about the TOE security functions and how they are associated with all SFRs mentioned in chapter 8.

Table 10.1: TOE Summary Specification: Briefing Table

Functional Class	SFRs	Briefing
Cryptographic Support (FCS)	FCS_CKM.1 FCS_CKM.4 FCS_COP.1 FCS_RNG.1	The TOE brings the ability to manage cryptographic objects, including the creation and destruction of keys, to execute cryptographic operations, such as signing and encrypting data, and generate random numbers through these SFRs.
Identification and authentication (FIA)	FIA_UID.1 FIA_UAU.1 FIA_AFL.1 FIA_UAU.6/KeyAuth	The TOE's users can authenticate by username and password, OTP, certificate or client token. Failed attempts and failures are handled through these SFRs.
User data protection (FDP)	FDP_IFC.1/KeyBasics FDP_IFF.1/KeyBasics FDP_ACC.1/KeyUsage FDP_ACF.1/KeyUsage FDP_ACC.1/Backup FDP_ACF.1/Backup FDP_SDI.2 FDP_RIP.1	The TOE guarantees, through these SFRs, controlled access to stored data and its integrity. Also, the TOE continuously monitors all operations and in case of error, notifies the user.
Trusted path/ channels (FTP)	FTP_TRP.1/Local FTP_TRP.1/External	While the FTP_TRP.1/Local SFR is not applicable, the TOE provides, through the FTP_TRP.1/External SFR, a trusted communication path between external client applications and itself. Also the TOE ensures protection of the data along the communication from modification and disclosure.
Protection of the TSF (FPT)	FPT_STM.1 FPT_TST_EXT.1 FPT_PHP.1 FPT_PHP.3 FPT_FLS.1	The TOE runs a test suite at power-on to verify all the cryptographic algorithms and generation of random numbers and the TOE also includes detection of physical attacks through these SFRs.

Security Management (FMT)	FMT_SMR.1 FMT_SMF.1 FMT_MTD.1/Unblock FMT_MTD.1/AuditLog FMT_MSA.1/GenKeys FMT_MSA.1/AKeys FMT_MSA.3/Keys	There are several management functions that can be performed using the TOE such as modify keys' attributes, export and delete audit data, backup and restoration of data, all of them under the control of the administrator role. These functions meet the associated SFRs.
Security audit data generation (FAU)	FAU_GEN.1 FAU_GEN.2 FAU_STG.2	The TOE is capable of generating an audit record of start-up and shutdown processes, cryptographic key generation and destruction, attempts to import and export keys, attempts to backup and restore data, self test completion, and others. These records meet the associated SFRs.

10.1.2 Mapping

In the following table will be presented a mapping between the TOE functions and how it meets the SFRs described in 8.

Table 10.2: TOE Summary Specification: Mapping

SFR	Dependencies	Fullfiled by	Observation
FCS_CKM.1	[FCS_CKM.2 or FCS_COP.1] FCS_CKM.4	FCS_COP.1 FCS_CKM.4	FCS_RNG.1 corroborates directly in key generation. Symmetric keys and seeds used for asymmetric keys generation are unmodified output from the module's FIPS Approved DRBG.
FCS_CKM.4	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1]	FCS_CKM.1	-
FCS_COP.1	[FDP_ITC.1 or FDP_ITC.2 or FCS_CKM.1] FCS_CKM.4	FCS_CKM.1 FCS_CKM.4	The TOE provides a vast range of cryptographic functions available to external applications. The detailed list of algorithms are described in Cryptographic Operations Table in FCS_COP.1.
FCS_RNG.1	No dependencies	-	The TOE's TRNG is used to generate cryptographic keys (FCS_CKM.1), initialization vectors for encryption (FCS_COP.1) and PSS paddings for signatures (FCS_COP.1).
FIA_UID.1	No dependencies	-	The TOE identifies the operator for each operation request individually, except for the list presented in FIA_UID.1.1.

FIA_UAU.1	FIA_UID.1	FIA_UID.1	The TOE authenticates the operator for each operation request individually, except for the list presented in FIA_UID.1.1.
FIA_AFL.1	FIA_UAU.1	FIA_UAU.1	The TOE is not blocked by unsuccessful identification or authentication attempts. The operator is blocked under the conditions presented in Authentication Failure Handling Table in FIA_AFL.1. Refer FMT_MTD.1/Unblock Application Note 38.
FIA_UAU.6/KeyAuth	No dependencies	-	-
FDP_IFC.1/KeyBasics	FDP_IFF.1	FDP_IFF.1/KeyBasics	-
FDP_IFF.1/KeyBasics	FDP_IFC.1 FMT_MSA.3	FDP_IFC.1/KeyBasics FMT_MSA.3/Keys	-
FDP_ACC.1/KeyUsage	FDP_ACF.1	FDP_ACF.1/KeyUsage	-
FDP_ACF.1/KeyUsage	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1/KeyUsage FMT_MSA.3/Keys	Identification and Authentication of a subject are done simultaneously (FIA_UID.1 and FIA_UAU.1). Whether a subject is currently authorised to change the attributes of a secret key is determined by the iterations of FMT_MSA.1
FDP_ACC.1/Backup	FDP_ACF.1	FDP_ACF.1/Backup	-
FDP_ACF.1/Backup	FDP_ACC.1 FMT_MSA.3	FDP_ACC.1/Backup FMT_MSA.3	Backups contain keys whose export flag (refer Key Attributes Modification Table) attribute marks them as 'non-exportable'. Keys marked as 'non-backupable' (refer FDP_IFF.1 - Application Note 21) are not included in backups.
FDP_SDI.2	No dependencies	-	The TOE checks the data integrity using FCS_COP.1.1 (Hash) cryptographic algorithm and protects the data against modification using FCS_COP.1.1 (Encrypt-AES) cryptographic algorithm. The integrity protection data in FDP_SDI.2.1 is included in the list of attributes identified in FMT_MSA.1/GenKeys and FMT_MSA.1/AKeys, and protects the value of the key and of its other security attributes.
FDP_RIP.1	No dependencies	-	Authorisation data is not stored persistently in the TOE, complementing FIA_UID.1 and FIA_UAU.1, since each operation request contain its own authorisation data.
FTP_TRP.1/Local	Not applicable	-	-

FTP_TRP.1/External	No dependencies	-	The trusted path is implemented by using key exchange, encryption and hash functionalities. It is supported by Go Library and FCS_COP.1.1, along with the authentication of the end points, provided by FIA_UID.1 and FIA_UAU.1 simultaneously.
FPT_STM.1	No dependencies	-	-
FPT_TST_EXT.1	No dependencies	-	The power-on self-test are executed when the TOE initializes, with no operator intervention. If any of the tests fail, the TOE will not initialize. The list of self tests are described in [3.2] and [3.3]
FPT_PHP.1	No dependencies	-	-
FPT_PHP.3	No dependencies	-	The TOE is certified FIPS 140-2 level 3 and, consequently, meets the physical requirements to protect itself against physical attacks.
FPT_FLS.1	No dependencies	-	-
FMT_SMR.1	FIA_UID.1	FIA_UID.1	-
FMT_SMF.1	No dependencies	-	The TOE provides security management functions such as export audit data, backup and restore data. These functions contribute to monitor and maintain the module.
FMT_MTD.1/Unblock	FMT_SMR.1 FMT_SMF.1	FMT_SMR.1 FMT_SMF.1	-
FMT_MTD.1/AuditLog	FMT_SMR.1 FMT_SMF.1	FMT_SMR.1 FMT_SMF.1	-
FMT_MSA.1/GenKeys	[FDP_ACC.1 or FDP_IFC.1] FMT_SMR.1 FMT_SMF.1	FDP_ACC.1/KeyUsage FDP_IFC.1/KeyBasics FMT_SMR.1 FMT_SMF.1	-
FMT_MSA.1/AKeys	[FDP_ACC.1 or FDP_IFC.1] FMT_SMR.1 FMT_SMF.1	FDP_ACC.1/KeyUsage FDP_IFC.1/KeyBasics FMT_SMR.1 FMT_SMF.1	-
FMT_MSA.3/Keys	FMT_MSA.1 FMT_SMR.1	FMT_MSA.1/GenKeys, FMT_MSA.1/AKeys FMT_SMR.1	-
FAU_GEN.1	FPT_STM.1	FPT_STM.1	The TOE is configured to generate audit logs of all events specified in FAU_GEN.1.1. The on board Real Time Clock (RTC) is used to provide the log entries timestamp. The TOE stores audit logs persistently and rotates the log entries to prevent exhaustion of internal memory.
FAU_GEN.2	FAU_GEN.1 FIA_UID.1	FAU_GEN.1 FIA_UID.1	-
FAU_STG.2	FAU_GEN.1	FAU_GEN.1	-

Bibliography

[1] Mark Bartel et al. *XML Signature Syntax and Processing Version 1.1. W3C Recommendation 11*. Apr. 2013. URL: `https://www.w3.org/TR/xmldsig-core1/`.

[2] *Common Criteria for Information Technology Security Evaluation. Evaluation Methodology. Version 3.1, Revision 5*. Apr. 2017. URL: `https://www.commoncriteriaportal.org/cc/`.

[3] *Common Criteria for Information Technology Security Evaluation. Part 1, 2 and 3. Version 3.1, Revision 5*. Apr. 2017. URL: `https://www.commoncriteriaportal.org/cc/`.

[4] David Cooper et al. *Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC 5280. May 2008. DOI: 10.17487/RFC5280. URL: `https://tools.ietf.org/html/rfc5280`.

[5] *Go Library - TLS Overview*. URL: `https://pkg.go.dev/crypto/tls@go1.10.3`.

[6] SOG-IS Crypto Working Group. *SOG-IS Crypto Evaluation Scheme Agreed Cryptographic Mechanisms*. Feb. 2023. URL: `https://www.sogis.eu/documents/cc/crypto/SOGIS-Agreed-Cryptographic-Mechanisms-1.3.pdf`.

[7] Russ Housley. *Internationalization Updates to RFC 5280*. RFC 8399. May 2018. DOI: 10.17487/RFC8399. URL: `https://tools.ietf.org/html/rfc8399`.

[8] Ed. K. Moriarty et al. *PKCS #1: RSA Cryptography Specifications Version 2.2*. RFC 8017. Nov. 2016. DOI: 10.17487/RFC8017. URL: `https://www.rfc-editor.org/rfc/rfc8017`.

[9] Alexey Melnikov and Wei Chuang. *Internationalized Email Addresses in X.509 Certificates*. RFC 8398. May 2018. DOI: 10.17487/RFC8398. URL: `https://tools.ietf.org/html/rfc8398`.

[10] Johannes Merkle and Manfred Lochter. *Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation*. RFC 5639. Mar. 2010. DOI: 10.17487/RFC5639. URL: `https://rfc-editor.org/rfc/rfc5639.txt`.

[11] Magnus Nystrom and Burt Kaliski. *PKCS #10: Certification Request Syntax Specification. Version 1.7*. RFC 2986. Nov. 2000. DOI: 10.17487/RFC2986. URL: `https://tools.ietf.org/html/rfc2986`.

[12] *Protection Profile 419221-5, Protection Profile for TSP Cryptographic Modules - Part 5*. June 2018. URL: `https://www.une.org/encuentra-tu-norma/busca-tu-norma/norma/?c=N0060093`.

[13] ISO Central Secretariat. *Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers*. May 2006. URL: `https://www.iso.org/standard/37971.html`.

[14] PHILIPS (NXP Semiconductors). *PCF2127AT Datasheet*. 2014. URL: `https://pdf1.alldatasheet.com/datasheet-pdf/view/1221783/PHILIPS/PCF2127AT.html`.

[15] OASIS Standard. *Key Management Interoperability Protocol Specification Version 1.4 Plus Errata 01*. Ed. by Tony Cox. July 2019. URL: `https://docs.oasis-open.org/kmip/spec/v1.4/errata01/os/kmip-spec-v1.4-errata01-os-redlined.html`.

[16] National Institute of Standards and Technology (NIST). *Advanced Encryption Standard (AES), NIST Federal Information Processing Standards Publication (FIPS PUB) 197*. Ed. by NIST. May 2023. URL: `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197-upd1.pdf`.

[17] National Institute of Standards and Technology (NIST). *Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC*. Ed. by NIST. Nov. 2007. URL: `https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf`.

[18] National Institute of Standards and Technology (NIST). *Recommendation for Block Cipher Modes of Operation: Methods and Techniques*. Ed. by NIST. Dec. 2001. URL: `https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf`.

[19] National Institute of Standards and Technology (NIST). *Recommendation for Block Cipher Modes of Operation, The CMAC Mode for Authentication*. Ed. by NIST. May 2005. URL: `https://doi.org/10.6028/NIST.SP.800-38B`.

[20] National Institute of Standards and Technology (NIST). *Recommendation for Random Number Generation Using Deterministic Random Bit Generators, NIST Special Publication 800-90A*. Ed. by NIST. June 2015. URL: `https://doi.org/10.6028/NIST.SP.800-90Ar1`.

[21] National Institute of Standards and Technology (NIST). *SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions, NIST Federal Information Processing Standards Publication (FIPS PUB) 202*. Ed. by NIST. Aug. 2015. URL: `https://doi.org/10.6028/NIST.FIPS.202`.

[22] National Institute of Standards and Technology (NIST). *The Keyed-Hash Message Authentication Code (HMAC), NIST Federal Information Processing Standards Publication (FIPS PUB) 198-1*. Ed. by NIST. July 2008. URL: `https://doi.org/10.6028/NIST.FIPS.198-1`.

[23] National Institute of Standards and Technology. *Digital Signature Standard (DSS), NIST Federal Information Processing Standards Publication (FIPS PUB) 186-5*. Ed. by NIST. Feb. 2023. URL: `https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf`.

[24] National Institute of Standards and Technology. *Secure Hash Standard (SHS), NIST Federal Information Processing Standards Publication (FIPS PUB) 180-4*. Ed. by NIST. Aug. 2015. URL: `https://doi.org/10.6028/NIST.FIPS.180-4`.

[25] National Institute of Standards and Technology. *Security Requirements for Cryptographic Modules (FIPS PUB) 140-2*. Ed. by NIST. May 2001. URL: `https://doi.org/10.6028/NIST.FIPS.140-2`.

[26] Sean Turner. *The application/pkcs10 Media Type*. RFC 5967. Aug. 2010. DOI: 10.17487/RFC5967. URL: `https://rfc-editor.org/rfc/rfc5967.txt`.

[27] Peter E. Yee. *Updates to the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile*. RFC 6818. Jan. 2013. DOI: 10.17487/RFC6818. URL: `https://tools.ietf.org/html/rfc6818`.