Guardtime Federal Black Lantern® BL300 Series and BL400 with BLKSI.2.2.2-FIPS

Security Target

Version 1.1

29 April 2024

Prepared for:

1700 Diagonal Road, Suite 320,
Alexandria, VA 22314

Prepared by:

Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive
Columbia, Maryland 21046

Revision History		
Date	**Author**	**Modifications**
8 November 2021	Leidos	Initial draft.
17 November 2021	Leidos	Update following vendor review.
20 December 2021	Leidos	Update following further vendor review.
4 January 2022	Leidos	Update following vendor clarifications.
18 January 2022	Leidos	Update following evaluator review.
19 January 2022	Leidos	Update following vendor clarification and further evaluator review.
15 March 2022	Leidos	Update following validator check-in review.
22 March 2022	Leidos	Update for new NIAP TDs.
4 May 2022	Leidos	Update following TSS evaluation activities.
19 May 2022	Leidos	Update to address last issues.
1 September 2022	Leidos	Final version for publication.
29 April 2024	Guardtime Federal	Update TOE version identifier for Assurance Continuity process.

Contents

List of Tables

1 Security Target Introduction

This section introduces the Target of Evaluation (TOE) and provides the Security Target (ST) and TOE identification, ST and TOE conformance claims, ST conventions, a glossary and list of abbreviations, and an overview and description of the TOE identifying its physical and logical boundaries.

The TOE is Guardtime Federal's Black Lantern®. Black Lantern is a secure network appliance designed to mitigate both remote and local physical attacks against a customer's infrastructure and applications. Black Lantern incorporates a Keyless Signature Infrastructure (KSI®) gateway and extender, allowing for secure implementation of KSI-based data assurance and cybersecurity solutions with built-in active anti-tamper measures. Note that the KSI gateway and extender functionality of Black Lantern and its ability to support KSI-based data assurance and cybersecurity solutions with built-in active anti-tamper measures has not been evaluated.

The focus of this evaluation is on the TOE functionality supporting the claims in the collaborative Protection Profile for Network Devices ([cPPND] – see Section 1.2 for specific version information). The security functionality specified in [cPPND] includes protection of communications between the TOE and external IT entities, identification and authentication of administrators, auditing of security-relevant events, ability to verify the source and integrity of updates to the TOE, and use of NIST-validated cryptographic mechanisms.

This ST includes the following additional sections:

- Security Problem Definition (Section 2)—describes the threats and assumptions that define the security problem to be addressed by the TOE and its environment
- Security Objectives (Section 3)—describes the security objectives for the TOE and its operational environment necessary to counter the threats and satisfy the assumptions that define the security problem
- IT Security Requirements (Section 4)—specifies the security functional requirements (SFRs) and security assurance requirements (SARs) to be met by the TOE
- TOE Summary Specification (Section 5)—describes the security functions of the TOE and how they satisfy the SFRs
- Protection Profile Claims (Section 6)—provides rationale supporting the claims for conformance of the ST and the TOE to [cPPND]
- Rationale (Section 7)—provides mappings and rationale for the security problem definition, security objectives, security requirements, and security functions to justify their completeness, consistency, and suitability.

1.1 Security Target, Target of Evaluation, and Common Criteria Identification

ST Title: Guardtime Federal Black Lantern® BL300 Series and BL400 with BLKSI.2.2.2-FIPS Security Target

ST Version: Version 1.1

ST Date: 29 April 2024

TOE Identification: Guardtime Federal Black Lantern® BL300 Series and BL400 with BLKSI.2.2.2-FIPS

TOE Developer: Guardtime Federal

Evaluation Sponsor: Guardtime Federal

CC Identification: Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, April 2017.

1.2 Conformance Claims

This ST and the TOE it describes conform to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Components, Version 3.1 Revision 5, April 2017
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Components, Version 3.1 Revision 5, April 2017
  - Part 3 Conformant.

This ST and the TOE it describes conform to the following Protection Profile:

- *collaborative Protection Profile for Network Devices*, Version 2.2e, 23 March 2020 [cPPND], including the following optional and selection-based SFRs: FAU_STG.1; FAU_STG_EXT.2/LocSpace; FAU_STG_EXT.3/LocSpace; FCS_HTTPS_EXT.1; FCS_NTP_EXT.1; FCS_TLSC_EXT.1; FCS_TLSC_EXT.2; FCS_TLSS_EXT.1; FCS_TLSS_EXT.2; FIA_X509_EXT.1/Rev; FIA_X509_EXT.2; FIA_X509_EXT.3; FMT_MOF.1/Functions; and FMT_MTD.1/CryptoKeys.

The following NIAP Technical Decisions apply to this PP and have been accounted for in the ST development and the conduct of the evaluation:

- TD0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1).
- TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4.
- TD0536: NIT Technical Decision for Update Verification Inconsistency.
- TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3.
- TD0538: NIT Technical Decision for Outdated link to allowed-with list.
- TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN.
- TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test.
- TD0556: NIT Technical Decision for RFC 5077 question.
- TD0563: NiT Technical Decision for Clarification of audit date information.
- TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria.
- TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7
- TD0570: NiT Technical Decision for Clarification about FIA_AFL.1.
- TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1.
- TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers.
- TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3.
- TD0591: NIT Technical Decision for Virtual TOEs and hypervisors.
- TD0592: NIT Technical Decision for Local Storage of Audit Records.

- TD0632: NIT Technical Decision for Consistency with Time Data for vNDs—note that although this TD modifies FPT_STM_EXT.1 to cater for time stamps obtained from a virtualization platform, the TOE does not include virtualized devices.
- TD0634: NIT Technical Decision for Clarification required for testing IPv6.
- TD0635: NIT Technical Decision for TLS Server and Key Agreement Parameters.

The following NIAP Technical Decisions issued for [cPPND] are not applicable to this evaluation, for the reasons stated:

- TD0546: NIT Technical Decision for DTLS – clarification of Application Note 63—this TD provides clarification about Application Note 63, which applies to FCS_DTLSC_EXT.1, but the ST does not claim this SFR.
- TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e—this TD modifies one of the selections available in FCS_CKM.1, but the ST has not included that selection.
- TD0631: NIT Technical Decision for Clarification of public key authentication for SSH Server—this TD provides clarification of public key authentication for SSH Server, but the ST does not claim FCS_SSHS_EXT.1.
- TD0633: NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance—this TD modifies evaluation activities associated with FCS_IPSEC_EXT.1, but the ST does not claim this SFR.
- TD0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH—this TD provides clarification of public key authentication for SSH client, but the ST does not claim FCS_SSHC_EXT.1.

1.3 Conventions

The following conventions are used in this document:

- Security Functional Requirements—Part 1 of the CC defines the approved set of operations that may be applied to functional requirements: iteration; selection; assignment; and refinement.
  - Iteration—allows a component to be used more than once with varying operations. This ST includes iterated requirements reproduced from [cPPND], which uses descriptive strings to distinguish iterations of a requirement. For example, [cPPND] identifies iterations of FCS_COP.1 as follows: FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, and FCS_COP.1/KeyedHash.
  - Selection—allows the specification of one or more elements from a list. Selections completed in the ST are indicated using bold italics and are enclosed by brackets (e.g., [***selection***]).
  - Assignment—allows the specification of an identified parameter. Assignments completed in the ST are indicated using bold text and are enclosed by brackets (e.g., [**assignment**]). An assignment within a selection is identified in bold underlined italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
  - Refinement—allows the addition of details. Refinements made in the ST of requirements drawn from [cPPND] would be indicated using bold for additions and strike-through for deletions (e.g., "… ~~some~~ **all** objects).
- Other sections of the ST—other sections of the ST use bolding and/or different fonts (such as `Courier`) to highlight text of special interest, such as captions, commands, or filenames specific to the TOE.

1.4 Abbreviations and Acronyms

Table 1: Abbreviations and Acronyms

Abbreviation	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CBC	Cipher Block Chaining
CLI	Command Line Interface
CSR	Certificate Signing Request
DDR	Double Data Rate Static RAM
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
ECDSA	Elliptic Curve Digital Signature Algorithm
HMAC	Hash-based Message Authentication Code
ICMP	Internet Control Message Protocol
IP	Internet Protocol
KSI	Keyless Signature Infrastructure
MAC	Message Authentication Code
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
PBKDF	Password-Based Key Derivation Function—a key derivation function also used for password hashing. The TOE implements PBKDF2.
PKCS	Public Key Cryptography Standards
PKI	Public Key Infrastructure
POST	Power On Self Test
RAM	Random Access Memory
SCI	Serial Console Interface
SHA	Secure Hash Algorithm
SRAM	Static RAM
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality

1.5 Product Overview

Black Lantern is a network device appliance intended to mitigate both remote and physical attacks against a customer infrastructure and applications. Black Lantern includes a Keyless Signature Infrastructure (KSI) gateway and extender that provides implementation of KSI-based data assurance.

KSI is a method and a globally distributed network infrastructure for the issuance and verification of KSI signatures. Unlike traditional digital signature approaches (such as Public Key Infrastructure (PKI)), which depend on asymmetric key cryptography, KSI uses only hash-function cryptography, allowing verification to rely only on the security of hash-functions and the availability of a public ledger commonly referred to as a blockchain.

A blockchain is a distributed public ledger—an append-only record of events where each new event is cryptographically linked to the previous. New entries are created using a distributed consensus protocol.

The KSI blockchain overcomes three major weaknesses of mainstream blockchain technologies—which were designed to facilitate asset transactions—making KSI suitable also for cybersecurity and data governance applications:

- Scalability—one of the most significant challenges with traditional blockchain approaches is scalability – they scale at O(n) complexity, meaning they grow linearly with the number of transactions. In contrast the KSI blockchain scales at O(t) complexity – it grows linearly with time and independently from the number of transactions. KSI can sustain billions of asset registration events every second without growing out of control.
- Settlement time—in contrast to the widely distributed crypto-currency approach, the number of participants in KSI blockchain distributed consensus protocol is limited. By limiting the number of participants, it becomes possible to achieve consensus synchronously, eliminating the need for Proof of Work and ensuring settlement can occur within one second.
- Formal security proof—unlike other blockchains, KSI blockchain has been subjected to end-to-end formal mathematical proof that provides assurance that the protocol does precisely what it says it does.

A user interacts with the KSI system by submitting a hash value of the data to be signed into the KSI infrastructure and is then returned a signature which provides cryptographic proof of the time of signature, integrity of the signed data, as well as attribution of origin, i.e., which entity generated the signature.

Note that the KSI gateway and extender functionality of Black Lantern and its ability to support KSI-based data assurance has not been evaluated.

1.6 TOE Overview

The Target of Evaluation (TOE) comprises the Black Lantern BL300-B2, BL300-C2, and BL400-A1 appliances with firmware version BLKSI.2.2.2-FIPS from Guardtime Federal. The focus of this evaluation is on the TOE functionality satisfying the requirements specified in *collaborative Protection Profile for Network Devices* ([cPPND]).

The requirements specified in [cPPND] include protection of communications between the TOE and external IT entities, identification and authentication of administrators, auditing of security-relevant events, ability to verify the source and integrity of updates to the TOE, and use of NIST-validated cryptographic mechanisms.

1.7 TOE Description

For the purpose of this evaluation, the TOE is treated as a network device offering NIST validated cryptographic functions, security auditing, secure administration, trusted updates, self-tests, and secure connections to other servers (e.g., to export audit records), protected using HTTPS/TLS.

Cryptographic functionality is performed by Guardtime Federal's Cryptographic Support Library (CSL) Direct. The module's FIPS-Approved cryptographic algorithms have obtained CAVP certificates.

The TOE audits security relevant events, stores audit records locally, and can be configured to forward its audit records to an external syslog server in the network environment over a TLS-protected connection. An administrator can configure the TOE to solicit time from an NTP server, and alternatively the administrator can manually set the TOE's time.

The TOE supports a local administration capability via an RS-232 serial interface that provides access to its Serial Console Interface (SCI). The SCI presents a command line interface (CLI). Suitably privileged administrative users can perform all management commands and configuration of the TOE via this interface.

The TOE also supports a remote administration capability via a RESTful interface that enables an administrator to submit management requests to the TOE via calls to the TOE's RESTful API. All communication via the RESTful API is protected using HTTPS.

The TOE implements a local password-based authentication mechanism to control both local and remote access to the TOE.

Administrators are able to query the current running version of the TOE firmware and to initiate firmware updates. The vendor provides TOE updates as encrypted and digitally signed packages. The TOE uses 256 bit AES to decrypt the image and ECDSA with NIST curve P-521 to verify the digital signature.

The TOE implements a battery of Power On Self-Tests (POSTs) that ensure the integrity and correct operation of the TOE.

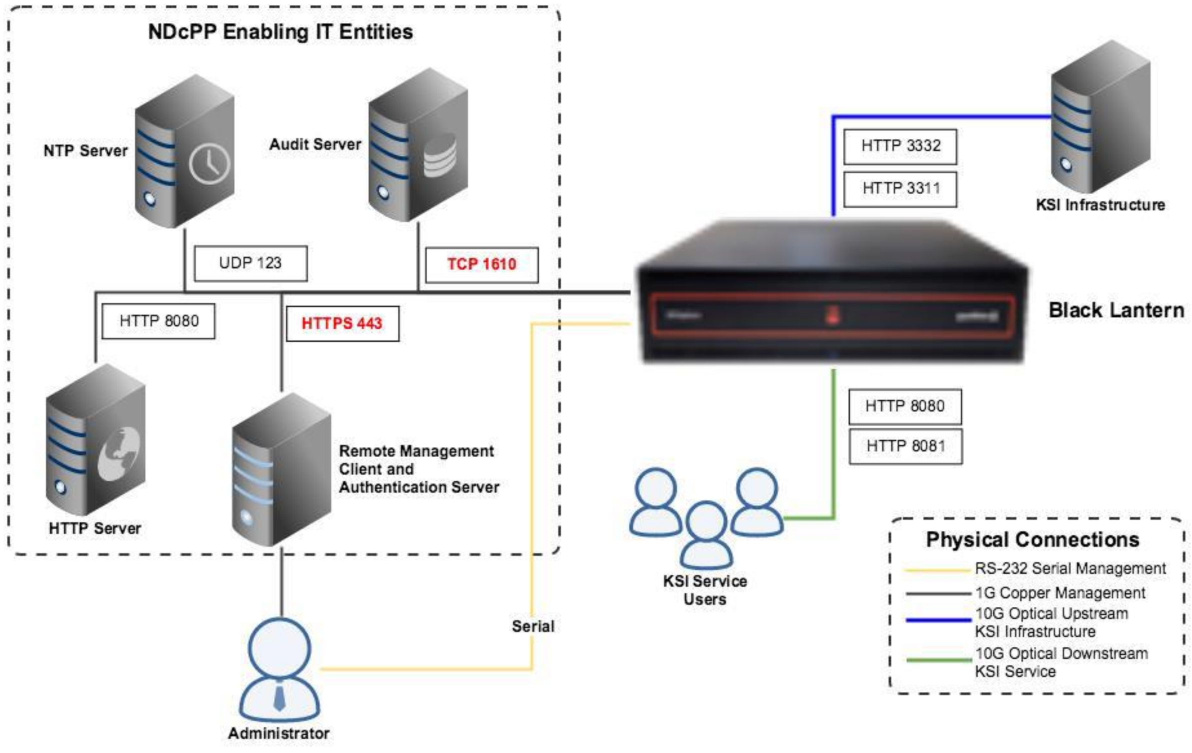
1.8 Physical Boundaries

Physically, the TOE is provisioned as a 2U (BL300) and 1U (BL400) rack mount device. The following table summarizes the hardware specifications of each of the appliance models included in the TOE, each of which is delivered pre-loaded with BLKSI.2.2.2-FIPS firmware.

Table 2: Black Lantern Hardware Appliance Specifications

Device Model	**BL300-B2 and BL300-C2**	**BL400-A1**
Processor	NXP T4240r2 QorIQ, 12 Dual Cores 64-bit Power Architecture (microarchitecture), 1667 MHz with SEC	NXP T4240r2 QorIQ, 12 Dual Cores 64-bit Power Architecture (microarchitecture), 1667 MHz with SEC
Storage	SSD: 1 TB	SSD: 1 TB Secondary SSD: 1 TB
Network Ports	4 x 10 GbE (Optical) 1 x 1 GbE (Copper)	4 x 10 GbE (Optical) 7 x 1 GbE (Copper)
Operating System	Green Hills Software (GHS) Integrity OS v11.4.4	Green Hills Software (GHS) Integrity OS v11.4.4

The following figure depicts the operational environment of the TOE.



The TOE in its evaluated configuration requires the following components in its operational environment:

- Syslog server for external storage of exported audit records
- Local management device directly connected to RS-232 serial port for local administrator access to SCI
- Remote management device supporting HTTPS for remote administrator access via RESTful interface.

The TOE in its evaluated configuration additionally supports the following optional components in its operational environment:

- NTP server supporting NTP v4 (RFC 5905)—the TOE supports synchronization to NTP servers
- HTTP server—the TOE supports connection to an HTTP server for the purpose of loading TOE updates onto the TOE (TOE update packages are encrypted and digitally signed).

The connections to the TOE not identified in the figure as “NDcPP Enabling IT Entities” are associated with provision of KSI services, for which no claims are being made, as follows:

- Downstream KSI services—the TOE provides the interface for users to obtain KSI services. Through this interface type, the user can request signing or extending services. The TOE uses ports 8080 and 8081 as default signing and extender request, respectively
- Upstream KSI Infrastructure—the TOE interfaces with the higher level aggregator in the KSI infrastructure through this type of interface. The service user does not have access to this interface—only the TOE is able to interact with the rest of the KSI infrastructure through this interface.

1.9 Logical Boundaries

This section summarizes the following security functions provided by the TOE:

- Security audit
- Cryptographic support
- Identification and authentication
- Security management
- Protection of the TSF
- TOE access
- Trusted path/channels.

1.9.1 Security Audit

The TOE is able to generate audit records of security relevant events. The TOE stores audit records locally and can also be configured to send the audit records to an external syslog server over a protected communication channel. The TOE protects locally stored audit records from unauthorized modification and deletion. By default, the TOE overwrites the oldest locally stored audit records and maintains a count of the number of overwritten records if space for storing newly generated audit records is exhausted. Alternatively, the administrator can configure the TOE to drop all new records and keep a counter of the audit records dropped when the local storage is full. In addition, the TOE generates a warning to inform the administrator before the audit trail exceeds the local audit storage capacity.

1.9.2 Cryptographic Support

The TOE includes Guardtime Federal's Cryptographic Support Library (CSL) Direct v2.0.0 cryptographic module, which provides the following CAVP-certified cryptographic services: random bit generation; asymmetric cryptographic key pair generation; key establishment; symmetric data encryption and decryption; digital signature generation and verification; cryptographic hashing; and keyed-hash message authentication. These services support implementation of higher-level cryptographic protocols, specifically TLS and HTTPS.

1.9.3 Identification and Authentication

The TOE requires all users to be successfully identified and authenticated prior to accessing its security management functions and other capabilities.

The TOE supports the local (i.e., on device) definition of administrators with usernames and passwords. When the TOE authenticates a user at the SCI, no information about the authentication data (i.e., password) is echoed to the user. Passwords can be composed of any combination of upper and lower case letters, numbers, and the following special characters: `!`; `@`; `#`; `$`; `%`; `^`; `&`; `*`; `(`; `)`; `_`; `?`; `<`; `>`; `,`; `.`; `~`; and `|`.

The TOE responds to consecutive failures to authenticate remote password-based login attempts. The TOE validates credentials in the HTTPS header of RESTful requests against a local user account and keeps a count of consecutive failed authentication attempts for each configured user. If the number of consecutive failed authentication attempts reaches the configured value for allowed failed attempts, the local account will be disabled and subject to be re-enabled by a Security Admin user. All users are subject to lockout following consecutive failed remote authentication attempts, but users with the Security Admin role can never be locked out of the SCI.

The TOE supports the use of X.509v3 certificates for TLS authentication and also supports certificate revocation checking using OCSP. The TOE will not accept a certificate if it is unable to establish a connection in order to determine the certificate's validity.

1.9.4 Security Management

The TOE supports local and remote security administration via the SCI and the RESTful API respectively.

The TOE supports the following two administrator roles that together provide the capabilities of the Security Administrator role as defined in [cPPND]—Security Admin and Network Admin.

The TOE provides the security management functions necessary to configure and administer its security capabilities. These capabilities include configuring a login access banner, configuring a local session inactivity time limit before session termination, configuring the audit function, including export of audit records to an external audit server, setting the system date and time and configuring NTP, performing firmware updates, and managing X.509 certificates.

1.9.5 Protection of the TSF

The TOE protects sensitive data such as stored passwords and cryptographic keys so that they are not accessible even by an administrator.

The TOE provides reliable time stamps for its own use and can be configured to synchronize its time via NTP.

The TOE provides a trusted means for determining the current running version of its firmware and to update its firmware. The TOE verifies the integrity of TOE updates using a digital signature.

The TOE implements various self-tests that execute during the power-on and start up sequence, including cryptographic known answer tests that verify the correct operation of the TOE's cryptographic functions.

1.9.6 TOE Access

The TOE will terminate local interactive sessions at the SCI after a configurable period of inactivity. The default time-out value is 300 minutes and this can be configured by a user with the Security Admin or Network Admin role.

The use of the RESTful API for remote security management means there is no concept of an interactive session for remote administrators—each request to the API is a self-contained, identified and authenticated request. As such, TSF-initiated termination of remote administrative sessions is deemed to occur immediately after the TOE services the request.

The TOE provides the capability for users to terminate their own local sessions by logging out of the TOE. For user-initiated termination of remote interactive sessions via the RESTful API, the interactive session is terminated immediately after the request is submitted to the interface.

The TOE can be configured to display an advisory and consent warning message before establishing a user session.

1.9.7 Trusted Path/Channels

The TOE protects communications with remote administrators using HTTPS (for access to the RESTful API).

The TOE is able to protect transmission of audit records to an external audit server using TLS.

1.10 TOE Documentation

The TOE is supplied with the following guidance documentation that describes the installation process for the TOE and provides guidance for configuration and secure use of its security features:

- Black Lantern® Guidance Documentation, Version 1.1, September 1, 2022.

2 Security Problem Definition

This ST includes by reference the Security Problem Definition (comprising threat statements, assumptions, and organizational security policies) from [cPPND], excluding A.COMPONENTS_RUNNING, which applies only to distributed TOEs, and A.VS_TRUSTED_ADMINISTRATOR, A.VS_REGULAR_UPDATES, A.VS_ISOLATION, and A.VS_CORRECT_CONFIGURATION, which apply only to virtual network devices. The PP offers additional information about the threats, assumptions, and organizational security policies, but that has not been reproduced here and the PP should be consulted if there is interest in that material.

In general, the [cPPND] has presented a Security Problem Definition appropriate for network infrastructure devices, and as such is applicable to Black Lantern.

3 Security Objectives

The [cPPND] defines the following security objectives for the operational environment of the TOE[1].

Table 3: Security Objectives for the Operational Environment

Objective	Description
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.NO_THRU_TRAFFIC_PROTECTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.
OE.TRUSTED_ADMIN	Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE’s trust store in case such certificate can no longer be trusted.
OE.UPDATES	The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_SECURE	The Administrator’s credentials (private key) used to access the TOE must be protected on any other platform on which they reside.
OE.RESIDUAL_INFORMATION	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

[1] Some of the objectives specified in [cPPND] include notes specific to virtual network devices (vNDs). Since the TOE does not include a vND, these notes have not been reproduced. Neither have the objectives OE.COMPONENTS_RUNNING and OE.VM_CONFIGURATION, which apply only to distributed TOEs and to vNDs respectively.

4 IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the TOE and to scope the evaluation effort.

The SFRs have all been drawn from [cPPND]. As such, operations on SFRs already performed in that PP are not identified here. Rather, the SFRs have been copied from [cPPND] and any formatting used in that PP has been removed. Operations performed on SFRs in the writing of this ST are identified in accordance with the conventions described in Section 1.3.

The SARs are the set of SARs specified in [cPPND].

4.1 Extended Requirements

All of the extended requirements in this ST have been drawn from [cPPND]. The [cPPND] defines the following extended SFRs and since they are not redefined in this ST, the [cPPND] should be consulted for more information in regard to these CC extensions.

- FAU_STG_EXT.1—Protected Audit Event Storage
- FAU_STG_EXT.2/LocSpace—Counting Lost Audit Data
- FAU_STG_EXT.3/LocSpace—Action in Case of Possible Audit Data Loss
- FCS_HTTPS_EXT.1—HTTPS Protocol
- FCS_NTP_EXT.1—NTP Protocol
- FCS_TLSC_EXT.1—TLS Client Protocol Without Mutual Authentication
- FCS_TLSC_EXT.2—TLS Client Support for Mutual Authentication
- FCS_TLSS_EXT.1—TLS Server Protocol Without Mutual Authentication
- FCS_TLSS_EXT.2—TLS Server Support for Mutual Authentication
- FCS_RBG_EXT.1—Random Bit Generation
- FIA_PMG_EXT.1—Password Management
- FIA_UIA_EXT.1—User Identification and Authentication
- FIA_UAU_EXT.2—Password-Based Authentication Mechanism
- FIA_X509_EXT.1/Rev—X.509 Certificate Validation
- FIA_X509_EXT.2—X.509 Certificate Authentication
- FIA_X509_EXT.3—X.509 Certificate Requests
- FPT_APW_EXT.1—Protection of Administrator Passwords
- FPT_SKP_EXT.1—Protection of TSF Data
- FPT_STM_EXT.1—Reliable Time Stamps
- FPT_TST_EXT.1—TSF Testing
- FPT_TUD_EXT.1—Trusted Update
- FTA_SSL_EXT.1—TSF-Initiated Session Locking

4.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the TOE.

Table 4: TOE Security Functional Components

Requirement Class	Requirement Component
FAU: Security audit	FAU_GEN.1—Audit data generation
	FAU_GEN.2—User identity association
	FAU_STG.1—Protected audit trail storage
	FAU_STG_EXT.1—Protected audit event storage
	FAU_STG_EXT.2/LocSpace—Counting lost audit data
	FAU_STG_EXT.3/LocSpace—Action in case of possible audit data loss
FCS: Cryptographic support	FCS_CKM.1—Cryptographic key generation
	FCS_CKM.2—Cryptographic key establishment
	FCS_CKM.4—Cryptographic key destruction
	FCS_COP.1/DataEncryption—Cryptographic operation (AES data encryption/decryption)
	FCS_COP.1/SigGen—Cryptographic operation (signature generation and verification)
	FCS_COP.1/Hash—Cryptographic operation (hash algorithm)
	FCS_COP.1/KeyedHash—Cryptographic operation (keyed hash algorithm)
	FCS_HTTPS_EXT.1—HTTPS protocol
	FCS_NTP_EXT.1—NTP protocol
	FCS_TLSC_EXT.1—TLS client protocol without mutual authentication
	FCS_TLSC_EXT.2—TLS client support for mutual authentication
	FCS_TLSS_EXT.1—TLS server protocol without mutual authentication
	FCS_TLSS_EXT.2—TLS server support for mutual authentication
	FCS_RBG_EXT.1—Random bit generation
FIA: Identification and authentication	FIA_AFL.1—Authentication failure management
	FIA_PMG_EXT.1—Password management
	FIA_UIA_EXT.1—User identification and authentication
	FIA_UAU_EXT.2—Password-based authentication mechanism
	FIA_UAU.7—Protected authentication feedback
	FIA_X509_EXT.1/Rev—X.509 certificate validation
	FIA_X509_EXT.2—X.509 certificate authentication
	FIA_X509_EXT.3—X.509 certificate requests
FMT: Security management	FMT_MOF.1/Functions—Management of security functions behavior
	FMT_MOF.1/ManualUpdate—Management of security functions behavior
	FMT_MTD.1/CoreData—Management of TSF data
	FMT_MTD.1/CryptoKeys—Management of TSF data
	FMT_SMF.1—Specification of management functions

Requirement Class	Requirement Component
	FMT_SMR.2—Restrictions on security roles
FPT: Protection of the TSF	FPT_APW_EXT.1—Protection of administrator passwords
	FPT_SKP_EXT.1—Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
	FPT_STM_EXT.1—Reliable time stamps
	FPT_TST_EXT.1—TSF testing
	FPT_TUD_EXT.1—Trusted update
FTA: TOE access	FTA_SSL_EXT.1—TSF-initiated session locking
	FTA_SSL.3—TSF-initiated termination
	FTA_SSL.4—User-initiated termination
	FTA_TAB.1—Default TOE access banners
FTP: Trusted path/channels	FTP_ITC.1—Inter-TSF trusted channel
	FTP_TRP.1/Admin—Trusted path

4.2.1 Security Audit (FAU)

4.2.1.1 Audit Data Generation (FAU_GEN.1)

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

a) Start-up and shut-down of the audit functions;
b) All auditable events for the not specified level of audit; and
c) All administrative actions comprising:
   - Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).
   - Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).
   - Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).
   - Resetting passwords (name of related user account shall be logged).
   - [***no other actions***];
d) Specifically defined auditable events listed in Table **5**.

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, 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 cPP/ST, information specified in column three of Table **5**.

Table 5: Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FAU_STG.1	None.	None.
FAU_STG_EXT.1	None.	None.
FAU_STG_EXT.2/LocSpace	None.	None.
FAU_STG_EXT.3/LocSpace	Low storage space for audit events.	None.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure
FCS_NTP_EXT.1	Configuration of a new time server Removal of configured time server	Identity of new/removed time server
FCS_RBG_EXT.1	None.	None.
FCS_TLSC_EXT.1	Failure to establish a TLS session	Reason for failure
FCS_TLSC_EXT.2	None	None
FCS_TLSS_EXT.1	Failure to establish a TLS session	Reason for failure
FCS_TLSS_EXT.2	Failure to authenticate the client	Reason for failure
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of the identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of the identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.2	None.	None.
FIA_X509_EXT.3	None.	None.
FMT_MOF.1/Functions	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update.	None.
FMT_MTD.1/CoreData	None.	None.
FMT_MTD.1/CryptoKeys	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time – either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1).	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.

4.2.1.2 User Identity Association (FAU_GEN.2)

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.

4.2.1.3 Protected Audit Trail Storage (FAU_STG.1)

FAU_STG.1.1 The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2 The TSF shall be able to prevent unauthorised modifications to the stored audit records in the audit trail.

4.2.1.4 Protected Audit Event Storage (FAU_STG_EXT.1)

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

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

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

FAU_STG_EXT.1.3 The TSF shall [***drop new audit data, overwrite previous audit records according to the following rule: [<u>overwrite the oldest audit record with the newly generated audit record</u>]***] when the local storage space for audit data is full.

Application Note: *By default, the TOE will overwrite the oldest locally stored audit record with the newly generated audit record. If the Security Administrator disables this option, the TOE drops all new records.*

4.2.1.5 Counting Lost Audit Data (FAU_STG_EXT.2/LocSpace)

FAU_STG_EXT.2.1/LocSpace The TSF shall provide information about the number of [***dropped, overwritten***] audit records in the case where the local storage has been filled and the TSF takes one of the actions defined in FAU_STG_EXT.1.3.

4.2.1.6 Action in Case of Possible Audit Data Loss (FAU_STG_EXT.3/LocSpace)

FAU_STG_EXT.3.1/LocSpace The TSF shall generate a warning to inform the Administrator before the audit trail exceeds the local audit trail storage capacity.

4.2.2 Cryptographic Support (FCS)

4.2.2.1 Cryptographic Key Generation (FCS_CKM.1)

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

- ***RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;***
- ***ECC schemes using 'NIST curves' [P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;***

].

4.2.2.2 Cryptographic Key Establishment (FCS_CKM.2)

FCS_CKM.2.1 The TSF shall perform cryptographic key establishment in accordance with a specified cryptographic key establishment method: [

- ***Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***

].

4.2.2.3 Cryptographic Key Destruction (FCS_CKM.4)

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method

- For plaintext keys in volatile storage, the destruction shall be executed by a [***single overwrite consisting of [zeroes]***];

- For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that [
    - ***logically addresses the storage location of the key and performs a [single-pass] overwrite consisting of [a new value of the key]***]

that meets the following: No Standard.

Application Note: *The TOE does not store plaintext keys in non-volatile storage, so this part of the requirement is vacuously met.*

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

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

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

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

- ***RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [<u>2048 bits and 4096 bits</u>]***
- ***Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [<u>256, 384, and 521 bits</u>]***

]

that meet the following: [

- ***For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3***
- ***For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4***

].

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

FCS_COP.1.1/Hash The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm [***SHA-1, SHA-256, SHA-384, SHA-512***] and message digest sizes [***160, 256, 384, 512***] bits that meet the following: ISO/IEC 10118-3:2004.

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

FCS_COP.1.1/KeyedHash The TSF shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm [***HMAC-SHA-256, HMAC-SHA-384***] and cryptographic key sizes [**16-512 bits in 8 bit increments**] and message digest sizes [***256, 384***] bits that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

4.2.2.8 HTTPS Protocol (FCS_HTTPS_EXT.1)

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 If a peer certificate is presented, the TSF shall [***not establish the connection***] if the peer certificate is deemed invalid.

4.2.2.9 NTP Protocol (FCS_NTP_EXT.1)

FCS_NTP_EXT.1.1 The TSF shall use only the following NTP version(s) [***NTP v4 (RFC 5905)***].

FCS_NTP_EXT.1.2 The TSF shall update its system time using [***Authentication using [SHA1, SHA256, SHA384, SHA512] as the message digest algorithm(s)***].

FCS_NTP_EXT.1.3 The TSF shall not update NTP timestamp from broadcast and/or multicast addresses.

FCS_NTP_EXT.1.4 The TSF shall support configuration of at least three (3) NTP time sources.

4.2.2.10 TLS Client Protocol without Mutual Authentication (FCS_TLSC_EXT.1)

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

- ***TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***

] and no other ciphersuites.

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches [***the reference identifier per RFC 6125 section 6, IPv4 address in CN or SAN***].

FCS_TLSC_EXT.1.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [***Not implement any administrator override mechanism***].

FCS_TLSC_EXT.1.4 The TSF shall [***present the Supported Elliptic Curves/Supported Groups Extension with the following curves/groups: [secp256r1, secp384r1, secp521r1] and no other curves/groups***] in the Client Hello.

4.2.2.11 TLS Client Support for Mutual Authentication (FCS_TLSC_EXT.2)

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

4.2.2.12 TLS Server Protocol without Mutual Authentication (FCS_TLSS_EXT.1)

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

- ***TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289***

].

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [***TLS 1.1***].

FCS_TLSS_EXT.1.3 The TSF shall perform key establishment for TLS using [***ECDHE curves [secp256r1, secp384r1, secp521r1] and no other curves***].

FCS_TLSS_EXT.1.4 The TSF shall support [***no session resumption or session tickets***].

4.2.2.13 TLS Server Support for Mutual Authentication (FCS_TLSS_EXT.2)

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

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

- ***Not implement any administrative override mechanism***

].

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

4.2.2.14 Random Bit Generation (FCS_RBG_EXT.1)

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

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

4.2.3 Identification and Authentication (FIA)

4.2.3.1 Authentication Failure Management (FIA_AFL.1)

FIA_AFL.1.1 The TSF shall detect when an Administrator configurable positive integer within [**1-100**] unsuccessful authentication attempts occur related to Administrators attempting to authenticate remotely using a password.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall [***prevent the offending Administrator from successfully establishing remote session using any authentication method that involves a password until [<u>action to enable the Administrator account</u>] is taken by an Administrator***].

4.2.3.2 Password Management (FIA_PMG_EXT.1)

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

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [***"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [<u>"_", "|", "<", ">", "?", ",", ".", "~"</u>]***];
b) Minimum password length shall be configurable to between [**8**] and [**32**] characters.

4.2.3.3 User Identification and Authentication (FIA_UIA_EXT.1)

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

- Display the warning banner in accordance with FTA_TAB.1;
- [***<u>[respond to ICMP echo request packets, if enabled]</u>***].

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

4.2.3.4 Password-Based Authentication Mechanism (FIA_UAU_EXT.2)

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

4.2.3.5 Protected Authentication Feedback (FIA_UAU.7)

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

4.2.3.6 X.509 Certificate Validation (FIA_X509_EXT.1/Rev)

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

- RFC 5280 certificate validation and certificate path validation supporting a minimum path length of three certificates.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The TSF shall validate the revocation status of the certificate using [***the Online Certificate Status Protocol (OCSP) as specified in RFC 6960***].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
  - Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
  - Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
  - Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.

- ○ OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

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

4.2.3.7 X.509 Certificate Authentication (FIA_X509_EXT.2)

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

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

4.2.3.8 X.509 Certificate Requests (FIA_X509_EXT.3)

FIA_X509_EXT.3.1 The TSF shall generate a Certificate Request as specified by RFC 2986 and be able to provide the following information in the request: public key and [***device-specific information***, ***Common Name***, ***Organization***, ***Organizational Unit***, ***Country***].

FIA_X509_EXT.3.2 The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

4.2.4 Security Management (FMT)

4.2.4.1 Management of Security Functions Behavior (FMT_MOF.1/ManualUpdate)

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

4.2.4.2 Management of Security Functions Behavior (FMT_MOF.1/Functions)

FMT_MOF.1.1/Functions The TSF shall restrict the ability to [***determine the behaviour of***, ***modify the behaviour of***] the functions [***handling of audit data, audit functionality when Local Audit Storage is full***] to Security Administrators.

4.2.4.3 Management of TSF Data (FMT_MTD.1/CoreData)

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

4.2.4.4 Management of TSF Data (FMT_MTD.1/CryptoKeys)

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

4.2.4.5 Specification of Management Functions (FMT_SMF.1)

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

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;

- Ability to update the TOE, and to verify the updates using [***digital signature***] capability prior to installing those updates;
- Ability to configure the authentication failure parameters for FIA_AFL.1;
- [
    - ***Ability to configure audit behavior (e.g. changes to storage locations for audit; changes to behavior when local audit storage space is full)***;
    - ***Ability to configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1***;
    - ***Ability to manage the cryptographic keys***;
    - ***Ability to re-enable an Administrator account***;
    - ***Ability to set the time which is used for time-stamps***;
    - ***Ability to configure NTP***;
    - ***Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors***;
    - ***Ability to import X.509v3 certificates to the TOE's trust store***;

].

4.2.4.6 Restrictions on Security Roles (FMT_SMR.2)

FMT_SMR.2.1 The TSF shall maintain the roles:

- Security Administrator.

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

FMT_SMR.2.3 The TSF shall ensure that the conditions

- The Security Administrator role shall be able to administer the TOE locally
- The Security Administrator role shall be able to administer the TOE remotely

are satisfied.

4.2.5 Protection of the TSF (FPT)

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

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

4.2.5.2 Protection of Administrator Passwords (FPT_APW_EXT.1)

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

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

4.2.5.3 Reliable Time Stamps (FPT_STM_EXT.1)

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

FPT_STM_EXT.1.2 The TSF shall [***allow the Security Administrator to set the time, synchronise time with an NTP server***].

4.2.5.4 Trusted Update (FPT_TUD_EXT.1)

FPT_TUD_EXT.1.1 The TSF shall provide Security Administrators the ability to query the currently executing version of the TOE firmware/software and [***the most recently installed version of the TOE firmware/software***].

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

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

4.2.5.5 TSF Testing (FPT_TST_EXT.1)

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [***during initial start-up (on power on)***] to demonstrate the correct operation of the TSF: [

- **Firmware integrity verification**
- **RAM March Test**
- **Error-correcting Code Memory Test**
- **Special Purpose Register Test**
- **General Purpose Register Test**
- **Condition Register Test**
- **Branch Test**
- **Integer Math Test**
- **Integer Load and Store Test**
- **Floating Point Unit Register Test**
- **Floating Point Unit Load and Store Test**
- **Floating Point Unit Math Test**
- **Timebase and Decrementer Test**
- **Data Cache Test**
- **RNG Test**
- **AES CBC Test**
- **AES GCM Test**
- **Sign and Verification Test**
- **SHA Test**
- **HMAC Test**

].

4.2.6 TOE Access (FTA)

4.2.6.1 TSF-Initiated Session Locking (FTA_SSL_EXT.1)

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

4.2.6.2 TSF-Initiated Termination (FTA_SSL.3)

FTA_SSL.3.1 The TSF shall terminate a remote interactive session after a Security Administrator-configurable time interval of session inactivity.

4.2.6.3 User-Initiated Termination (FTA_SSL.4)

FTA_SSL.4.1 The TSF shall allow Administrator-initiated termination of the Administrator's own interactive session.

4.2.6.4 Default TOE Access Banners (FTA_TAB.1)

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

4.2.7 Trusted Path/Channels (FTP)

4.2.7.1 Inter-TSF Trusted Channel (FTP_ITC.1)

FTP_ITC.1.1 The TSF shall be capable of using [***TLS***] to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: audit server, [***no other capabilities***] that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from disclosure and detection of modification of the channel data.

FTP_ITC.1.2 The TSF shall permit the TSF or the authorized IT entities to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [**export of audit records to external audit server**].

4.2.7.2 Trusted Path (FTP_TRP.1/Admin)

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

FTP_TRP.1.2/Admin The TSF shall permit remote Administrators to initiate communication via the trusted path.

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

4.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference from the [cPPND].

Table 6: Assurance Components

Requirement Class	Requirement Component
ASE: Security Target	ASE_CCL.1—Conformance claims
	ASE_ECD.1—Extended components definition
	ASE_INT.1—ST introduction
	ASE_OBJ.1—Security objectives for the operational environment
	ASE_REQ.1—Stated security requirements
	ASE_SPD.1—Security Problem Definition
	ASE_TSS.1—TOE summary specification
ADV: Development	ADV_FSP.1—Basic functional specification
AGD: Guidance documents	AGD_OPE.1—Operational user guidance
	AGD_PRE.1—Preparative procedures
ALC: Life-cycle support	ALC_CMC.1—Labelling of the TOE
	ALC_CMS.1—TOE CM coverage
ATE: Tests	ATE_IND.1—Independent testing – conformance
AVA: Vulnerability assessment	AVA_VAN.1—Vulnerability survey

5 TOE Summary Specification

This section describes the following security functions implemented by the TOE to satisfy the SFRs claimed in Section 4.2:

- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted Path/Channels.

5.1 Security Audit

5.1.1 Audit Data Generation

The TOE generates audit records of the following events:

- Start-up and shut-down of the audit function
- Administrative login and logout—all use of the identification and authentication mechanism, including the origin
- All management activities of TSF data, including changes to TSF data related to configuration changes and what was changed
- Generating/import of, changing, or deleting of cryptographic keys—the TOE logs the specific administrator action that was performed and identifies cryptographic keys by identifying the certificate associated with the key, using the certificate subject identifier and certificate issuer identifier
- Resetting passwords, including identification of the relevant user account
- Any failure to establish an HTTPS or TLS session, including the reason for the failure
- Configuration and removal of NTP server, including the identity of the server
- The number of consecutive failed authentication attempts has been reached, including the origin of the failed attempts
- Unsuccessful attempts to validate an X.509 certificate, including the reason for failure
- Addition, replacement and removal of a certificate in the trust store, including identification of the certificate
- Initiation of a manual update of the TOE and the result (success, failure) of the update
- Discontinuous changes to the system time, including both the old and new values for the time, and the origin of the attempt to change the time (both success and failure)
- Termination of interactive sessions by the session locking mechanism
- Termination by a user of their own interactive session
- Initiation and termination of trusted channels and failure of trusted channel functions, including initiator and target of failed initiation attempts
- Initiation, termination and failure of trusted path functions.

Each audit record generated by the TOE includes the following information, at a minimum: date and time stamp of when the auditable event occurred; type of event; identity of the subject that initiated the auditable event; and the outcome of the event (e.g., success or failure). Audit records of events resulting from the actions of identified users include the relevant user identity.

This aspect of the Security Audit security function satisfies FAU_GEN.1 and FAU_GEN.2.

5.1.2 Audit Storage and Audit Record Export

The TOE is a single standalone appliance that is able to store generated audit records locally on the appliance. The logs comprising the audit trail are stored in the TOE's file system and protected from unauthorized modification and deletion by file system permissions.

The local audit storage size is configurable from 500MB to 2GB. The TOE reports audit log warning messages when the local storage has been reduced to 25%, 15%, 10%, 5%, 4%, 3%, 2%, and 1% of available storage space. The warning messages are written to the audit log.

Users with the Security Admin or Network Admin role can enable and disable generation of audit records and can configure the behavior of the TOE when local audit storage is full. By default, the TOE will overwrite the oldest locally stored audit record with the newly generated audit record and keep a count of the number of records the TOE has overwritten. The TOE displays the overwritten counter value as a warning whenever an administrator invokes the `viewlog` command. If an administrator disables this option, the TOE drops all new records and keeps a counter of the audit records dropped when the local storage is full. There are no situations in which lost audit data is not counted. The Security Admin is able to view the count of dropped audit records and clear local audit storage. There are two methods to clear the local audit storage—by purging the entire local audit log, or by removing a subset of the local log data. In both cases, clearing local audit storage resets the counters of overwritten and dropped audit records to 0.

The TOE can be configured to export audit records to an external audit server over a trusted channel protected by TLS. In this circumstance, the TOE acts as a TLS client. The audit records are exported in real time (i.e., as they are generated).

This aspect of the Security Audit security function satisfies FAU_STG.1, FAU_STG_EXT.1, FAU_STG_EXT.2/LocSpace, and FAU_STG_EXT.3/LocSpace.

5.2 Cryptographic Support

The TOE incorporates Guardtime Federal's Cryptographic Support Library (CSL) Direct v2.0.0 cryptographic module.

5.2.1 Cryptographic Operations

The TOE includes NIST-validated cryptographic algorithms providing supporting cryptographic functions. The following functions implemented by the CSL Direct cryptomodule included in the TOE have been certified in accordance with the identified standards.

Table 7: Cryptographic Functions Implemented by CSL Direct v2.0.0

Functions	Standards	Certificates
Asymmetric Key Generation (FCS_CKM.1)		
RSA (2048, 4096 bits)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3	A1515 RSA KeyGen (FIPS 186-4)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4	A1515 ECDSA KeyGen (FIPS186-4)
Key establishment (FCS_CKM.2)		
Elliptic curve-based scheme	NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”	A1515 KAS-ECC-SSC Sp800-56Ar3
Data encryption (FCS_COP.1/DataEncryption)		
AES in GCM mode (256 bits)	ISO 18033-3 (AES) ISO 19772 (GCM mode)	A1515 AES-GCM
Digital signature generation and verification (FCS_COP.1/SigGen)		
RSA (2048, 4096 bit modulus)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSAPKCS2v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A1515 RSA SigGen (FIPS 186-4) A1515 RSA SigVer (FIPS 186-4)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 6 and Appendix D, Implementing “NIST curves” [P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4	A1515 ECDSA SigGen (FIPS 186-4) A1515 ECDSA SigVer (FIPS 186-4)
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A1515 SHA-1 A1515 SHA2-256 A1515 SHA2-384 A1515 SHA2-512
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-256 (key sizes: 16-256 bits in 8 bit increments, digest size 256 bits) HMAC-SHA-384 (key sizes: 16-512 bits in 8 bit increments, digest size 384 bits)	ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”	A1515 HMAC-SHA2-256 A1515 HMAC-SHA2-384

Functions	Standards	Certificates
Deterministic random bit generation (FCS_RBG_EXT.1)		
Counter DRBG (AES)	ISO/IEC 18031:2011	A1515 Counter DRBG

The TOE performs AES encryption and decryption in accordance with ISO 18033-3, with key size of 256 bits, in the GCM mode of operation as specified in ISO 19772.

The TOE provides cryptographic signature services using the following algorithms:

- RSA Digital Signature Algorithm with key size (modulus) of 2048 or 4096 bits, in accordance with FIPS 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Scheme RSASSA-PKCS1v1.5.
- Elliptic Curve Digital Signature Algorithm with key sizes of 256, 384, or 521 bits, in accordance with FIPS 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, implementing NIST curves P-256, P-384, and P-521.

The TOE is able to perform cryptographic hashing services using SHA-1, SHA-256, SHA-384, and SHA-512, in accordance with ISO/IEC 10118-3:2004. The TOE uses the SHA hash algorithms as follows:

- as part of the HMAC algorithm that provides data integrity for TLS (SHA-384)
- as part of the HMAC algorithm that supports Password-Based Key Derivation Function 2 (PBKDF2) used when storing authentication credentials (SHA-256)
- as part of RSA and ECDSA digital signature generation and verification (SHA-256, SHA-384, SHA-512)
- for NTP authentication (SHA-1, SHA-256, SHA-384, SHA-512).

The TOE performs keyed-hash message authentication in accordance with ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2". The following table summarizes the hash function, key length, block size, and output MAC lengths used by the HMAC function. The TOE uses HMAC-SHA-384 in support of TLS and HMAC-SHA-256 in support of PBKDF.

Table 8: HMAC Function Values

Hash Function	Key Length	Block Size	Output MAC Length
SHA-256	16-256 bits in 8 bit increments	512 bits	256 bits
SHA-384	16-512 bits in 8 bit increments	1024 bits	384 bits

This aspect of the Cryptographic Support security function satisfies FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, FCS_COP.1/KeyedHash.

5.2.2 Random Bit Generation

The TOE implements a NIST-approved AES-CTR Deterministic Random Bit Generator (DRBG), as specified in ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions". The implementation uses one platform-based entropy source, which accumulates entropy from one hardware-based noise source, which has a minimum 256 bits of entropy.

This aspect of the Cryptographic Support security function satisfies FCS_RBG_EXT.1.

5.2.3 Cryptographic Key Generation and Establishment

The TOE is able to generate RSA asymmetric key pairs with cryptographic key size (modulus) of 2048 bits and 4096 bits, in accordance with Appendix B.3 of FIPS PUB 186-4, “Digital Signature Standard (DSS)”. The TOE uses RSA keys in support of TLS client and TLS server authentication.

The TOE is able to generate ECC asymmetric key pairs with key sizes of 256, 384, and 521 bits over NIST curves P-256, P-384, and P-521 respectively, in accordance with Appendix B.4 of FIPS PUB 186-4, “Digital Signature Standard (DSS)”. The TOE uses ECC keys in support of TLS key establishment and TLS client and TLS server authentication.

The TOE acts as both the client and the server for TLS key establishment schemes. It acts as a client for export of audit records to an external audit server, and as a server for remote administration (supporting HTTPS) via the RESTful API.

The following table summarizes the key establishment schemes implemented by the TOE, the relevant protocol SFR for each scheme, and the TOE services associated with each scheme.

Table 9: Key Establishment Scheme Usage by TOE

Scheme	SFR	Service
ECDH	FCS_TLSC_EXT.1	Audit server
ECDH	FCS_TLSS_EXT.1	Administration

This aspect of the Cryptographic Support security function satisfies FCS_CKM.1 and FCS_CKM.2.

5.2.4 Cryptographic Key Destruction

The TOE uses the following secret keys, private keys, and critical security parameters (CSPs).

Table 10: Private Keys, Symmetric Keys, and CSPs

Key/CSP	Origin, Use and Storage
TOE root key (256 bits)	Derived from a hardware-based secret (one-time programmable master key, or OTPMK). Used to encrypt TOE key. Stored in RAM.
TOE key (256 bits)	Generated by TOE. Used to encrypt logs, configurations database, user database, TOE-generated files. Stored on SSD, encrypted using 256 bit AES.
RSA private key (2048 or 4096 bit modulus)	Generated by TOE. Used to authenticate the TOE in TLS sessions. Stored on SSD, encrypted using 256 bit AES.
ECDSA private key (P-256, P-384, or P-521 curve)	Generated by TOE. Used to authenticate the TOE in TLS sessions. Stored on SSD, encrypted using 256 bit AES.
ECDSA random ephemeral secret	Generated by TOE. Used in TLS key exchange. Stored in RAM.
EC DH ephemeral private key (P-256, P-384, P-521 NIST curves)	Generated by TOE. Used in TLS key exchange. Stored in RAM.

Key/CSP	Origin, Use and Storage
TLS shared ECDHE secret	Generated by TOE. Used in TLS. Stored in RAM.
TLS pre-master secret	Generated by TOE. Used in TLS. Stored in RAM.
TLS master secret	Generated by TOE. Used in TLS. Stored in RAM.
TLS record layer AES and HMAC keys	Generated by TOE. Used to protect (AES) and verify integrity of (HMAC) data communicated in TLS sessions. Stored in RAM.
DRBG parameters (seed, entropy input)	Generated by TOE. Used to instantiate DRBG. Stored in RAM.

The TOE does not store plaintext keys in non-volatile memory—all keys are encrypted at rest using 256 bit AES. As such, that part of FCS_CKM.4 is vacuously met.

For keys stored in or decrypted into volatile memory, once the keys are no longer needed, the TOE deallocates the memory back to the kernel. The memory is zeroized when power is removed from the TOE. There are no configurations or circumstances that do not conform to the key destruction requirement.

The root or top-level key-encrypting key is also an AES 256 bit key, derived from a special hardware-based secret value called the OTPMK (one time programmable master key). The OTPMK is implemented in specially-designed circuitry by the chip manufacturer. The vendor uses this key value to protect long-term keys stored on the TOE at rest.

This aspect of the Cryptographic Support security function satisfies FCS_CKM.4.

5.2.5 Cryptographic Protocols

The TOE implements the following cryptographic protocols to protect communications between itself and non-TOE entities:

- TLS as a client—the TOE acts as a TLS client when exporting audit records to an external audit server
- TLS as a server—the TOE acts as a TLS server supporting remote management access via the RESTful API
- HTTPS—in conjunction with TLS, the TOE supports the use of HTTPS for remote management access via the RESTful API
- NTP—the TOE can synchronize its system clock with an NTP server.

5.2.5.1 TLS Client Protocol

The TOE's TLS client implementation supports TLS communication with mutual authentication using X.509v3 certificates. The TOE supports TLS v1.2 only and the following TLS cipher suites when acting as a TLS client:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

The TOE supports reference identifiers per RFC 6125 section 6, and also supports IPv4 addresses in the CN or SAN field. The TOE supports the use of wildcards in each of these cases. The TOE enforces canonical format for IPv4 addresses in accordance with RFC 3986. When comparing an IPv4 address in the CN field to the reference identifier, the TOE converts the text representation of the IP address to a binary representation in network byte order.

An administrator (Security Admin or Network Admin) configures reference identifiers for the external audit server by configuring the applicable server hostname or IPv4 address parameters. The TOE supports certificate pinning by allowing the Security Admin to import Intermediate and Root Certificate Authority (CA) certificates and associating them with a predefined host.

The TOE presents the Supported Elliptic Curves Extension in its Client Hello message. By default, the Supported Elliptic Curves Extension specifies the following NIST curves: secp256r1; secp384r1; and secp521r1.

This aspect of the Cryptographic Support security function satisfies FCS_TLSC_EXT.1 and FCS_TLSC_EXT.2.

5.2.5.2 TLS Server Protocol

The TOE's TLS server implementation supports TLS v1.2 only and the following TLS cipher suites when acting as a TLS server:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

The TOE accepts ClientHello messages that specify support for TLS v1.2 and rejects ClientHello messages that do not specify support for TLS v1.2.

When the TOE negotiates a cipher suite, it sends a Server Key Exchange message that specifies the supported NIST curve, the ECDH public key, and the associated domain parameters. The TOE can perform key establishment using the secp256r1, secp384r1, and secp521r1 ECDHE curves.

The TOE's TLS server implementation supports TLS communication with mutual authentication of TLS clients using X.509v3 certificates. If the client certificate is invalid, The TOE will not establish a TLS connection. The TOE does not support any fallback authentication functions and does not implement any administrative override. The expected identifier of a client is configured into the TOE as a remote client configuration parameter. The TOE supports Fully Qualified Domain name (FQDN) and IPv4 address identifiers. The TOE matches FQDN identifiers according to RFC 6125, whereas for IPv4 address identifiers, the TOE performs a bit-wise comparison. During a TLS connection attempt, the TOE compares the expected identifier with the identifier in the client certificate's Subject Alternative Name (SAN) field, if it is available; otherwise, it is compared to the Common Name (CN) field. If the expected identifier does not match the SAN or CN, then the connection is not established.

The TOE does not support session resumption or session tickets.

This aspect of the Cryptographic Support security function satisfies FCS_TLSS_EXT.1 and FCS_TLSS_EXT.2.

5.2.5.3 HTTPS

The TOE uses HTTPS to protect communications between itself and remote users accessing the TOE via its RESTful API. The TOE implements the HTTPS protocol according to RFC 2818 by using a TLS v1.2 session to secure the HTTP connection. The TOE performs mutual authentication and it expects the peer to present a valid certificate before establishing the connection.

This aspect of the Cryptographic Support security function satisfies FCS_HTTPS_EXT.1.

5.2.5.4 NTP Protocol

The TOE can synchronize its system clock with an NTP server. The TOE supports NTP v4 as defined in RFC 5905 and supports use of SHA-1, SHA-256, SHA-384, or SHA-512 as its means for authenticating the NTP timestamps it receives from configured NTP servers. The TOE can support up to ten NTP time sources and will not update NTP timestamps from broadcast or multicast addresses.

This aspect of the Cryptographic Support security function satisfies FCS_NTP_EXT.1.

5.3 Identification and Authentication

5.3.1 User Identification and Authentication

The TOE offers only the following services to external entities prior to identification and authentication:

- display the advisory notice and consent warning message prior to completing the establishment of an interactive user session
- respond to ICMP echo request (ping) packets, if enabled to do so.

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

The TOE implements a local password-based mechanism to authenticate administrators attempting to access the TOE using the SCI. In order to login at the SCI, the administrator enters a username and password. The TOE checks the credentials against its local user database. Once the TOE identifies and authenticates the administrator, the administrator will have access to the SCI's command line interface (CLI) that allows the administrator to manage the TOE. While the administrator enters the password at the SCI, the TOE does not display any characters in the password field. If the administrator enters an incorrect password for the claimed user ID, the TOE displays "Login failed. Please, try again". The TOE displays the same message if the user ID does not exist in the local user database.

The TOE additionally supports remote administrative access via a RESTful interface over an HTTPS channel. Each RESTful request (protected within the HTTPS channel) includes a username and password. The TOE checks these credentials against its local user database. If the TOE authenticates the user and the user possesses the necessary permissions, the TOE performs the management action included in the RESTful request. If the user is not authenticated (i.e., the provided password does not match the claimed user identity), or if the user does not have the necessary permission, the TOE rejects the RESTful request and generates an audit record.

A user with the Security Admin role enables remote administrative access by configuring a supplemental authentication server and specifying 'localhost', using the SCI command `setconfig --kp management.authserver localhost`.

This aspect of the Identification and Authentication security function satisfies FIA_UIA_EXT.1, FIA_UAU_EXT.2, and FIA_UAU.7.

5.3.2 Authentication Failure Management

The TOE implements a mechanism to respond to consecutive failures to authenticate remote management requests. This mechanism is disabled by default but is enabled when an administrator enables remote administrative access (by configuring 'localhost' as a supplemental authentication server, as described in Section 5.3.1). A user in the Security Admin role can use the `setconfig` SCI command or a RESTful API `PUT` method on the `config` endpoint to configure the number of consecutive failed remote authentication attempts. This number can be any integer between 1 and 100 inclusive and has a default value of 5.

As described in Section 5.3.1, the TOE validates credentials in the HTTPS header of RESTful requests against a local user account. When enabled, the TOE keeps a count of consecutive failed remote authentication attempts for each configured user. If the number of consecutive failed remote authentication attempts reaches the configured value for allowed failed attempts, the TOE disables the user account. Only a Security Admin can enable the account. Note that although all users are subject to lockout after consecutive failed remote authentication attempts, users with the Security Admin role can never be locked out of the SCI. All other users are subject to lockout at both the local and remote interfaces.

This aspect of the Identification and Authentication security function satisfies FIA_AFL.1.

5.3.3 Password Management

The TOE maintains a local database of user accounts with their corresponding passwords. Passwords can be composed of any combination of:

- Upper and lower case letters
- Numbers
- The following special characters: `!@#$%^&*()_|<>?,.~`.

The minimum password length in the TOE is configurable by a user with the Security Admin or Network Admin role and can be set to any integer value in the range 8 to 32. The default minimum password length is 8 and the maximum length supported for passwords is 32 characters.

This aspect of the Identification and Authentication security function satisfies FIA_PMG_EXT.1.

5.3.4 X.509 Certificates

The TOE performs RFC 5280 certificate validation and certificate path validation on all X.509 certificates presented to it in support of secure communication over TLS with:

- External audit servers
- Remote management using RESTful interface.

The TOE supports a path length of at least three certificates.

The TOE validates X.509 certificates using the path validation algorithm defined in RFC 5280, which can be summarized as follows:

- The public key algorithm and parameters are checked
- The current date/time is checked against the validity period of the certificate
- The revocation status is checked
- The issuer name is checked to ensure that it equals the subject name of the previous certificate in the path
- Name constraints are checked, to make sure the subject name is within the permitted subtrees list of all previous CA certificates and not within the excluded subtrees list of any previous CA certificate

- The asserted certificate policy OIDs are checked against the permissible OIDs of the previous certificate, including any policy mapping equivalencies asserted by the previous certificate
- Policy constraints and basic constraints are checked, to ensure that any explicit policy requirements are not violated and that the certificate is a CA certificate, respectively
- The path length is checked to ensure that it does not exceed any maximum path length asserted in this or a previous certificate
- The key usage extension is checked
- Any other critical extensions are recognized and processed.

The certificate chain is validated to the root, and a revocation check is performed on each certificate (except the root certificate) using Online Certificate Status Protocol (OCSP).

The TOE uses the following rules for validating the extendedKeyUsage[2] field:

- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field
- OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

The TOE will not allow any certificate to be imported that does not pass validation. The TOE uses OCSP as specified in RFC 6960 to check revocation status of certificates during the path validation process. When the TOE receives a certificate, during importing or PKI-based communication, it checks the Authority Information Access field to be configured with the OCSP flag as well as with a URI for the OCSP server. The URI can either be a hostname or an IP address. If the certificate does not have Authority Information Access field populated, then the TOE will bypass the OCSP check for that certificate. If the OCSP server does not respond or if the certificate is invalid, the TOE does not establish the connection.

A user with the Security Admin role is able to use the `gencsr` SCI command to generate a Certificate Signing Request (CSR) as specified in RFC 2986 and is able to specify the Common Name (CN), Organization, Organizational Unit, Country, and device-specific information such as DNS name and IP address that is stored in the Subject Alternative Name (SAN) list.

The TOE stores CSRs in PKCS#10 PEM format. The TOE displays the CSRs at the SCI for the Security Admin to transmit to a Certificate Authority (CA). The Security Admin performs this transmission manually. When the Security Admin receives a signed certificate back from the CA, the Security Admin imports the certificate, along with the CA chain, into the TOE localhost. CSRs generated by the TOE are required to be associated with a private/public key pair.

All the certificates are imported manually into the TOE. The TOE's localhost certificate is used for all PKI-based communication. The TOE allows the Security Admin to import the entire CA chain of (peer) remote hosts with which the TOE is expected to communicate over TLS channels. This is done for the purpose of authenticating the peer during the establishment of a TLS channel (dual authentication). In those

[2] Certificates are not used for trusted updates or executable code integrity. Therefore, the TOE does not support the rules for validating certificates with the Code Signing purpose in the extendedKeyUsage field, and this part of the requirement is trivially satisfied.

instances, the TOE receives the peer certificate and uses its trust store certificates (associated with the host) to validate the peer's certificate to its root CA.

This aspect of the Identification and Authentication security function satisfies FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, and FIA_X509_EXT.3.

5.4 Security Management

5.4.1 Security Roles

The TOE implements the following two default administrator roles that together provide the capabilities of the Security Administrator role defined in FMT_SMR.2:

- Security Admin—responsible for managing all security-related services and configuration of the TOE. The Security Admin role is able to: add administrative users and roles; modify the password policy; modify behavior of the audit function; manage cryptographic keys, CSRs, and certificates; perform firmware updates; and configure the TOE access banner
- Network Admin—responsible for managing all network-related services and configurations. The Network Admin is able to: view and modify network configuration; modify behavior of the audit function; set the system time; and configure route tables.

The TOE defines two additional roles that do not contribute to any capabilities required of the Security Administrator role:

- Application role—responsible for managing all KSI-related services and configuration
- Recovery-Agent—specific role associated with backing up and recovering the TOE root key.

This aspect of the Security Management security function satisfies FMT_SMR.2.

5.4.2 Specification of Management Functions

The TOE administrators have the ability to administer the TOE locally via the SCI and remotely via the RESTful interface. The following table lists TOE management functions, identifies the SCI command the TOE provides to invoke each function, and identifies the functions available via the RESTful interface.

Table 11: TOE Management Functions

Management Function	SCI Command	RESTful API Action
Configure the access banner	`banner`	n/a
Configure the session inactivity time before termination of the local administrative session	`setconfig`	`PUT` method on `config` endpoint
Update the TOE and verify TOE updates prior to installation using digital signature	`updatebl`	n/a
Configure the authentication failure parameters	`setconfig`	`PUT` method on `config` endpoint
Configure audit behavior (i.e., configure local log storage size, configure TOE behavior when local audit storage space is full)	`setconfig`	`PUT` method on `config` endpoint

Management Function	**SCI Command**	**RESTful API Action**
Configure the list of TOE-provided services available before an entity is identified and authenticated, per FIA_UIA_EXT.1	`setconfig`	`PUT` method on `config` endpoint
Manage cryptographic keys	`genkey`, `gencsr`, `rm`	n/a
Re-enable an administrator account	`moduser`	n/a
Set the time used for time stamps	`settime`	n/a
Configure NTP	`setconfig`	`PUT` method on `config` endpoint
Manage the TOE's trust store and designate X.509 v3 certificates as trust anchors	`import`	n/a
Import X.509 v3 certificates to the TOE's trust store	`import`	n/a

This aspect of the Security Management security function satisfies FMT_SMF.1.

5.4.3 Management of Security Functions Behavior

The TOE places restrictions on the ability to determine and modify the behavior of certain security management functions, as follows:

- Handling of audit data (i.e., configuring local log storage size)—users with the Security Admin or the Network Admin role can determine the behavior of this function (i.e., determine the configured space available for local storage of audit records) using the `getconfig` SCI command or via a RESTful API `GET` method on the `config` endpoint. Users in these roles can modify the behavior of this function (i.e., modify the configured space available for local storage of audit records) using the `setconfig` SCI command or via a RESTful API `PUT` method on the `config` endpoint.

- Audit functionality when Local Audit Storage is full—users with the Security Admin or the Network Admin role can determine the behavior of this function (i.e., determine if new audit records are dropped or if new audit records overwrite oldest stored audit record) using the `getconfig` SCI command or via a RESTful API `GET` method on the `config` endpoint. Users in these roles can modify the behavior of this function (i.e., specify if new audit records are dropped or if new audit records overwrite oldest stored audit record) using the `setconfig` SCI command or via a RESTful API `PUT` method on the `config` endpoint.

The ability to perform TOE updates is restricted to the Security Admin role.

This aspect of the Security Management security function satisfies FMT_MOF.1/Functions and FMT_MOF.1/ManualUpdate.

5.4.4 Management of TSF Data – Cryptographic Keys

Table 10 in Section 5.2.4 above lists the keys and CSPs used by the TOE. The TOE manages all of these keys and CSPs automatically without administrator involvement, with the exception of the RSA private kay and the ECDSA private key. A user with the Security Admin role is able to perform the following operations on these keys:

- Generate an RSA or ECDSA key pair using the `genkey` SCI command
- Associate the generated RSA or ECDSA key pair with an X.509v3 certificate using the `gencsr` SCI command to generate a CSR (see Section 5.3.4 above for further details)
- Import the signed X.509v3 certificate associated with the generated RSA or DCDSA key pair, along with the certificate chain up to and including the signing root CA, using the `import` SCI command (see Section 5.3.4 above for further details)
- Delete the generated RSA or ECDSA keys using the `rm` SCI command to remove the files containing the keys from the file system.

This aspect of the Security Management security function satisfies FMT_MTD.1/CryptoKeys.

5.4.5 Management of TSF Data – Core Data

The ability to manage TSF data is restricted to the Security Admin and Network Admin roles. For the Security Admin, this includes the ability to manage the TOE's trust store by uploading X.509 v3 certificates and CA certificates.

The following table lists the SCI commands TOE administrators can use to manage (i.e., create, view, modify, delete, clear, etc.) TSF data and identifies the roles permitted to invoke each command.

Table 12: Functions for Managing TSF Data

Command	Purpose	Roles
`addrole`	Adds a security management role to a user	Security Admin
`adduser`	Create new user account on TOE	Security Admin
`banner`	Configure TOE login banner	Security Admin
`delrole`	Remove a security management role from a user	Security Admin
`deluser`	Delete a user account from the TOE	Security Admin
`gencsr`	Generate certificate signing request	Security Admin
`genkey`	Generate a cryptographic key	Security Admin
`getcert`	View certificate information	Security Admin
`getconfig`	View TOE configuration information	Security Admin Network Admin
`gettime`	View the system time	Security Admin Network Admin
`getuser`	View information about TOE user accounts	Security Admin
`import`	Import configuration information; manage the TOE's trust store by uploading X.509 v3 certificates and CA certificates	Security Admin
`moduser`	Enable or disable user account	Security Admin

Command	Purpose	Roles
`passwd`	Change password on user account	Security Admin Network Admin
`rm`	Remove directories or files from TOE, including keys and local audit logs	Security Admin
`setconfig`	Modify TOE configuration information	Security Admin Network Admin
`settime`	Set the system time	Network Admin

Note, the Security Admin and Network Admin roles can also perform `getconfig` and `setconfig` functions via the RESTful interface, using `GET` and `PUT` methods on the appropriate endpoint.

This aspect of the Security Management security function satisfies FMT_MTD.1/CoreData.

5.5 Protection of the TSF

5.5.1 Protection of Administrator Passwords

The TOE protects administrator passwords using Password Based Key Derivation Function 2 (PBKDF2), as specified in NIST SP 800-132. The TOE's implementation of PBKDF2 uses HMAC-SHA-256 and a random salt generated by the TOE's Counter DRBG implementation to derive a pseudorandom value from the password that the TOE then stores, rather than storing the password itself or encrypting the password prior to storage. The TOE does not offer any functions that will disclose to any users a plaintext administrative password.

This aspect of the TSF Protection security function satisfies FPT_APW_EXT.1.

5.5.2 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)

The TOE relies on a top-level key-encrypting key, termed the root key, which is derived from a hardware-based secret called the one-time programmable master key (OTPMK). The OTPMK is implemented in specially-designed circuity by the chip manufacturer. The TOE uses the root key to protect long-term keys. All pre-shared, symmetric, and private keys that are stored in the TOE are encrypted using 256 bit AES in GCM mode, with the root key as the encryption key. The TOE protects such keys from unauthorized modification or substitution. When keys need to be used by the TOE, it decrypts the keys into volatile memory and does not provide an interface to view those keys. When the TOE is done using a plaintext key, the TOE deallocates the memory location back to the kernel.

This aspect of the TSF Protection security function satisfies FPT_SKP_EXT.1.

5.5.3 TSF Testing

The TOE performs primary and secondary firmware loads. These images are encrypted and signed while they are at rest. The keys to verify the digital signature and decrypt the firmware images are stored in the TOE. During initial start-up of the TOE, each firmware image is verified prior to executing its code, to ensure it has not been modified. If either of the images does not verify correctly, then the TOE reports this as a fault. If neither image verifies correctly, the TOE does not boot up.

In addition to performing verification and decryption of the firmware files, the extended boot also performs a series of Power On Self Tests (POSTs). The TOE performs the following POSTs:

- RAM March Test—verifies RAM by writing an incrementing integer value and a decrementing integer value into every memory cell. After each write, it reads the value back and compares it to the expected value.
- Error-correcting Code Memory Test—consists of two parts: single-bit error correction; and multi-bit error detection. For the single-bit error correction test, one bit error is injected into each DDR SRAM device. Once the single-bit error is injected, the test confirms the error is detected and corrected. Similarly for the multi-bit error detection test, a multi-bit error is injected into each DDR SRAM device and the test confirms the multi-bit error is detected (note, correction cannot be done on a multi-bit error).
- Special Purpose Register Test—verifies each bit can be read and written in the Special Purpose Registers (SPRs), with the exception of those bits that may put the processor into an unstable state. The SPRs verified by this test comprise the Core SPRs, Processor SPRs, Exception SPRs, Interrupt Vector SPRs, Configuration SPRs, and Debug SPRs. In addition to those SPRs that may put the processor into an unstable state, SPRs that are read-only, write-only, write-once and read-clear are also excluded from this test.
- General Purpose Register Test—verifies each bit in each general purpose register can be read and written correctly. The test starts by filling the general purpose register under test with 1s. Next, it loops to shift a 0 into each bit position and verifies the data read back matches the expected value.
- Condition Register Test—verifies the Condition Registers (CRs) are working correctly. It first sets certain bits in each CR and reads back the CR to verify that the bits were written. Next, it performs instructions that set bits in the CRs as a side effect and verifies the associated bits were set with the expected value.
- Branch Test—verifies the processor's branch instructions are functioning as expected. It performs each branch instruction and verifies the program counter ends up at the expected location. In addition, it also verifies the state of the link and counter registers for each branch instruction test. This test does not verify the branch absolute instruction (which branches to an absolute address).
- Integer Math Test—verifies every integer arithmetic instruction to make sure the instruction functions correctly. It executes every integer arithmetic instruction (including compares and shifts) using test patterns that utilize all the data paths and functional operations of the integer arithmetic unit.
- Integer Load and Store Test—verifies the load and store instructions are functioning as expected. It performs loads and stores between the General Purpose Registers and RAM using variations of load and store instructions with different data patterns.
- Floating Point Unit Register Test—verifies the functioning of the floating point registers. It writes multiple patterns to each floating point register and verifies the value read back matches the written pattern.
- Floating Point Unit Load and Store Test—verifies the floating point load and store instructions are working as expected. It uses the floating point registers as well as the floating point instructions to load and store data from/to memory.

- Floating Point Unit Math Test—verifies every floating point math instruction functions correctly. It executes every floating point math instruction and verifies that the floating point status register and the observed result value match the expected values.
- Timebase and Decrementer Test—verifies the Timebase and Decrementer Registers are functioning as expected. First, it writes and verifies each bit of the Timebase and Decrementer registers. Next, it checks for rollover in each bit of the Timebase and Decrementer registers. Finally, it measures the Timebase and Decrementer increment/decrement rate to make sure it is at the expected rate.
- Data Cache Test—verifies the Data Cache is functioning as expected. With memory coherency enabled, it verifies that the written test data pattern is flushed from the cache into memory when the data cache is invalidated. With memory coherency disabled, it verifies the written test data pattern is not written to memory.
- RNG Test—consists of four tests from FIPS 140: Monobit; Poker; Runs; and Repetition Count.
- AES CBC Test—verifies the hardware AES engine in CBC mode is functioning as expected. First it populates a test buffer with random data, and feeds the test buffer, along with a known Key and Initial Vector (IV), to the AES engine to be encrypted using CBC mode. Next, it verifies the AES engine can decrypt correctly by feeding it with the encrypted buffer received from the Encrypt operation with the same known Key and IV.
- AES GCM Test—verifies the hardware AES engine in Galois/Counter Mode (GCM) is functioning as expected. First, it populates a test buffer with random data, and feeds the test buffer, with a known Key and Initial Vector (IV), to the AES engine to be encrypted using GCM. Next, it verifies the AES engine can decrypt correctly by feeding it with the encrypted buffer received from the Encrypt operation with the same known Key and IV.
- Sign and Verification Test—verifies the RSA and ECDSA modes by signing data with a known key and then verifying that the correct signature was generated.
- SHA Test—verifies the hardware SHA engine is functioning as expected. It performs SHA-1, SHA-256, SHA-384 and SHA-512 on a sample data buffer and compares the returned hash with known hashes to make sure they are the same.
- HMAC Test—verifies the hardware HMAC engine is functioning as expected. It performs HMAC-SHA-256 and HMAC-SHA-384 on a sample data buffer and compares the returned results with known answers to make sure they are the same.

If all these tests pass, the overall status on the POST will indicate “OK”. Otherwise, any failed POST will trigger the TOE to display “Failure” on its “Power On Self Test” status. In addition, any errors encountered are printed out at bootup.

The combination of firmware verifications and POSTs performed during bootup, along with the approach taken, enables the TOE to convey to the Administrator that the TOE is operating properly. Furthermore, in the event of catastrophic failures, errors output during bootup will provide the Administrator adequate information to diagnose the issue.

This aspect of the TSF Protection security function satisfies FPT_TST_EXT.1.

5.5.4 Trusted Update

Users with the Security Admin or Network Admin role are able to query the current executing (i.e., primary) and most recently installed (i.e., secondary) versions of the TOE firmware using the `getconfig` SCI command. The two values should match since the TOE reboots following installation of a firmware update.

The TOE does not provide automatic updates to the firmware version running on the TOE. A user with the Security Admin role executes the `updatebl` SCI command to initiate the TOE firmware update process. As part of the command, the Security Admin specifies the location from which the TOE is to read the firmware update file. The location can be a URL or a local file path. The `updatebl` command additionally allows the Security Admin to download the firmware update file from a specified URL and store it locally on the TOE (in the `downloads` directory) without initiating the update process, which allows for updates to be performed offline. The TOE does not install firmware updates with a delayed activation—once the Security Admin initiates the firmware update, the process runs to completion.

The update process starts with the TOE requesting confirmation from the Security Admin to proceed with the update process. Upon confirmation, the TOE automatically proceeds with the rest of the update process without requiring further input from the Security Admin.

The vendor encrypts firmware update packages using 256 bit AES and digitally signs them using ECDSA with NIST curve P-521. The TOE verifies the digital signature and decrypts the image using keys pre-installed during manufacturing of the TOE. The TOE copies the firmware update package from the location specified in the `updatebl` command and proceeds to install the firmware in the primary location. The TOE then reboots. Coming up from reboot, the TOE verifies the firmware installed in the primary location. If verification succeeds, the TOE copies the primary location firmware into the secondary location, and reboots. If the primary location firmware verification fails, the TOE attempts to boot from the secondary location (which still retains the current version of firmware prior to the update attempt).

If any failures occur, including digital signature verification or decryption, the TOE aborts the firmware update process, outputs error messages to the SCI, and records relevant audit logs. In the event of a failed firmware update attempt, the Security Admin can use the information displayed by the SCI and the logs to troubleshoot the incident, and retry.

This aspect of the TSF Protection security function satisfies FPT_TUD_EXT.1.

5.5.5 Reliable Time Stamps

The TOE includes a real-time clock (RTC) within the TOE hardware. The TOE depends on the RTC to provide accurate date and time for the generated audit records and track inactivity of administrative local sessions. However, there exists inherent drift within the hardware, and therefore the TOE supports synchronization with external NTP servers. The TOE supports NTP v4 as specified in RFC 5905. The TOE authenticates the timestamps it receives from NTP servers using SHA-1, SHA-256, SHA-384, or SHA-512. The Security Admin or Network Admin can configure up to 10 external NTP servers. The TOE does not update its real-time clock based on timestamps received from broadcast or multicast addresses.

In addition, the Network Admin can set the system time directly using the `settime` SCI command.

This aspect of the TSF Protection security function satisfies FPT_STM_EXT.1.

5.6 TOE Access

The following methods of administrative access to the TOE are available:

- Local access to the SCI using a laptop directly connected to the TOE appliance via its RS-232 serial interface
- Remote access via the RESTful API. The RESTful API does not have a concept of interactive session since each request is self-contained and individually identified and authenticated.

5.6.1 Access Banner

The TOE displays a custom login banner prior to an administrative session via the SCI. The Security Admin is the only user role permitted to configure the banner in the TOE.

Note, the RESTful API is used for remote management of the TOE. The TOE does not maintain an interactive session over the RESTful API, so it does not display the login banner over this interface.

This aspect of the TOE Access security function satisfies FTA_TAB.1.

5.6.2 Session Termination

The TOE terminates local interactive sessions after a configurable period of inactivity. The Security Admin or Network Admin uses the `setconfig` SCI command to set the `serialsessiontimeout` configuration parameter, which specifies the inactivity timeout value in minutes. At the end of this period of inactivity on the local interactive session, the TOE terminates the session. The `serialsessiontimeout` parameter can take any integer value between 0 and 2,147,483,647, and has a default value of 300. If `serialsessiontimeout` is set to 0, session termination is disabled and the local interactive session will never timeout. This is not allowed in the evaluated configuration.

For remote management of the TOE, the RESTful API is used. The TOE does not maintain an interactive session over the RESTful API as each request is a self-contained, identified, and authenticated request. As such, the TOE does not establish an authenticated state that is preserved across multiple commands.

The TOE allows administrators to terminate their own local sessions using the `exit` command.

This aspect of the TOE Access security function satisfies FTA_SSL_EXT.1, FTA_SSL.3, and FTA_SSL.4.

5.7 Trusted Path/Channels

The TOE can be configured to export its audit records to an external syslog server over TLS. In this case, the TOE acts as a TLS client and initiates the connection to the syslog server. The TOE identifies and authenticates the syslog server by validating the syslog server's X.509 certificate that is presented during the TLS negotiation.

All remote administrative communication is performed via the RESTful API, which uses the HTTPS protocol with TLS and mutual authentication. An administrator uses a RESTful client to remotely manage the TOE by sending RESTful requests. Once the administrator is authenticated, the TOE processes the request and the HTTPS connection is terminated.

The Trusted Path/Channels security function satisfies FTP_ITC.1 and FTP_TRP.1/Admin.

6 Protection Profile Claims

This ST conforms to the collaborative Protection Profile for Network Devices, Version 2.2e, 23 March 2020 [cPPND] and includes the following optional and selection-based SFRs: FAU_STG.1; FAU_STG_EXT.2/LocSpace; FAU_STG_EXT.3/LocSpace; FCS_HTTPS_EXT.1; FCS_NTP_EXT.1; FCS_TLSC_EXT.1; FCS_TLSC_EXT.2; FCS_TLSS_EXT.1; FCS_TLSS_EXT.2; FIA_X509_EXT.1/Rev; FIA_X509_EXT.2; FIA_X509_EXT.3; FMT_MOF.1/Functions; and FMT_MTD.1/CryptoKeys.

As explained in Section 2, Security Problem Definition, the Security Problem Definition of the [cPPND] has been included by reference into this ST.

As explained in Section 3, Security Objectives, the ST reproduces the security objectives for the operational environment from [cPPND].

As explained in Section 4, IT Security Requirements, the SFRs have all been drawn from [cPPND]. As such, operations on SFRs already performed in that PP are not identified in this ST. Rather, the SFRs have been copied from [cPPND] and any formatting used in that PP has been removed. Operations performed on SFRs in the writing of this ST are identified in accordance with the conventions described in Section 1.3.

The SARs for the TOE are included by reference from the [cPPND].

7 Rationale

This ST includes by reference the [cPPND] Security Problem Definition and SARs and reproduces the security objectives for the Operational Environment. The ST makes no additions to the [cPPND] assumptions. [cPPND] SFRs have been reproduced with the Protection Profile operations completed. Operations on the SFRs follow [cPPND] application notes and assurance activities. Consequently, [cPPND] rationale applies but is incomplete. The TOE Summary Specification rationale below serves to complete the rationale required for the security target.

7.1 TOE Summary Specification Rationale

Each subsection in Section 5, the TOE Summary Specification (TSS), describes a security function of the TOE. Each description identifies the SFRs that are covered by that description and, as such, provides the rationale that indicates which requirements are satisfied by aspects of the corresponding security function. The security functions work together to satisfy all of the security functional requirements. Furthermore, all of the security functions are necessary in order for the TSF to provide the required security functionality.

This section, in conjunction with the TSS, provides evidence that the security functions are suitable to meet the TOE security requirements. The collection of security functions work together to provide all of the security requirements. The security functions described in the TSS are all necessary for the required security functionality in the TSF. Table 13: Security Functions vs. Requirements Mapping summarizes the relationship between security requirements and security functions.

Table 13: Security Functions vs. Requirements Mapping

Specification	Security audit	Cryptographic support	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
FAU_GEN.1	X						
FAU_GEN.2	X						
FAU_STG.1	X						
FAU_STG_EXT.1	X						
FAU_STG.2/LocSpace	X						
FAU_STG.3/LocSpace	X						
FCS_CKM.1		X					
FCS_CKM.2		X					
FCS_CKM.4		X					
FCS_COP.1/DataEncryption		X					

Specification	Security audit	Cryptographic support	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
FCS_COP.1/SigGen		X					
FCS_COP.1/Hash		X					
FCS_COP.1/KeyedHash		X					
FCS_HTTPS_EXT.1		X					
FCS_NTP_EXT.1		X					
FCS_RBG_EXT.1		X					
FCS_TLSC_EXT.1		X					
FCS_TLSC_EXT.2		X					
FCS_TLSS_EXT.1		X					
FCS_TLSS_EXT.2		X					
FIA_AFL_EXT.1			X				
FIA_PMG_EXT.1			X				
FIA_UIA_EXT.1			X				
FIA_UAU_EXT.2			X				
FIA_UAU.7			X				
FIA_X509_EXT.1/Rev			X				
FIA_X509_EXT.2			X				
FIA_X509_EXT.3			X				
FMT_MOF.1/Functions				X			
FMT_MOF.1/ManualUpdate				X			
FMT_MTD.1/CoreData				X			
FMT_MTD.1/CryptoKeys				X			
FMT_SMF.1				X			
FMT_SMR.2				X			
FPT_APW_EXT.1					X		
FPT_SKP_EXT.1					X		
FPT_TST_EXT.1					X		
FPT_TUD_EXT.1					X		
FPT_STM_EXT.1					X		

Specification	Security audit	Cryptographic support	Identification and authentication	Security management	Protection of the TSF	TOE access	Trusted path/channels
FTA_SSL_EXT.1						X	
FTA_SSL.3						X	
FTA_SSL.4						X	
FTA_TAB.1						X	
FTP_ITC.1							X
FTP_TRP.1/Admin							X