5151 California Ave
Suite 210
Irvine, CA 92617

Black Lantern

Security Target

Version 1.2

December 5, 2017

Table of Contents

1 ST Introduction (ASE_INT)

This section introduces the Target of Evaluation (TOE) and provides the Security Target (ST) and TOE identification, ST conventions, glossary and list of abbreviations. This Security Target is composed of the following sections:

- Section 1: Security Target Introduction
- Section 2: Conformance Claims
- Section 3: Security Problem Definition
- Section 4: Security Objectives
- Section 5: IT Security Requirements
- Section 6: TOE Summary Specification

The structure and content of this ST comply with the requirements specified in the Common Criteria (CC), Part 1, Annex A, and Part 2.

1.1 ST Reference and TOE Reference

Table 1: ST, TOE, and PP Identification

Item	Description
ST Title	Black Lantern Security Target
ST Version	1.2
ST Publication Date	12/05/2017
ST Author	Guardtime USA
TOE Reference	Black Lantern
TOE Hardware Model	BL300-B2, BL300-C2
TOE Software Version	1.5.2
TOE Developer	Guardtime USA
PP Identification	Collaborative Protection Profile for Network Devices, Version 1.0, data February 27, 2015 [NDcPPv1.0]

1.2 TOE Overview

The Target of Evaluation (TOE) is Guardtime's Black Lantern. The Black Lantern is a secure network device appliance that is an integrated hardware and software platform purposely built to mitigate both remote and physical attacks against a customer infrastructure and applications. The Black Lantern comes with a built-in KSI gateway and extender, which allows for secure

implementation of KSI-based data assurance and cybersecurity solutions with built-in active anti-tamper measures.

Black Lantern extends the power of the Keyless Signature Infrastructure (KSI) Industrial Blockchain for real-time cybersecurity and data-centric asset protection, directly supporting enhanced continuity of operations, data loss prevention, and is a new form of real-time Advanced Persistent Threat (APT) detection. KSI is designed to provide scalable digital signature-based authentication for electronic data, machines and humans.

1.3 TOE Description

The Black Lantern device comprises specialized security appliance hardware with advanced anti-tamper functionality, a software suite for real-time monitoring of the integrity state of the network, and a service that provides KSI signatures needed for network instrumentation.

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.

The benefits of the KSI are as follows:

- Massive Scale—the KSI signatures can be generated at exabyte-scale. Even if an exabyte (1,000 petabytes) of data is generated around the planet every second, every data record (a trillion records assuming 1MB average size) can be signed using KSI with negligible computational, storage and network overhead.
- Portability—the properties of the signed data can be verified even after that data has crossed geographic or organizational boundaries and service providers.
- Quantum Immunity—the cryptography behind the KSI signatures ensures that they never expire and remain quantum-immune, i.e., secure even after the realization of quantum computation.
- Independent Verification—the properties of the signed data can be verified without reliance or need for a trusted authority.
- Data Privacy—KSI does not ingest any customer data; data never leaves the customer premises. Instead the system is based on one-way cryptographic hash functions that result in hash values uniquely representing the data, but are irreversible such that one cannot start with the hash value and reconstruct the data—data privacy is guaranteed at all times.

1.3.1 TOE Architecture

The Black Lantern device (models BL300-B2 and BL300-C2) is a hardened hardware appliance consisting of the following major components:

- An NXP (formerly Freescale) QorIQ Power Architecture-based microprocessor
- Green Hills Software secure real time operating system
- Black Lantern Platform Manager software
- KSI software

Internally the Black Lantern is specified as follows:

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

1.3.1.1 Physical Scope of the TOE

Physically, the TOE is a 2U rack mount device. The following Figure 1 provides a graphical representation of the operational environment of the TOE. The figure depicts the connectivity of the KSI application entities (not applicable to NDcPP) as well as the IT entities that enable the compliance with the NDcPP requirements.

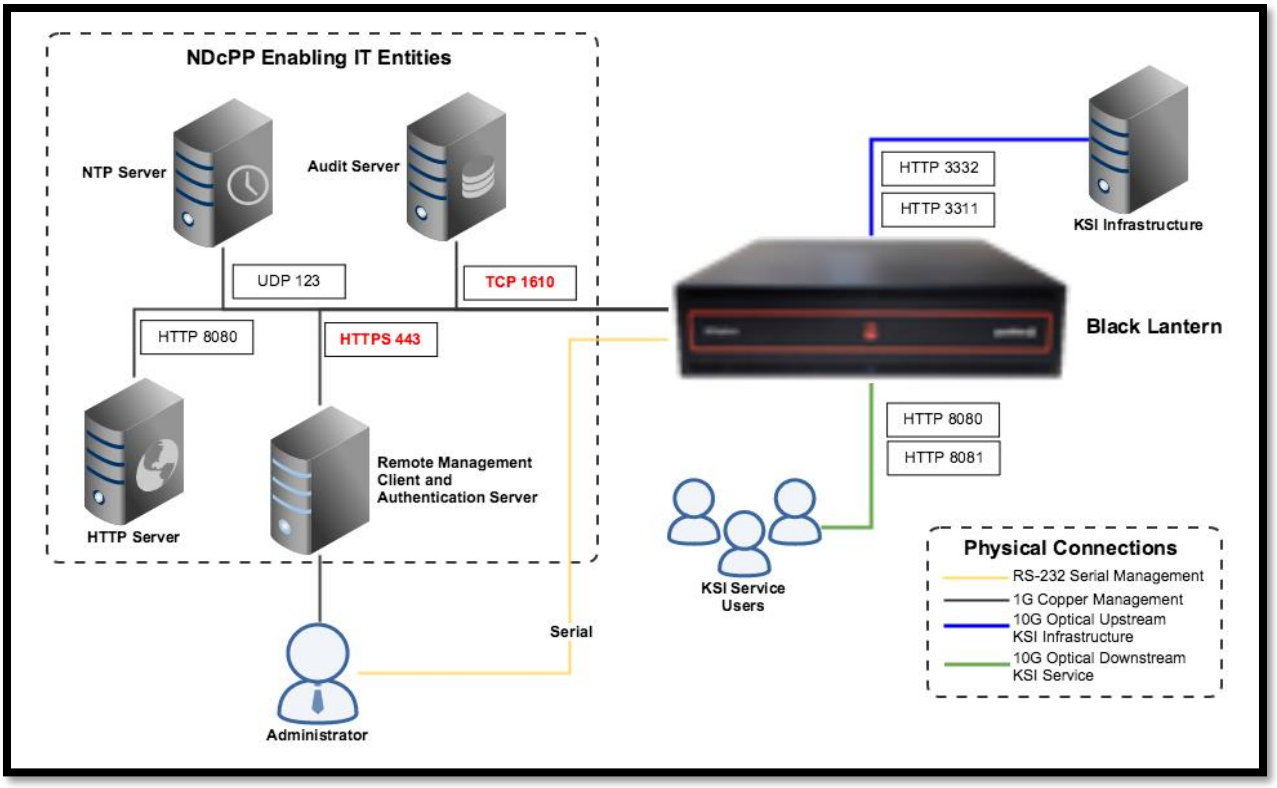

Figure 1: Black Lantern Operational Environment

Figure 1 combines the physical and logical interconnectivity of the TOE to external IT entities and users. All network connections are associated with a port the TOE uses for specific communication; some ports are configurable. Secured ports are designated with color red.

All IT entities that enable the TOE to comply with the NDcPP requirements are connected to the TOE via at the management interface. The following are the required entities:

- Local Management: NDcPP requires the TOE to provide the capabilities to be managed locally. Local management of the TOE is performed via the RS 232 serial interface, which provides access to its Serial Console Interface (SCI). All management commands and configurations of the TOE are accessible via this interface by the appropriate administrator user. For this interface, the TOE uses its local authentication mechanisms to grant access to the administrator.
- Remote Management: NDcPP requires the TOE to be able to be managed remotely. The TOE provides a RESTful Interface that enables it to be managed remotely by an administrator making requests with calls to the TOE's RESTful application program interface (API).
- Remote Authentication Service: In order to allow remote management by administrators, the TOE uses a remote authentication service. This service is detailed in the TOE Guidance Documentation. The TOE uses its RESTful Interface to communicate with the authentication server. All remote management requests come with an Authentication Token (authToken), which is issued by the authentication server. The TOE extracts the authToken, and verifies it with the authentication server. The authToken is composed in

a standard JSON Web Token (JWT). Once the TOE verifies the veracity of the authToken, the TOE serves the remote management request, which came through its RESTful Interface.
- Audit Server: NDcPP requires the TOE to send logs to an external audit server over a secured channel.

The non-required IT entities that help the BL comply with NDcPP requirements are:

- NTP Server: The BL supports synchronization to NTP servers, which helps comply with the requirement of providing reliable time services.
- HTTP Server: The BL supports connection to an HTTP server for the purpose of updating the software in the BL.

The non NDcPP IT entities are the ones associated with providing KSI services. The TOE utilizes four 10G optical interface connections to provide KSI services to the service users. The following are the types of connections required to provide KSI services. The Security Administrator has the ability to determine if a KSI service connection type requires 1, 2, or 3 physical connections.

- 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 BL uses ports 8080 and 8081 as default SDK signing and extender request, respectively.
- Upstream KSI Infrastructure: The BL interface with the higher level aggregator in the KSI infrastructure through this type of interface. Service user does not have access to this interface, only the TOE is allow to interact with the rest of the KSI infrastructure through this interface.

1.3.1.2 Logical Scope of the TOE

This section identifies the security functions that TOE provides, consistent with the functional requirements specified in [NDcPP]. These comprise the following:

- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TOE Security Functions
- TOE Access
- TOE Trusted Path and Channel

1.3.1.2.1 Security Audit

The following events are auditable by the TOE in compliance with [NDcPP]:

- startup and shutdown of the TOE's auditing function
- all administrative actions, specifically including at least the following:

    - Administrative login and logout (name of Administrator user accounts are logged)
    - Security related configuration changes (in addition to the information that a change occurred, what has been changed is also logged)
    - Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself the unique key name or key reference is also logged)
    - Resetting passwords (name of related user account is also logged)
    - Starting and stopping services (when applicable)
- any failure to establish a TLS or HTTPS session, including the reason for the failure
- all use of identification and authentication mechanisms, including the provided user identity and the origin of the attempt (e.g., host name)
- unsuccessful attempts to validate X.509 certificates, including the reason for failure
- modification of the behavior of the handling of audit data
- modification of the behavior of the TSF
- all management activities of TSF data
- any attempt to initiate a TOE update and the result of the update attempt (success or failure)
- changes to time, including the old and new values for the time and the origin of the attempt to change the time (e.g., host name) for both successful and failed attempts
- any attempts at unlocking of an interactive session (in practice, this event would never be generated by Black Lantern because it does not provide a capability to lock local interactive sessions at the console, only to terminate them)
- termination of a remote session by the session locking mechanism (in practice, it is expected this event would never be generated by Black Lantern, since a “remote administrative session” via the RESTful API constitutes a single request that is terminated immediately after the request is submitted to the interface and is never open for any measurable period of inactivity)
- termination of an interactive session by the administrator
- initiation and termination of trusted channels and failure of trusted channel functions, including identification of the initiator and target of failed trusted channel establishment attempts
- initiation and termination of trusted path and failure of trusted path functions, including the claimed user identity.

The TOE captures and records certain information about the audit event in the audit log, including at minimum the date and time of the event, the event type, the subject identity and the outcome of the event. For audit events resulting from the actions of identified users, the identity of the user is recorded in the generated audit record.

The TOE is able to store generated audit records locally and to export audit records (logs) securely to an external syslog server over TLS. In the event the space available for storing audit records locally is exhausted, the TOE will drop new audit data until such time as space is again available. The TOE is able to keep track of the number of dropped audit records and to write this number to the audit trail once it has been cleared and space has been made available for storage of new audit records.

The TOE writes a warning to the audit trail when the space available for storage of audit records reaches the following thresholds: 25% space remaining; 15% space remaining; 10% space remaining; 5, 4, 3, 2, and 1% space remaining.

1.3.1.2.2 Cryptographic Support

The TOE incorporates the Guardtime Crypto Support Library (CSL) Direct v1.0.0 to provide cryptographic algorithms and support cryptographic protocols, including TLS and HTTPS. In order to satisfy the cryptographic requirements specified in [NDcPP], the TOE's implementation of each of the required cryptographic algorithms are certified via the NIST Cryptographic Algorithm Validation Program (CAVP). The following table lists all the CAVP certifications for all the cryptographic algorithms within the TOE:

Table 2: Black Lantern CAVP Certified Cryptographic Algorithms

Functionality	Algorithm	CAVP Certificate #
RSA scheme key generation [FCS_CKM.1]	RSA: FIPS 186-4 KEY(gen)	2456
ECC scheme key generation [FCS_CKM.1]	ECDSA: FIPS 186-4 • PKG: Curves (P-256, P-384, P-521) • PKV: Curves (P-256, P-384, P-521)	1095
CVL ECC Primitives [FCS_CKM.2]	ECC CDH: P-256, P-384, and P-521 curves	1196
CVL KAS ECC for Key Establishment [FCS_CKM.2]	Key pair generation; Ephemeral Unified; Initiator and Responder • EC: Curve: P-256, SHA: SHA-256 • ED: Curve: P-384, SHA: SHA-384 • EE: Curve: P-512, SHA: SHA-512	1488
CVL KDF for Key Establishment [FCS_CKM.2]	NIST 800-135 KDF TLS 1.2: • SHA-256, SHA-384, SHA-512	1489
AES-CBC cryptographic algorithm [FCS_COP.1(1)]	AES CBC (e/d; 128, 256)	4508
AES-GCM cryptographic algorithm [FCS_COP.1(1)]	AES GCM (KS: AES_128 (e/d)), (KS: AES_256 (e/d))	4508
RSA signature services [FCS_COP.1(2)]	ALG [RSASSA-PKCS1-v1_5] SIG(gen) 2048, 3072, and 4096 SHA (256, 384, 512) SIG(ver) 2048 and 3072 SHA (256, 384, 512);	2456
ECDSA signature services [FCS_COP.1(2)]	Key Pair Generation: CURVES P-256, P-384, P-521 SigGen: CURVES P-256, P-384, P-521: (SHA (256, 384, 512)) SigVer: CURVES P-256, P-384, P-521: (SHA (256, 384, 512))	1095
Hashing Algorithm [FCS_COP.1(3)]	SHS: SHA-1, SHA-256, SHA-384, SHA-512	3697
Keyed Hash [FCS_COP.1(4)]	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	2979
Random Bit Generator [FCS_RBG_EXT.1]	CTR_DRBG (AES-128, AES-256)	1472

Note: TOE is capable of generating RSA keys of 4096 and verifying RSA signatures of 4096, however there are no official NIST test for it.

The cryptographic mechanisms support cryptographic protocols used for secure communication—TLS (both as client and server and with mutual authentication) and HTTPS.

1.3.1.2.3 Identification and Authentication

The TOE requires users (i.e., administrators) to be successfully identified and authenticated before they can access any security management functions available in the TOE. The TOE offers both a locally connected console and a network accessible interface over HTTPS (the RESTful API) to support administration of the TOE.

The TOE supports the local (i.e., on device) definition of administrators with usernames and passwords. When a user is authenticated at the local console, 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 supports the use of X.509v3 certificates for TLS authentication and also supports certificate revocation checking using OCSP mechanism. It will not accept a certificate if it is unable to establish a connection in order to determine the certificate's validity

1.3.1.2.4 Security Management

The TOE implements a role-based access control model with the following three defined roles:

- Security Administrator—has authorizations to manage users (add user, update user, add user to group, delete user from group), provision Black Lantern, update TOE software, and upload certificates
- Network Administrator—has authorizations to manage network-related configuration (device network configuration, remote host configuration)
- KSI Administrator—has authorizations to manage all KSI-related configuration (all aggregator and extender configuration)

The TOE supports the following required security management functions:

- Configure TOE access banner
- Configure session inactivity time-out before session termination
- Update the TOE, and verify updates using a digital signature capability prior to installing those updates
- Configure audit behavior
- Configure the cryptographic functionality.

Only the Security and Network Administrator has the necessary authorizations to be able to manage the TOE security functionality and TSF data, as specified by the TSF of [NDcPP]. The TOE provides a local console connection and a RESTful API as mechanisms to access its security management functions. Security management commands are limited to administrators and are available only after they have provided acceptable user identification and authentication data to the TOE.

1.3.1.2.5 Protection of the TOE Security Functions

The TOE protects sensitive data such as stored passwords and cryptographic keys so that they are not accessible even by an administrator. It also provides its own timing mechanism to ensure that reliable time information is available (e.g., for log accountability).

The TOE provides mechanisms to view the current version of the TOE and to install updates of the TOE software. TOE updates are initiated manually by the Security Administrator. The TOE can verify the integrity of the update prior to installation using a decryption mechanism and a digital signature.

The TOE performs tests for connection integrity and cryptographic known-answer tests.

1.3.1.2.6 TOE Access

The TOE will terminate local interactive sessions at the local console interface after a configurable period of inactivity. The default time-out value is 15 minutes and this can be configured by a user with the Security Administrator 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. NIAP has confirmed the requirement for TSF-initiated termination of remote interactive sessions can be interpreted as being vacuously satisfied, since the "remote interactive session" is terminated immediately after the request is submitted to the interface and is never open for any measurable period of inactivity.

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, NIAP has confirmed this can be interpreted as being satisfied because the "Administrator's own interactive session" is terminated immediately after the request is submitted to the interface.

The TOE is able to display an administrator-configurable advisory and consent warning message at the local console prior to an administrator establishing an interactive session with the TOE. NIAP has confirmed the requirement for a TOE access banner is not applicable to the RESTful API, as it is considered to be a "programmatic connection" as described in Application Note 38 of [NDcPP].

1.3.1.2.7 TOE Trusted Path/Channel

The TOE utilizes TLS version 1.2, in compliance with RFC 5246, to support secure path and channel communications. The TOE supports the establishment of a trusted path between a remote management tool and the TOE, and initiated by the remote management tool. The TOE establishes trusted channels between itself and the audit server and authentication server. All TLS connections are mutual authenticated. Note that the communication with the remote management tool and the authentication server uses HTTP over TLS.

2 Conformance Claims (ASE_CCL)

This ST and the TOE, it describes, are conformant to the following CC specifications:

- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Components, Version 3.1, Revision 4, September 2012
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Components, Version 3.1, Revision 4, September 2012
  - Part 3 Conformant

This ST and the TOE, it describes, are conformant to the following Protection Profile

- collaborative Protection Profile for Network Devices Version 1.0, 27 February 2015, [NDcPP] and including the following optional SFRs: FAU_STG.1, FAU_STG_EXT.2, FAU_STG_EXT.3, FCS_HTTPS_EXT.1, FCS_TLSC_EXT.2, and FCS_TLSS_EXT.2, FMT_MOF.1(2)/AdminAct, FMT_MOF.1(2)/Audit, FMT_MTD.1/AdminAct. The following NIAP Technical Decisions apply to this PP and have been accounted for in the ST development and the conduct of the evaluation:
  - TD0235: NIT Technical Decision adding DH group 14 to the selection in FCS_CKM.2
  - TD0228: NIT Technical Decision for CA certificates - basicConstraints validation
  - TD0226: NIT Technical Decision for TLS Encryption Algorithms
  - TD0201: NIT Technical Decision for Use of intermediate CA certificates and certificate hierarchy depth
  - TD0199: NIT Technical Decision for Elliptic Curves for Signatures
  - TD0188: NIT Technical Decision for Optional use of X.509 certificates for digital signatures
  - TD0187: NIT Technical Decision for Clarifying FIA_X509_EXT.1 test 1
  - TD0185: NIT Technical Decision for Channel for Secure Update
  - TD0168: NIT Technical Decision for Mandatory requirement for CSR generation
  - TD0156: NIT Technical Decision for SSL/TLS Version Testing in the NDcPP v1.0 and FW cPP v1.0
  - TD0155: NIT Technical Decision for TLSS tests using ECDHE in the NDcPP v1.0
  - TD0154: NIT Technical Decision for Versions of TOE Software in the NDcPP v1.0 and FW cPP v1.0
  - TD0153: NIT Technical Decision for Auditing of NTP Time Changes in the NDcPP v1.0 and FW cPP v1.0
  - TD0152: NIT Technical Decision for Reference identifiers for TLS in the NDcPP v1.0 and FW cPP v1.0
  - TD0151: NIT Technical Decision for FCS_TLSS_EXT Testing - Issue 1 in NDcPP v1.0
  - TD0130: NIT Technical Decision for Requirements for Destruction of Cryptographic Keys
  - TD0126: NIT Technical Decision for TLS Mutual Authentication
  - TD0125: NIT Technical Decision for Checking validity of peer certificates for HTTPS servers
  - TD0117: NIT Technical Decision for FIA_X509_EXT.1.1 Requirement in NDcPP
  - TD0116: NIT Technical Decision for a Typo in reference to RSASSA-PKCS1v1_5 in NDcPP and FWcPP

- TD0113: NIT Technical Decision for testing and trusted updates in the NDcPP v1.0 and FW cPP v1.0
- TD0112: NIT Technical Decision for TLS testing in the NDcPP v1.0 and FW cPP v1.0
- TD0111: NIT Technical Decision for third party libraries and FCS_CKM.1 in NDcPP and FWcPP
- TD0095: NIT Technical Interpretations regarding audit, random bit generation, and entropy in NDcPP
- TD0090: NIT Technical Decision for FMT_SMF.1.1 Requirement in NDcPP

3 Security Problem definition (ASE_SPD)

3.1 Introduction

This section defines the security problem in terms of Threats, Organizational Security Policies, and Assumptions that the operational environment will present to the TOE. The security problem is in accordance with NDcPP, and SFRs detailed in this ST were selected to address the threats, organizational security policies, and assumptions of the operational environment.

3.2 Threats

3.2.1 Communications with the Network Device

3.2.1.1 T.UNAUTHORIZED_ADMINISTRATOR_ACCESS

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

3.2.1.2 T.WEAK_CRYPTOGRAPHY

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

3.2.1.3 T.UNTRUSTED_COMMUNICATION_CHANNELS

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

3.2.1.4 T.WEAK_AUTHENTICATION_ENDPOINTS

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

3.2.2 Valid Updates

3.2.2.1 T.UPDATE_COMPROMISE

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

3.2.3 Audited Activity

3.2.3.1 T.UNDETECTED_ACTIVITY

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

3.2.4 Administrator and Device Credentials and Data

3.2.4.1 T.SECURITY_FUNCTIONALITY_COMPROMISE

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

3.2.4.2 T.PASSWORD_CRACKING

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

3.2.5 Device Failure

3.2.5.1 T.SECURITY_FUNCTIONALITY_FAILURE

A component of the network device may fail during start-up or during operations causing a compromise or failure in the security functionality of the network device, leaving the device susceptible to attackers.

3.3 Organizational Security Policies (OSPs)

3.3.1 P.ACCESS_BANNER

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

3.4 Assumptions

3.4.1 A.PHYSICAL_PROTECTION

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

3.4.2 A.LIMITED_FUNCTIONALITY

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

3.4.3 A.NO_THRU_TRAFFIC_PROTECTION

A standard/generic network device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the network device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the network device, destined for another network entity, is not covered by the ND cPP. It is assumed that this protection will be covered by cPPs for particular types of network devices (e.g, firewall).

3.4.4 A.TRUSTED_ADMINISTRATOR

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

3.4.5 A.REGULAR_UPDATES

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

3.4.6 A.ADMIN_CREDENTIALS_SECURE

The administrator's credentials (private key) used to access the network device are protected by the platform on which they reside.

4 Security Objectives (ASE_OBJ)

This ST includes by reference the Security Objectives for the TOE specified in the *collaborative Protection Profile for Network Devices*, Version 1.0, 27 February 2015. The security objectives for the operational environment are reproduced below, since these objectives characterize technical and procedural measures each consumer must implement in their operational environment.

4.1 Security objectives for the operational environment

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

5 Security Requirements (ASE_REQ)

This section identifies the Security Functional Requirements (SFR) for the TOE. The SFRs included in this section were identified as applicable, to the TOE, from the Collaborative Protection Profile for Network Devices, Version 1.0, data February 27, 2015 [NDcPPv1.0]; all incomplete selection and assignments have been performed herein. The Security Assurance Requirements (SARs) were taken directly from the NDcPPv1.0.

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the [NDcPPv1.0].

- FCS_HTTPS_EXT.1: HTTPS Protocol
- FCS_TLSC_EXT.2: TLS Client Protocol with Authentication
- FCS_TLSS_EXT.2: TLS Server Protocol with Mutual Authentication
- FCS_RBG_EXT.1: Random Bit Generation
- FIA_X509_EXT.1: X.509 Certificate Validation
- FIA_X509_EXT.2: X.509 Certificate Authentication
- FIA_X509_EXT.3: X.509 Certificate Request
- FPT_SKP_EXT.1: Protection of TSF Data (for reading of all symmetric keys)
- FIA_UAU_EXT.2: Password-based Authentication Mechanism
- FIA_UIA_EXT.1: User Identification and Authentication
- FIA_PMG_EXT.1: Password Management
- FPT_APW_EXT.1: Protection of Administrator Password
- FTA_SSL_EXT.1: TSF-initiated Session Locking
- FPT_TST_EXT.1: TSF Testing (Extended)
- FPT_TUD_EXT.1: Trusted Update
- FAU_STG_EXT.1: Protected Audit Event Storage
- FAU_STG_EXT.2: Counting Lost Audit Data
- FAU_STG_EXT.3: Display Warning for Local Storage Space

5.2 Security Functional Requirement

This section specifies the SFRs for the TOE.

Table 3: TOE's Security Functional Requirements List

Requirement Class	Requirement Component
FAU: Security Audit	FAU_GEN.1: Audit Data Generation
	FAU_GEN.2: User Identity Association
	FAU_STG_EXT.1: Protected Audit Event Storage
	FAU_STG.1: Protected Audit Trail Storage
	FAU_STG_EXT.2: Counting Lost Audit Data

Requirement Class	Requirement Component
	FAU_STG_EXT.3: Display Warning for Local Storage Space
FCS: Cryptographic Support	FCS_CKM.1: Cryptographic Key Generation (Refined)
	FCS_CKM.2: Cryptographic Key Establishment (Refined)
	FCS_CKM.4: Cryptographic Key Destruction
	FCS_COP.1(1): Cryptographic Operation (AES Data Encryption/ Decryption)
	FCS_COP.1(2): Cryptographic Operation (Signature Generation and Verification)
	FCS_COP.1(3): Cryptographic Operation (Hash Algorithm)
	FCS_COP.1(4): Cryptographic Operation (Keyed Hash Algorithm)
	FCS_RBG_EXT.1: Random Bit Generation
	FCS_HTTPS_EXT.1: HTTPS Protocol
	FCS_TLSC_EXT.2: TLS Client Protocol with Authentication
	FCS_TLSS_EXT.2: TLS Server Protocol with Mutual Authentication
FIA: Identification and Authentication	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: 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(1)/TrustedUpdate: Management of Security Functions Behaviour
	FMT_MTD.1: Management of TSF Data
	FMT_SMF.1: Specification of Management Functions
	FMT_SMR.2: Restrictions on Security Roles

Requirement Class	Requirement Component
	FMT_MOF.1(2)/Audit: Management of Security Functions Behaviour
	FMT_MOF.1(2)/AdminAct: Management of Security Functions Behaviour
	FMT_MTD.1/AdminAct: Management of TSF data
FPT: Protection of the TSF	FPT_SKP_EXT.1: Protection of TSF Data (for reading of all symmetric keys)
	FPT_APW_EXT.1: Protection of Administrator Passwords
	FPT_TST_EXT.1: TSF testing
	FPT_TUD_EXT.1: Trusted update
	FPT_STM.1: Reliable Time Stamps
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: Trusted Path

5.2.1 Security Audit (FAU)

5.2.1.1 FAU_GEN.1 - Audit Data Generation

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).*
- *Security related 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).*
- *Starting and stopping services (if applicable)*

- *[**no other actions**];*

d) *Specifically defined auditable events listed in Table 4.*

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 PP/ST, *information specified in column three of Table 4.*

Table 4: Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None	None
FAU_GEN.2	None	None
FAU_STG_EXT.1	None	None
FAU_STG.1	None	None
FAU_STG_EXT.2	None	None
FAU_STG_EXT.3	Warning about 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(1)	None	None
FCS_COP.1(2)	None	None
FCS_COP.1(3)	None	None
FCS_COP.1(4)	None	None
FCS_RBG_EXT.1	None	None
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure.
FCS_TLSC_EXT.2	Failure to establish a TLS Session.	Reason for failure.
FCS_TLSS_EXT.2	Failure to establish a TLS Session.	Reason for failure.

Requirement	Auditable Events	Additional Audit Record Contents
FIA_PMG_EXT.1	None	None
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None	None
FIA_X509_EXT.1	Unsuccessful attempt to validate a certificate.	Reason for failure.
FIA_X509_EXT.2	None	None
FIA_X509_EXT.3	None	None
FMT_MOF.1(1)/TrustedUpdate	Any attempt to initiate a manual update.	None
FMT_MTD.1	All management activities of TSF data.	None
FMT_SMF.1	None	None
FMT_SMR.2	None	None
FMT_MOF.1(2)/Audit	Modification of the behaviour of the handling of audit data.	None
FMT_MOF.1(2)/AdminAct	Starting and stopping of services.	None
FMT_MTD.1/AdminAct	Modification, deletion, generation/import of cryptographic keys.	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).	No additional information.

Requirement	Auditable Events	Additional Audit Record Contents
FPT_STM.1	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	Any attempts at unlocking of an interactive session.	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	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	Identification of the claimed user identity.

5.2.1.2 FAU_GEN.2 - User identity association

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

5.2.1.3 FAU_STG_EXT.1 - Protected Audit Event Storage

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

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

FAU_STG_EXT.1.3 The TSF shall [***drop new audit data***] when the local storage space for audit data is full.

5.2.1.4 FAU_STG.1 - Protected Audit Trail Storage

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.

5.2.1.5 FAU_STG_EXT.2 - Counting Lost Audit Data

FAU_STG_EXT.2.1 The TSF shall provide information about the number of [***dropped***] 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.

5.2.1.6 FAU_STG_EXT.3 - Display warning for local storage space

FAU_STG_EXT.3.1 The TSF shall generate a warning to inform the user before the local space to store audit data is used up and/or the TOE will lose audit data due to insufficient local space.

5.2.2 Cryptographic Support (FCS)

5.2.2.1 FCS_CKM.1 - Cryptographic Key Generation

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;***

]

5.2.2.2 FCS_CKM.2 - Cryptographic Key Establishment

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, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***

]

5.2.2.3 FCS_CKM.4 - Cryptographic Key Destruction

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 [selection:*
    - *logically addresses the storage location of the key and performs a [selection: single, [assignment: number of passes]-pass] overwrite consisting of [selection: a pseudo-random pattern using the TSF's RBG, zeroes, ones, a new value of the key, [assignment: a value that does not contain any CSP]];*

- *instructs a part of the TSF to destroy the abstraction that represents the key]]*

that meets the following: No Standard.

Application Note: No plaintext keys are stored in non-volatile storage.

5.2.2.4 FCS_COP.1(1) - Cryptographic Operation (AES Data Encryption/Decryption)

FCS_COP.1.1(1) The TSF shall perform encryption/decryption in accordance with a specified cryptographic algorithm AES used in [***CBC***, ***GCM***] mode and cryptographic key sizes [***128 bits, 256 bits***] that meet the following: AES as specified in ISO 18033-3, [***CBC as specified in ISO 10116***, ***GCM as specified in ISO 19772***].

5.2.2.5 FCS_COP.1(2) - Cryptographic Operation (Signature Generation and Verification)

FCS_COP.1.1(2) 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) [2048 bits]***
- ***Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256, 384, and 521 bits]***

]

that meet the following: [:

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

].

Application Note: The TOE supports RSA digital signature with cryptographic key size of 4096 as specified in FIPS PUB 186-2.

5.2.2.6 FCS_COP.1(3) - Cryptographic Operation (Hash Algorithm)

FCS_COP.1.1(3) The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm [***SHA-1, SHA-256, SHA-384, SHA-512***] ~~*and cryptographic key sizes [assignment: cryptographic key sizes]*~~ that meet the following: ISO/IEC 10118-3:2004.

5.2.2.7 FCS_COP.1(4) - Cryptographic Operation (Keyed Hash Algorithm)

FCS_COP.1.1(4) The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm [***HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512***] and cryptographic key sizes [***256, 512, 2048 bits for HMAC-SHA-1 and***

HMAC-SHA-256; 256, 1024, and 2048 bits for HMAC-SHA-384 and HMAC-SHA-512] **and message digest sizes** [***160, 256, 384, 512***] **bits** that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

5.2.2.8 FCS_RBG_EXT.1 - Random Bit Generation

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

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [***[1] hardware-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.

5.2.2.9 FCS_HTTPS_EXT.1 - HTTPS Protocol

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 The TSF shall establish the connection only if [***the peer presents a valid certificate during handshake***].

5.2.2.10 FCS_TLSC_EXT.2 - TLS Client Protocol with Mutual Authentication

FCS_TLSC_EXT.2.1 The TSF shall implement [***TLS 1.2 (RFC 5246)***] supporting 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_TLSC_EXT.2.2 The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.2.3 The TSF shall only establish a trusted channel if the peer certificate is valid.

FCS_TLSC_EXT.2.4 The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [***secp256r1, secp384r1, secp521r1***] and no other curves.

FCS_TLSC_EXT.2.5 The TSF shall support mutual authentication using X.509v3 certificates.

5.2.2.11 FCS_TLSS_EXT.2 - TLS Server Protocol with Mutual Authentication

FCS_TLSS_EXT.2.1 The TSF shall implement [***TLS 1.2 (RFC 5246)***] supporting 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.2.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, and [***TLS 1.1***].

FCS_TLSS_EXT.2.3 The TSF shall [***generate EC Diffie-Hellman parameters over NIST curves [secp256r1, secp384r1, secp521r1] and no other curves***].

FCS_TLSS_EXT.2.4 The TSF shall support mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.5 The TSF shall not establish a trusted channel if the peer certificate is invalid.

FCS_TLSS_EXT.2.6 The TSF shall not establish a trusted channel if the distinguished name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match the expected identifier for the peer.

5.2.3 Identification and Authentication (FIA)

5.2.3.1 FIA_PMG_EXT.1 - Password Management

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

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [***“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(”, “)”, [“_”, “?”, “<”, “>”, “.”, “~”, “|”]***];

b) Minimum password length shall be settable by the Security Administrator, and shall support passwords of 15 characters or greater.

5.2.3.2 FIA_UIA_EXT.1 - User Identification and Authentication

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

- Display the warning banner in accordance with FTA_TAB.1;
- [***ICMP echo***]

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

5.2.3.3 FIA_UAU_EXT.2 - Password-based Authentication Mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local password-based authentication mechanism, [***none***] to perform administrative user authentication.

5.2.3.4 FIA_UAU.7 - Protected Authentication Feedback

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

5.2.3.5 FIA_X509_EXT.1 - X.509 Certificate Validation

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

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The TSF shall validate the revocation status of the certificate using [***the Online Certificate Status Protocol (OCSP) as specified in RFC 2560***].
- 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 The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

5.2.3.6 FIA_X509_EXT.2 - X.509 Certificate Authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [***TLS, 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***].

5.2.3.7 FIA_X509_EXT.3 - X.509 Certificate Requests

FIA_X509_EXT.3.1 The TSF shall generate a Certificate Request Message 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.

5.2.4 Security Management (FMT)

5.2.4.1 FMT_MOF.1(1)/TrustedUpdate - Management of Security Functions Behavior

FMT_MOF.1.1(1)/TrustedUpdate The TSF shall restrict the ability to enable the functions *to perform manual update to Security Administrators*.

5.2.4.2 FMT_MTD.1 - Management of TSF Data

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

5.2.4.3 FMT_SMF.1 - Specification of Management Functions

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

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using [***digital signature***] capability prior to installing those updates;
- [
    - ***Ability to configure audit behavior;***
    - ***Ability to configure the cryptographic functionality;***]

5.2.4.4 FMT_SMR.2 - Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

- *Security Administrator*.

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

FMT_SMR.2.3 The TSF shall ensure that the conditions

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

are satisfied.

5.2.4.5 FMT_MOF.1(2)/Audit - Management of Security Functions Behavior

FMT_MOF.1.1(2)/Audit The TSF shall restrict the ability to determine the behaviour of, modify the behaviour of the functions *handling of audit data* to *Security Administrators*.

5.2.4.6 FMT_MOF.1(2)/AdminAct - Management of Security Functions Behavior

FMT_MOF.1.1(2)/AdminAct The TSF shall restrict the ability to enable, disable the functions *services to Security Administrators*.

5.2.4.7 FMT_MTD.1/AdminAct - Management of TSF Data

FMT_MTD.1.1/AdminAct The TSF shall restrict the ability to *modify, delete, generate/import* the *cryptographic keys* to *Security Administrators*.

5.2.5 Protection of TSF (FPT)

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

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

5.2.5.2 FPT_APW_EXT.1 - Protection of Administrator Passwords

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

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

5.2.5.3 FPT_TST_EXT.1 - TSF Testing

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

Test	Description
BOOT_CODE_RAM_MARCH	5N RAM March that will cover the RAM area used when boot is copied from flash to RAM
BOOT_CODE_ECC	Test ECC error generation and checking by inserting known errors and ensuring they are correctly detected and in the single bit error case fixed
BIT_INT_REG_TEST	Test the 32 general purpose registers, This test will walk ones through each of the general purpose registers from GPR 2 through GPR 31.
BIT_SPR_REG_TEST	Test the special purpose registers, this test will set bits to one and zero and verify that each bit can be set and cleared correctly.
BIT_INT_COMPARE_TEST	Test the integer condition register and check the compare instructions.
BIT_BRANCH_TEST	The branch instructions and ensure that the correct branch is taken, and the special purpose registers associated with the branch instruction are updated.

Test	Description
BIT_INT_MATH_TEST	Execute every integer arithmetic instructions including compares and shifts, using test patterns that use all the data paths and functional operation of the integer arithmetic unit.
BIT_INT_LOAD_STORE_TEST	Test each of the available general purpose register memory load and store instructions.
BIT_FPU_REG_TEST	Test the 32 floating point registers, This test will walk ones through each of the floating point registers.
BIT_FPU_LOAD_STORE_TEST	Test each of the available floating point register memory load and store instructions.
BIT_FPU_MATH_TEST	Execute every floating point arithmetic instructions including compares and shifts, using test patterns that use all the data paths and functional operation of the floating point arithmetic unit.
BIT_TIMEBASE_TEST	Check the timebase and decrementer registers and ensure the correct rollover and and incrementing/decrementing also check that the timebase and decrementer are keeping time within bounds as compared to the processor tics.
BIT_DCACHE_TEST	Test the L2 and CPC cache to ensure the core is capable of using the flushing, invalidating and using the data cache correctly.
BIT_RAM_MARCH_TEST	Perform a 5N RAM march on the available DDR.
BIT_RNG_TEST	Consist of four tests from FIPS 140-1: Monobit, Poker, Runs, and Long Run.
BIT_SHA_TEST	Use known values to compare the results for SHA-1, SHA-256, SHA-384, SHA-512.
BIT_AES_CBC_TEST	AES-CBC mode test that uses verifies the data that is encrypted with a known key/iv can be decrypted correctly with the same known key/iv.
BIT_HMAC_TEST	HMAC test that uses known answers to compare results for HMAC_SHA-1, HMAC_SHA_256, HMAC_SHA_384, and HMAC_SHA_512.
BIT_AES_GCM_TEST	AES-GCM mode test that uses verifies the data that is encrypted with a known key, iv, aad, and tag can be decrypted correctly with the same known key, iv, aad, and the updated tag.

Test	Description
BIT_SIGN_VER_TEST	Test the RSA and ECDSA algorithms function correctly by signing and verifying an image with known curves and keys.

].

5.2.5.4 FPT_TUD_EXT.1 - Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide *Security Administrators* the ability to query the currently executing version of the TOE firmware/software and [***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 mechanism***] prior to installing those updates.

5.2.5.5 FPT_STM.1 - Reliable Time Stamps

FPT_STM.1.1 The TSF shall be able to provide reliable time stamps.

5.2.6 TOE Access (FTA)

5.2.6.1 FTA_SSL_EXT.1 - TSF-initiated Session Locking

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [

- ***terminate the session***]

after a Security Administrator-specified time period of inactivity.

5.2.6.2 FTA_SSL.3 -TSF-initiated Termination

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

5.2.6.3 FTA_SSL.4 -User-initiated Termination

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

5.2.6.4 FTA_TAB.1 - Default TOE Access Banners

FTA_TAB.1.1 Refinement: 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.

5.2.7 Trusted Path / Channels (FTP)

5.2.7.1 FTP_ITC.1 - Inter-TSF Trusted Channel

FTP_ITC.1.1 The TSF shall be **capable of using** [***TLS, HTTPS***] **to** provide a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server,** [***authentication server***] 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 [***requesting authentication services to authentication server (over HTTPS), transaction of audit records to audit server (over TLS)***].

5.2.7.2 FTP_TRP.1 - Trusted Path

FTP_TRP.1.1 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 The TSF shall permit **remote administrators** to initiate communication via the trusted path.

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

5.3 Security Assurance Requirements (SAR)

This section outlines all TOE security assurance requirements (SAR), which are taken directly from the [NDcPP]. The following table identifies the security assurance requirements that are specified by [NDcPP].

Table 5: NDcPP Security Assurance Requirements

Assurance Class	Assurance Components
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

5.3.1 SAR Rationale

5.3.1.1 ADV_FSP.1

Guardtime will provide a functional specification that contains traceability to the SFRs. The functional specification will describe the purpose and method of use for each SFR-enforcing and SFR-supporting TSFI. The functional specification will identify all parameters associated with each SFR-enforcing and SFR-supporting TSFI. The functional specification will provide rationale for the implicit categorization of interfaces as SFR-non-interfering. The tracing demonstrates that the SFRs trace to the TSFIs in the functional specification.

5.3.1.2 AGD_OPE.1

Guardtime will provide a clear and reasonable operational user guidance for the TOE, which will:

- describe, for each user role, the user-accessible functions and privileges that should be controlled in a secure processing environment, including appropriate warnings.
- describe, for each user role, how to use the available interfaces provided by the TOE in a secure manner.
- describe, for each user role, the available functions and interfaces, in particular all security parameters under the control of the user, indicating secure values as appropriate.
- for each user role, clearly present each type of security-relevant event relative to the user-accessible functions that need to be performed, including changing the security characteristics of entities under the control of the TSF.
- identify all possible modes of operation of the TOE (including operation following failure or operational error), their consequences and implications for maintaining secure operation.
- for each user role, describe the security measures to be followed in order to fulfill the security objectives for the operational environment as described in the ST.

5.3.1.3 AGD_PRE.1

Guardtime will provide the TOE along with its guidance documentation to perform acceptance, installation, and initial configuration of the TOE.

5.3.1.4 ALC_CMC.1

Guardtime will provide a properly labelled TOE and documentation regarding how to spot and read the label on the TOE.

5.3.1.5 ALC_CMS.1

Guardtime will provide a configuration list for the TOE, which will include: the TOE itself; and the evaluation evidence required by the SARs. The configuration will uniquely identify the configuration items.

5.3.1.6 ATE_IND.1

Guardtime will provide a suitable TOE for testing.

5.3.1.7 AVA_VAN.1

Guardtime will provide a suitable TOE for testing.

6 TOE Summary Specification (ASE_TSS)

6.1 Security Audit (FAU)

The TOE generates audit records as specified in FAU_GEN.1 SFR and Table 4. The event types include:

- System configuration and management
- Identification and authentication
- Cryptographic operations and
- Secure communication

The TOE follows the "Syslog Protocol", as specified in RFC 5424, to format each of its audit records. The following is an example of an audit record:

```
<109>1 2017-03-30T22:04:03.0Z bldev-irv1-04 DeviceMgr 0 - - Ev: Generate key item (Success), ID: admin, Src: Serial, Rsn: [no error], Item: testrsa.prv.enc
```

The mapping to the Syslog Protocol is the following:

Priority	`<109>`
Version	`1`
Timestamp	`2017-03-30T22:04:03.0Z`
Hostname	`bldev-irv1-04`
App Name	`DeviceMgr`
Process ID	`0`
Message ID	`-`
Structure Data	`-`
MSG	`Ev: Generate key item (Success), ID: admin, Src: Serial, Rsn: [no error], Item: testrsa.prv.enc`

The required audit record information, as specified in FAU_GEN.1.2 and within the "Additional Audit Record Content" column of the Table 4, is detailed within the Message ID field. The TOE is capable of sending audit records to a specified external audit server, store it locally, or both. The TOE protects its transmission of audit records by using TLS with mutual authentication to establish a trusted communication channel between itself and the external audit server.

Management of local audit storage is restricted to Security Administrators. Management functions of local audit are as follows:

- Reading
- Deleting
- Starting / Stopping
- Storage size setting

Reading functionality of audit logs is the only management function available to the other type of administrators of the TOE. Every management action is recorded by the TOE when audit functionality is enabled. The TOE checks the permissions of the administrator before executing the command. Non-administrators are not allowed to perform any audit management functions.

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 message are stored (local storage, external storage, or both) and managed in accordance to the TOE's audit logs configuration, as described above. When the local storage is full, the TOE drops all new records and keeps a counter of the audit records dropped. A Security Administrator user is capable of viewing the dropped audit records counter and clearing local storage. There are two methods to clear the local storage, by removing the entire local storage data or by removing a subset of the local log data.

Within the TOE, many of the required NDcPP security functionalities behaviors are managed via configuration changes to the TOE. These changes in configurations are always performed by the Administrator user but the actual execution of the functionality is performed by the TOE. This approach is reflected in the when the TOE generates logs. For example, the following audit records are generated when Administrator user enables remote audit reporting:

```
<109>1 2017-03-24T19:15:21.0Z bldev-irv1-04 DeviceMgr 0 - - Ev: Set config
item (Success), ID: admin, Src: Serial, Rsn: [no error], Item:
device.deviceconfig.managementserver.enableremotelogging = 1

<109>1 2017-03-24T19:15:22.0Z bldev-irv1-04 LogDistService 0 - - Ev: Starting
remote log service (Success), ID: [SYSTEM]
```

The first audit record associates the Administrator user (ID: admin) as the initiator of change in configuration (DeviceMgr). The second audit record associates the TOE (ID: [SYSTEM]) as the initiator of the remote audit service (LogDistService). The two audit records combine provide all the information necessary to determine that it was the Administrator user who enabled the remote audit service.

Depending on the audit record reporting, on power cycle, the TOE will follow the set configurations and establish the appropriate TLS channel to transmit audit records to the external audit server, if the TOE already has such configurations.

6.2 Cryptographic Support (FCS)

6.2.1 Cryptographic Key Management

The TOE utilizes the Guardtime Crypto Support Library (CSL) Direct v1.0.0, to comply with the cryptographic key management SFRs.

The TOE generates asymmetric cryptographic keys in accordance with the following key generation algorithms:

- RSA schemes using cryptographic key sizes[1] of 2048 bits that meets FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3
- ECC schemes using "NIST curves" P-256, P-384, P-521 that meets FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4

Both schemes can be used to generate a Certificate Signing Request (CSR).

The TOE supports the following cryptographic key establishment method:

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

The relevant CAVP certificate numbers are listed in Table 2.

The TOE does not store any keys in plaintext; all keys are encrypted at rest. For keys decrypted into volatile memory, once the keys are no longer needed, the TOE deallocates the memory back to the kernel. The memory gets zeroized when power is removed from the TOE.

All keys are encrypted at rest with AES 256. The root or top-level key-encrypting key is also an AES 256 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. Guardtime uses this key value to protect long-term keys stored on the TOE at rest.

The following Table 6 summarizes the TOE's keys and critical security parameters (CSPs).

Table 6: TOE Relevant Keys and CSPs

Key & CSP Type	Usage	Generation	Storage & Destruction	Encryption
Certificate Signing Requests	TOE Certificate signing request to be signed by CA	By TOE on Security Admin request	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Certificate Signing Requests
TOE TLS Certificate	Used by TOE for TLS	Outside the TOE	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Encrypted with AES 256 key
TOE TLS Private Key; RSA or ECDSA (configured by admin)	Used by TOE for TLS	TOE RNG/Outside the TOE	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Encrypted with AES 256 key

[1] RSA key size of 4096 is supported but it's not certified since there is no official NIST test to date.

Key & CSP Type	Usage	Generation	Storage & Destruction	Encryption
CA Certificates	Used by TOE for verification of certificate chains as in TLS	Outside the TOE	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Encrypted with AES 256 key
TLS ECDSA random ephemeral secret	TLS	GHS SSL module	Stored in RAM. Destroyed by deallocating memory to kernel. Memory gets zeroized when power is removed.	
TLS ECC ephemeral private key for key exchange (ECDHE)				
TLS shared ECDHE secret				
TLS premaster secret				
TLS master Secret				
TLS record layer AES and MAC keys				
TOE Key	Encrypt logs, configurations database, user database and other TOE generated files on TOE	TOE RNG	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Encrypted with AES 256 key
Software update key	Encrypts software updates to TOE	Outside of TOE	SSD storage. Destroyed by zeroizing the memory location, and perform a read after write verify.	Encrypted with AES 256 key

FCS_CKM.1 - Cryptographic Key Generation

FCS_CKM.2 - Cryptographic Key Establishment

FCS_CKM.4 - Cryptographic Key Destruction

6.2.2 Cryptographic Operations

The TOE utilizes the Guardtime Crypto Support Library (CSL) Direct v1.0.0, to comply with the cryptographic operations SFRs.

The TOE performs encryption and decryption using AES as specified in ISO 18033-3 in modes:

- CBC with 128 and 256 bits key sizes as specified in ISO 10116; and
- GCM with 128 and 256 bits key sizes as specified in ISO 19772.

The TOE performs the following cryptographic signature services (generation and validation) per FIPS PUB 186-4, "Digital Signature Standard (DSS)":

- RSA Digital Signature Algorithm (Section 5.5) with key sizes[2] of 2048 bits.
- Elliptic Curve Digital Signature Algorithm (Section 6) with key sizes 256, 384, and 512 bits, implementing P-256, P384, and P-521 NIST curves.

The TOE performs cryptographic hashing services using SHA-1, SHA-256, SHA-384, and SHA-512 in accordance with ISO/IEC 10118-3:2004. Hashing is used as part of RSA and ECDSA key generation and verification.

The TOE performs keyed-hash message authentication using HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512, with the following cryptographic parameters:

Table 7: TOE HMAC Parameters

HMAC	Key Size Range	Key Length (Bits)	Block Size (Bits)	Hash Function	Output MAC Length (Bits)
HMAC-SHA-1	KS<BS	256	512	SHA-1	160
	KS=BS	512	512	SHA-1	160
	KS>BS	2048	512	SHA-1	160
HMAC-SHA-256	KS<BS	256	512	SHA-256	256
	KS=BS	512	512	SHA-256	256
	KS>BS	2048	512	SHA-256	256
HMAC-SHA-384	KS<BS	256	1024	SHA-384	384
	KS=BS	1024	1024	SHA-384	384
	KS>BS	2048	1024	SHA-384	384
HMAC-SHA-512	KS<BS	256	1024	SHA-512	512
	KS=BS	1024	1024	SHA-512	512
	KS>BS	2048	1024	SHA-512	512

The relevant CAVP certificate numbers are listed in Table 2.

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

[2] RSA signature using key sizes of 4096 is certified for FIPS PUB 186-2.

FCS_COP.1(2) - Cryptographic Operation (Signature Generation and Verification)

FCS_COP.1(3) - Cryptographic Operation (Hash Algorithm)

FCS_COP.1(4) - Cryptographic Operation (Keyed Hash Algorithm)

6.2.3 Random Bit Generation

The TOE utilizes the Guardtime Crypto Support Library (CSL) Direct v1.0.0, to comply with the random bit generation SFRs. 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 entropy source, which accumulates entropy from one hardware-based noise source, which has a minimum 256-bits of entropy. The random bit generator supports for cryptographic operations, key management, and TLS/HTTPS session establishment. The relevant CAVP certificate numbers are listed in Table 2.

FCS_RBG_EXT.1 - Random Bit Generation

6.2.4 HTTPS Implementation

The TOE uses HTTPS when connecting to Authorized IT entities using its RESTful Interface. The TOE's HTTPS protocol complies with RFC 2818 and is implemented using TLS 1.2 (RFC 5246). The TOE performs mutual authentication and it expects the peer to present a valid certificate before establishing the connection.

FCS_HTTPS_EXT.1 - HTTPS Protocol

6.2.5 TLS Implementation

6.2.5.1 TLS Client Protocol with Mutual Authentication

The TOE supports the secure communication, when acting as a client, by implementing only TLS 1.2 protocol. As a client, the TOE supports the following ciphersuites:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

The TOE only supports host name as reference identifier, and is capable of handling wildcards. The TOE does not support IP addresses. The reference identifier is set manually by the Security Administrator by setting a Network Configuration parameter, HostName. The TOE supports certificate pinning by allowing the Security Administrator to import Intermediate and Root Certificate Authority (CA) certificate, and associate them with a predefined host.

The TOE supports the secp256r1, secp384r1, and secp521r1 elliptic curves extensions by default when a TOE certificate, containing either of those curves, is imported into the TOE. The TOE supports mutual authentication using X.509v3 certificates.

FCS_TLSC_EXT.2 - TLS Client Protocol with Mutual Authentication

6.2.5.2 TLS Server Protocol with Mutual Authentication

The TOE supports the secure communication, when acting as a server, by implementing only TLS 1.2 protocol, and denying older TLS and SSL versions. As a server, the TOE supports the following ciphersuites:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384

The TOE supports elliptic curve key establishment, as specified in FCS_TLSS_EXT.2.3. The TOE uses the last certificate imported into it, which specifies the type of key to be used. The TOE supports mutual authentication using X.509v3 certificates.

The expected identifier of the peer is configured into the TOE as a remote host configuration parameter. During a TLS connection attempt, the TOE performs a bit-wise comparison of the expected identifier with the Subject Alternative Name (SAN) if it's available else it's compared to the Common Name (CN) in the peer's certificate (which the peers sends to the TOE). If the expected identifier does not match the SAN or CN, then the connection is not established.

FCS_TLSS_EXT.2 - TLS Server Protocol with Mutual Authentication

6.3 Identification and Authentication (FIA)

6.3.1 Password Management

The TOE has a local database of user with their corresponding passwords. The passwords are composed of any combination of:

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

The minimum password length in the TOE is configurable by the Security Administrator, and supports at least 15 characters long. In addition to locally managed passwords, the TOE supports communication with a remote authentication server to facilitate remote management over TLS connection.

FIA_PMG_EXT.1 - Password Management

6.3.2 Authentication Mechanism

The TOE requires that all users are successfully identified and authenticated before allowing any TSF mediated actions to be performed, with the exception of the following:

- Displaying the warning banner at the local console
- ICMP echo from a remote IT entity

The TOE provides two logon processes that exercises different identification and authentication mechanisms:

- Local – which is exercised via the local console.
- Remote – which is exercised via the RESTful interface over a HTTPS channel.

For Local access, the TOE expects the Administrators to use the serial (local) console to provide the appropriate credentials (User ID and Password). The TOE checks the credentials against its local user database. Once the Administrator is successfully identified and authenticated, the Administrator will have access to the TOE Serial Console Interface (SCI), allowing the Administrator to manage the TOE. While the Administrator users enter their password at the local console, the TOE does not display any characters in the password field. If the User ID or Password is wrongly entered, the TOE displays "Login failed. Please, try again". The same message is displayed if the User ID doesn't exist in the local user database.

For Remote access, the TOE provides a RESTful API that allows for a remote entity to perform the Identification and Authentication (I&A) of the Administrator user, and the TOE validates the response of the entity. The RESTful API is established over HTTPS connection and it works as follows:

- An Administrator user, using a remote console, requests an Access Token from the remote I&A server. The server, depending on the permissions of the Administrator user, grants that access token.
- The Administrator user sends the management functionality command and the Access Token to the TOE over a RESTful request.
- The TOE extracts the Access Token, and establishes another RESTful request to the I&A server to verify validity of the Access Token.
- When the I&A verifies the validity of the Access Token and TOE receives the response, the TOE ends the RESTful interface, and processes the response from the I&A server.
- If the response is valid, the TOE extracts the Administrator user privileges, and processes the management functionality command.
- Once the command is processed and the response sent back to the remote console, the original RESTful interface is terminated.

This process is repeated for every remote management operation is performed by establishing RESTful requests.

FIA_UIA_EXT.1 - User Identification and Authentication

FIA_UAU_EXT.2 - Password-based Authentication Mechanism

FIA_UAU.7 - Protected Authentication Feedback

6.3.3 X.509 Certificates

The TOE uses X.509v3 certificate as specified in RFC 5280 in order to support the secure communication over TLS with

- External audit servers.

- Authentication server using RESTful interface.
- Remote management using RESTful interface

The TOE is capable of generating Certificate Signing Request (CSR). The TOE allows for the CSR to include all the information listed in FIA_X509_EXT.3.1. The TOE allows its generated CSRs to have device specific information, such as DNS name and IP address, stored as the Subject Alternative Name (SAN), as configured by the Security Administrator.

The TOE stores CSRs in PKCS#10 PEM format. The TOE displays the CSRs to the SCI for the Security Administrator to transmit to a Certificate Authority (CA). This transmission is performed manually by the Security Administrator. When the Security Administrator receives a signed certificate back from the CA, the Security Administrator imports the certificate, along with the CA chain, into the TOE localhost. CSRs generate by the TOE are required to be associated with a private/public key pair. The TOE also allows for externally generated keys that can be associated with a TOE's CSR or a certificate.

All the certificates are imported manually into the TOE. TOE's localhost certificate is used for all PKI-based communication. The TOE allows the Security Administrator to import the entire CA chain of (peer) remote hosts that 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 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.

The X.509 certificates are validated using the certificate path validation algorithm specified 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.
- Revocation status is checked.
- Issuer name of X matches the subject name of X+1.
- Name constraints are checked.
- Policy OIDs are checked.
- Policy constraints are checked; issuers are ensured to have CA signing bits.
- Path length is checked.
- Critical extensions are processed.

The TOE will not allow any certificate to be imported it if it hasn't passed validation. Online Certificate Status Protocol (OCSP) is used to check revocation status of certificates during 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 doesn't 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.

FIA_X509_EXT.1 - X.509 Certificate Validation

FIA_X509_EXT.2 - X.509 Certificate Authentication

FIA_X509_EXT.3 - X.509 Certificate Requests

6.4 Security Management (FMT)

6.4.1 Management of Trusted Updates

The TOE restricts the ability to perform manual update to Security Administrators. Manual updates are initiated via the SCI in the TOE.

FMT_MOF.1(1)/TrustedUpdate

6.4.2 Management of TSF Data

No administrative interfaces are available prior to successful authentication. The following data can be generated, modified, or overwritten by the Administrator user depending on administrator's role:

- Internal user database (including salted hashed passwords)
- Internally generated cryptographic keys
- Generated CSR
- Imported certificates
- Locally stored audit records

All management of data listed above is available through the SCI, and some is available through the RESTful interface.

FMT_MTD.1 - Management of TSF Data

FMT_MTD.1/AdminAct - Management of TSF Data

6.4.3 Management Functions

The TOE provides all the capabilities necessary to securely be managed. The Security Administrators can connect to the TOE to perform these functions using the SCI via the local console. Some of these functions are accessible using the RESTful interface via a remote console.

The specific management capabilities available from the TOE include:

- Local and remote administration of the TOE and the services provided by the TOE.
- The ability to manage the warning banner message and content – allows the Security Administrator the ability to define warning banner that is displayed prior to establishing a session (note this applies to the interactive (human) users via the local console.
- The ability to manage the time limits of session inactivity on the serial console. Due to RESTful interface, session inactivity is not applicable to remote management.
- The ability to update the TOE's software. The validity of the image is provided using digital signature prior to installing the update.
- The ability to manage audit behavior and the audit logs which allows the Security Administrator to configure, view, and clear the audit logs.
- The ability to generate cryptographic keys and CSRs, and importing certificates.
- The ability to manage password complexity.
- The ability to manage the NTP server synchronization parameters.

FMT_SMF.1 - Specification of Management Functions

6.4.4 Management Roles

The TOE maintains different type of administrators. These administrators have different privileges to administer the TOE locally and remotely. The TOE performs role-based authorization, using the TOE authentication and authorization mechanisms. The different administrator types supported by the TOE are the following:

- Security Administrator: managing all security related services and configuration – Add administrator users and roles; modify password policy; enable/disable audit behavior; managing cryptographic keys, CSRs, and certificates; software updates; managing banner.
- Network Administrator: managing all network related services and configurations – viewing and setting network configurations; setting route tables.
- KSI Administrator: managing all KSI related services and configurations.

For the purpose of this evaluation, the TOE's Security and Network Administrator combined are equivalent to the NDcPP Security Administrator. The TOE is designed to authenticate and authorized all access to the SCI using a username and password when connected via the local console. For remote access, the TOE communicates with an authentication server before processing a RESTful interface request from a remote console.

FMT_SMR.2 - Restrictions on Security Roles

6.4.5 Management of Security Functions Behavior

As stated throughout this section, the TOE allows only the Security Administrator to manage the behavior of functions that handle audit data, and to enable and disable services (such as logging).

FMT_MOF.1(2)/Audit - Management of Security Functions Behavior

FMT_MOF.1(2)/AdminAct - Management of Security Functions Behavior

6.5 Protection of TSF (FPT)

6.5.1 Protection of Stored Keys

The root or top-level key-encrypting key is an AES 256 key, derived from a special hardware-based secret called the OTPMK. The OTPMK is implemented in specially-designed circuity by the chip manufacturer. Guardtime uses this key value to protect long-term keys stored on the TOE at rest. All pre-shared, symmetric, and private keys that are stored in the TOE are stored and protected at rest using AES 256 encryption format. 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 TOE is done using a plaintext key, the TOE deallocates the memory location back to the kernel.

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

6.5.2 Protection of Administrator Passwords

The TOE makes use of the Password Based Key Derivation Function 2 (PBKDF2), as outlined in RFC 2898 and NIST SP 800-132. The PBKDF2 uses HMAC-SHA-256 cryptographic hash function and random salt (generated from the CTR_DRBG) to transform the plaintext password into a pseudo-random value before the TOE stores it. No user or administrator is able to read the plaintext password through "normal" interfaces of the TOE.

FPT_APW_EXT.1 - Protection of Administrator Passwords

6.5.3 Self-Test

The BL has a 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 BL. During initial start-up of the BL, each firmware image gets verified/authenticated prior to executing its code to make sure it did not get modified while sitting at rest. If one of the images does not verify correctly, then the BL reports that as a fault. If both images do not verify correctly, the BL does not boot up.

In addition to performing authentication and decryption for the firmware files, the extended boot also performs a series of Power On Self Tests (POST). The following are the list of POST tests being performed by the TOE's boot firmware:

- **RAM March Test**
  This test verifies RAM memory by writing an incrementing integer value as well as a decrementing integer value into every memory cell. After each write, it reads the value back and compares it to an expected value.

- **Error-correcting Code Memory Test**
  This test consists of two parts, single-bit error correction and multi-bit error detection. For the single-bit error correction test, one bit error gets injected to each DDR SRAM device. Once the single-bit error is injected, the test verifies to make sure the error is detected and corrected. Similarly for the multi-bit error detection test, a multi-bit error gets injected to each DDR SRAM device. However, for the multi-bit error verification, this test only verifies to make sure the multi-bit error is detected as correction cannot be done on multi-bit error.

- **Special Purpose Register (SPR) Test**
  This test verifies each bit that can be read and written in the Special Purpose Registers (SPRs), with the exception to those bits that may put the processor in an unstable state. The special purpose registers being verified under this test include 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 in an unstable state, SPRs that are read-only, write-only, write-once and read-clear are also not included as part of this test.

- **General Purpose Register Test**
  This test verifies each bit in each general purpose register to make sure it can be read and

written correctly. The test starts with filling the general purpose register under test with 1s. Next, it loops to shift a 0 into each bit position and verifies to make sure the data read back matches the expected value.

- **Condition Register Test**
  This test verifies the Condition Registers (CR) to make sure they're working correctly. It first sets certain bit(s) in each CR and reads back the CR to verify that the bit(s) were written. Next, it performs instructions that set bits in the CR(s) as a side effect and verifies that the associated CR(s) bits were set with the expected value/state.

- **Branch Test**
  This test verifies to make sure the "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**
  This 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**
  This test verifies that the load and store instructions are functioning as expected. It performs loads and stores between the General Purpose Registers and RAM using many variations of load and store instructions with different data patterns.

- **Floating Point Unit Register Test**
  This test verifies the functioning of the floating point registers. It writes multiple patterns to each floating pointer register and verifies to make sure the value read back matches the written pattern.

- **Floating Point Unit Load and Store Test**
  This 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**
  This test verifies every floating point math instruction to make sure the 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**
  This 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's at

the expected rate.

- **Data Cache Test**
  This test verifies to make sure the Data-Cache is functioning as expected. With memory coherency enabled, it verifies that the written test data pattern get flushed from the cache into memory when the data cache is invalidated. With memory coherency disabled, it verifies the written test data pattern does not get written to memory.

- **RNG Test**
  Consists of four tests from FIPS 140: Monobit, Poker, Runs, and Long Run.

- **SHA Test**
  This test verifies the hardware SHA engine to make sure it's 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.

- **AES CBC Test**
  This test verifies the hardware AES engine, using the Cipher Block Chaining (CBC) mode, to make sure it's 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.

- **HMAC Test**
  This test verifies the hardware HMAC engine to make sure it's functioning as expected. It performs HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512 on a sample data buffer and compares the returned results with known answers to make sure they are the same.

- **AES GCM Test**
  This test verifies the hardware AES engine, using the Galois/Counter Mode (GCM), to make sure it's 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 the GCM 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.

- **Sign and Verification Test**
  This test verifies the SEC RSA and ECDSA modes by signing data with a known key and then verifying that the correct signature was generated.

If all the above tests passed, the overall status on the POST will indicate Ok. Otherwise, any failed POST test will trigger the BL to display Failure on its "Power On Self Test" status. In addition, any errors encountered are printed out at bootup.

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

FPT_TST_EXT.1 - TSF Testing

6.5.4 Trusted Updates

The TOE does not provide automatic updates to the software version running on the TOE. Only the Security Administrator is capable of query the current executing and most recently installed versions of the TOE software image; both should match since the TOE reboots itself after is done installing updated software.

When software needs to be updated, the Security Administrator commands the TOE to initiate the update process. The update process starts with the TOE requesting confirmation from the Security Administrator to proceed with the update process. Upon confirmation, the TOE automatically proceeds with the rest of the update process without requiring input from the Security Administrator.

The TOE downloads a signed and encrypted package (containing the software update) from a pre-set up HTTP (or HTTPS) server. The TOE uses ECDSA-521 to perform digital signature verification and AES 256 to decrypt the image. The keys required for those cryptographic operations are pre-installed in the TOE during manufacturing of the TOE. The TOE proceeds to install the software in the primary location, and then reboots itself. Coming up from reboot, the TOE verifies the software installed in the primary location. If primary location software is verifies properly, the TOE copies the primary location software into the secondary location, and reboots again. If the primary location software verification fails, the TOE copies the secondary location software into the primary location, and reboots itself.

If any failures occur, including digital signature verification or decryption, the TOE aborts the software update process, and outputs error messages to the serial console and records relevant audit logs. In the event of a fail software update attempt, the Security Administrator will use the information displayed in the Serial Console and the logs to troubleshoot the incident, and retry again.

FPT_TUD_EXT.1 - Trusted Update

6.5.5 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 a primary and secondary NTP servers. The Security Administrator can configure the NTP synchronization configuration. The TOE does not provide the Security Administrator to manually set the date and time of the TOE.

FPT_STM.1 - Reliable Time Stamps

6.6 TOE Access (FTA)

6.6.1 Session Protection

The TOE allows the Security Administrator the ability to configure the inactivity time out period of all local sessions. At the end of this period, the TOE terminates the session. The TOE allows the administrators to initiate the termination of their own local sessions by commanding the TOE to exit, reboot, or shutdown.

For remote management of the TOE, the RESTful interface is utilized. The RESTful Interface does not have a concept of interactive session since each request to the API is a self-contained, identified, and authenticated request. This can be interpreted as the “remote interactive session” being terminated immediately after the request is serviced and is never open for any measurable period of inactivity. Therefore, inactivity period and session termination is not applicable to the TOE.

FTA_SSL_EXT.1 - TSF-initiated Session Locking

FTA_SSL.3 -TSF-initiated Termination

FTA_SSL.4 -User-initiated Termination

6.6.2 Access Banner

The TOE displays a custom login banner prior to an administrative session via the local console. The Security Administrator is the only one allowed to configure the banner in the TOE. For remote management of the TOE, the RESTful interface is utilized. Due to the nature of the RESTful interface, the login banner is not displayed. However, the Security Administrator is able to configure the content of the login banner from the remote console.

FTA_TAB.1 - Default TOE Access Banners

6.7 Trusted Path / Channels (FTP)

6.7.1 Trusted Channel

The TOE protects communication with audit server and authentication server IT entities via TLS and HTTPS, respectively. Mutual authentication is performed for all TLS based communication, which includes HTTPS. This protects the data from disclosure by encrypting the packages that transmitted between the TOE and the authorized IT entities.

FTP_ITC.1 - Inter-TSF Trusted Channel

6.7.2 Trusted Path

All remote administrative communication is performed via the RESTful Interface, which uses the HTTPS protocol. The Security 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. As in the trusted channel, HTTPS connections made for a trusted path uses TLS mutual authentication.

FTP_TRP.1 - Trusted Path