Forum Systems, Inc.
Sentry v8.1.641

Security Target

Document Version: 1.2

Prepared for:

FORUMSYSTEMS
A Crosscheck Networks Company

Forum Systems, Inc.
199 Wells Avenue, Suite 105
Newton, MA 02459
United States of America

Phone: +1 781 791-7510
Email: info@forumsys.com
http://www.forumsys.com

Prepared by:

Corsec

Corsec Security, Inc.
13135 Lee Jackson Memorial Hwy., Suite 220
Fairfax, VA 22033
United States of America

Phone: +1 703 267 6050
Email: info@corsec.com
http://www.corsec.com

Table of Contents

Table of Figures

List of Tables

1 Introduction

This section identifies the Security Target (ST), Target of Evaluation (TOE), and the organization of the ST. The TOE is the Forum Sentry v8.1.641, and will hereafter also be referred to as the TOE. The TOE is an application-layer gateway server that provides application-firewall functionality via content inspection of common protocols. The Sentry can then protect against application-level attacks and enforce access control rules based on the information discovered in the application-layer traffic.

1.1 Purpose

This ST is divided into nine sections, as follows:

- Introduction (Section 1) – Provides a brief summary of the ST contents and describes the organization of other sections within this document. It also provides an overview of the TOE security functions and describes the physical and logical scope for the TOE, as well as the ST and TOE references.
- Conformance Claims (Section 2) – Provides the identification of any Common Criteria (CC) and Protection Profile package claims. It also identifies whether the ST contains extended security requirements.
- Security Problem (Section 3) – Describes the threats, organizational security policies, and assumptions that pertain to the TOE and its environment.
- Security Objectives (Section 4) – Identifies the security objectives that are satisfied by the TOE and its environment.
- Extended Components (Section 5) – Identifies new components (extended Security Functional Requirements (SFRs) and extended Security Assurance Requirements (SARs)) that are not included in CC Part 2 or CC Part 3.
- Security Requirements (Section 6) – Presents the SFRs and SARs met by the TOE.
- TOE Summary Specification (Section 7) – Describes the security functions provided by the TOE that satisfy the security functional requirements and objectives.
- Rationale (Section 8) - Presents the rationale for the SFR dependencies as to their consistency, completeness, and suitability.
- Acronyms and Terms (Section 9) – Defines the acronyms and terminology used within this ST.

1.2 Security Target and TOE References

Table 1 below shows the ST and TOE references.

Table 1 ST and TOE References

ST Title	Forum Systems, Inc. Sentry v8.1.641 Security Target
ST Version	Version 1.2
ST Author	Corsec Security, Inc.
ST Publication Date	2014-06-02
TOE Reference	Forum Sentry v8.1.641
FIPS[1] 140-2 Status	Level 2, Validated crypto module, Certificate No. 1743

[1] FIPS – Federal Information Processing Standard

1.3 TOE Overview

The TOE Overview summarizes the usage and major security features of the TOE. The TOE Overview provides a context for the TOE evaluation by identifying the TOE type, describing the product, and defining the specific evaluated configuration.

The TOE is a hardware TOE that includes the physical appliance as well as the applications running on the appliances. The Forum Sentry provides application gateway functionality for a variety of web-based (such as HTTP[2] and XML[3]) and non-web-based (such as SFTP[4] and LDAP[5]) protocols[6]. This functionality is implemented by an engine that examines application-layer traffic elements to apply policy to traffic. Policy takes the form of application-firewall rules, an antivirus engine, and pattern recognition rules. This allows the product to protect against attacks and to enforce access control rules based on the information within XML requests. The TOE can use cryptography to protect information being sent across the network as well.

The TOE includes a hardware appliance, a Hardware Security Module (HSM), and a web Graphical User Interface (GUI) that allows administrators to configure the policies designed to protect the network and network traffic. Additionally the TOE offers a Command Line Interface (CLI) that offers a broad set of functionality for configuring the appliance, access controls, and several other security features of the product. The TOE offers various application firewall services, such as XML Gateway. These features are all present in the TOE software, but must be activated through license keys.

1.3.1 Forum Sentry Appliance

The TOE hardware is a rack-mountable general purpose computing platform. Installed on the hardware is the Forum Operating System (OS) and Forum Sentry software. The combination of hardware appliance and software receives traffic flowing through the network, inspects the headers to determine the traffic type, and performs deep inspection of supported traffic types. Deep inspection is performed by reading the payload data from supported traffic and comparing the contents to policy rules and signatures stored on the TOE that indicate whether traffic is of a suspicious or malicious nature. If the TOE determines that traffic is undesirable, the TOE can block the connection from proceeding, or simply record that the event occurred.

The TOE allows administrators to define users. Users are individuals on the network connecting to web services controlled by the TOE. Policies can define access control rules. Access controls can be used to limit access to specified services on the network for specified users. The TOE also designates administrator accounts, which can be used to access the CLI or GUI to view the TOE logs or configuration. Administrator accounts are logically separate from user accounts, and can't be used to control access to web services. Administrators can also gain enable access via the CLI, or privileged access to the GUI, which provides full access to modify the TOE configuration. Although the CLI can be used to configure a broad range of TOE capabilities, policies can only be managed via the GUI.

1.3.2 Hardware Security Module

The TOE comes with a Thales nShield 6000e F3 HSM PCIe[7] card. The HSM is a hardware device designed to efficiently perform cryptographic calculations. The TOE uses the HSM to implement RSA[8], DSA[9], AES[10], and 3-key Triple-DES[11] cryptographic operations, including encryption, decryption,

[2] HTTP – Hypertext Transfer Protocol
[3] XML – Extensible Markup Language
[4] SFTP – Secure File Transfer Protocol
[5] LDAP – Lightweight Directory Access Protocol
[6] A list of all protocols supported can be found on Forum's website http://www.forumsys.com
[7] PCIe – Peripheral Component Interconnect Express
[8] RSA – Rivest, Shamir, and Adelman (cryptographic algorithm)
[9] DSA – Digital Signature Algorithm

signature generation, and signature verification. The TOE also uses the HSM to generate cryptographic keys for use with these algorithms.

The Forum OS and applications running on the OS make use of the HSM's cryptographic functionality much the same way they would make use of a cryptographic library. The HSM provides all encryption, decryption, and other cryptographic operations, which are then used by functionality in the OS and other applications to provide security services for the TOE.

The HSM uses a global key (Security World key) to encrypt all other keys stored on the TOE. The global key is stored on a separate smart card and is encrypted with an administrator's password. The smart card and reader are outside of the scope of this evaluation.

Forum Sentry uses only FIPS-validated cryptography provided by a FIPS-140-2 validated cryptographic module (certificate #1743).

1.3.3 TOE Environment

The TOE is composed of the physical hardware, the HSM, the Operating System (OS), and application software running on the physical hardware. The hardware is a general purpose server. The HSM is a pluggable cryptographic accelerator card that provides cryptographic services for the TOE. The software includes the OS and TOE application software that provide all of the functionality of the TOE. There is also a smart card reader and smart card storing the Security World key for the TOE, but the smart card and reader are both considered to be part of the TOE Environment.

It is assumed that there will be no untrusted users or software on the TOE components. In addition, the TOE components are intended to be deployed in a physically secured cabinet, room, or data center with the appropriate level of physical access control and physical protection (e.g. badge access, fire control, locks, alarms, etc.).

The TOE can be deployed in an inline deployment or a one-port deployment. In an inline deployment, the TOE sits on the network between web clients and web services servers and receives all traffic from both parties en route, as depicted in Figure 1 below.

[10] AES – Advanced Encryption Standard
[11] DES – Data Encryption Standard

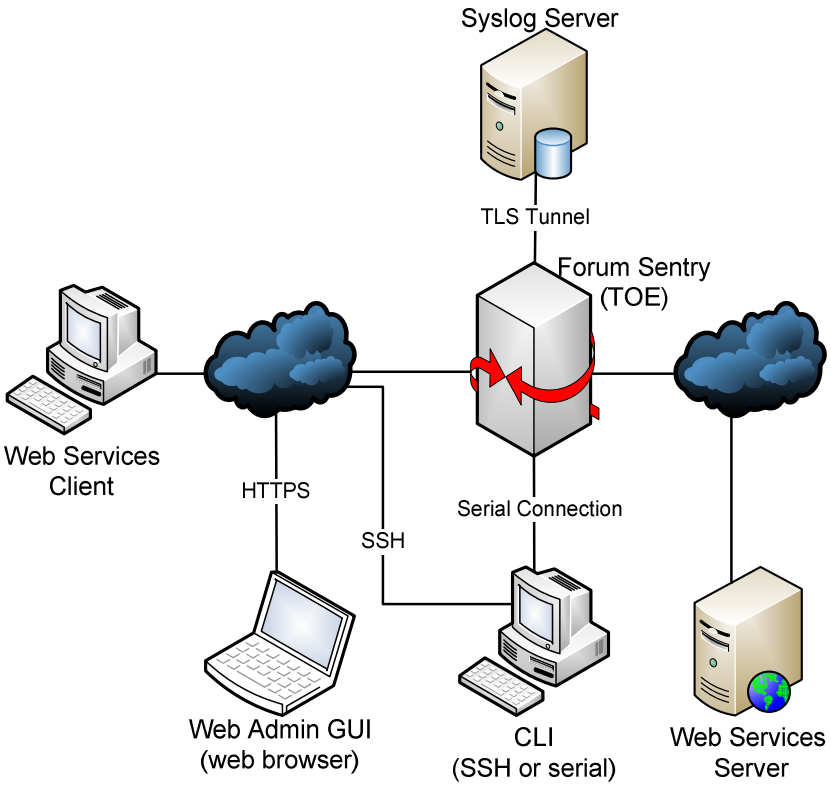

Figure 1 Inline Deployment

In a one-port deployment, the TOE sits between the web clients and web services servers, but traffic is redirected to the TOE by a router or switch before being sent to the server, rather than the TOE sitting as one of the nodes between client and server. This deployment is depicted in Figure 2 below.

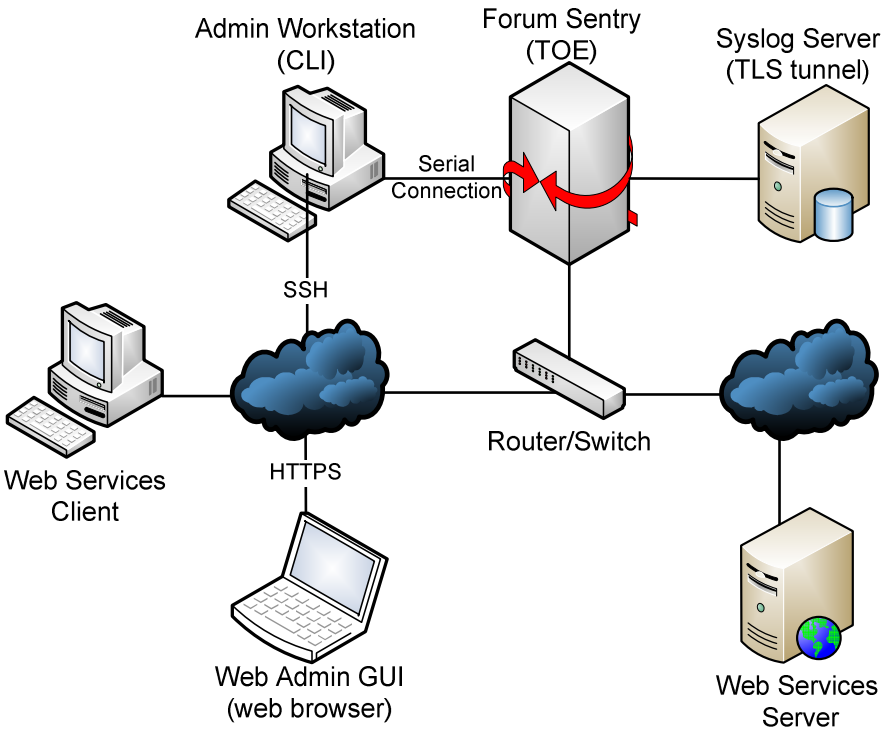

Figure 2 One-Port Deployment

In the two typical deployments, traffic being sent from web services clients to web services servers is either sent through the TOE or forwarded to the TOE. The TOE examines the traffic and responds according to the configured policies. Administrators log onto the CLI from a workstation running either a terminal emulator (serial connection) or an SSH[12] client (SSH connection). Administrators log onto the GUI by using a web browser and an encrypted Secure Hypertext Transfer Protocol (HTTPS) connection. Administrator actions and certain policy-driven events are logged and sent across a TLS (Transport Layer Security) tunnel to the syslog server.

1.4 TOE Description

This section primarily addresses the physical and logical components of the TOE included in the evaluation.

1.4.1 Physical Scope

Figure 3 illustrates the physical scope and the physical boundary of the TOE. The TOE is a software and hardware device. The hardware is the Forum Sentry 4564 appliance, a general purpose server platform running an Intel Xeon E5620 processor. The operating system is a highly customized version of Linux. Included with the general purpose computing hardware is a Thales nShield 6000e F3 HSM. The HSM provides cryptographic acceleration and key storage capabilities. The Forum Sentry Application software is a custom built application that provides the majority of the TOE's functionality.

[12] SSH – Secure Shell

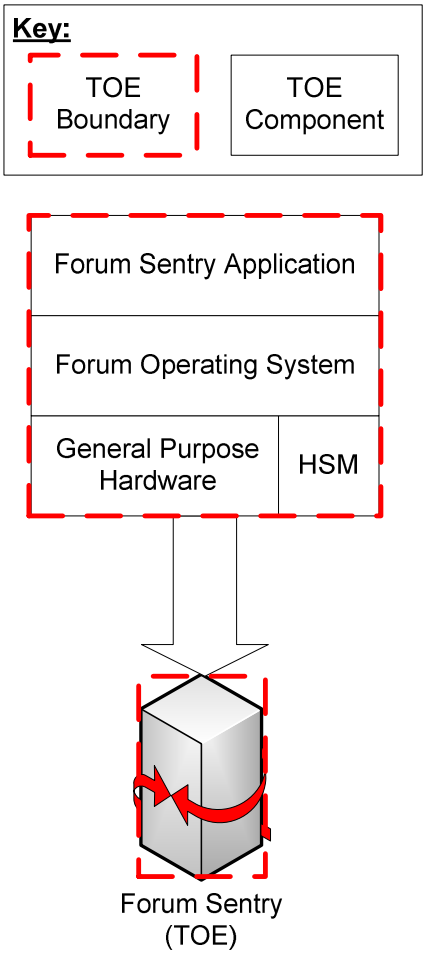

Figure 3 Physical TOE Boundary

The TOE Boundary includes all the Forum developed parts of the Sentry v8.1.641 product. Any third party source code or software that Sentry v8.1.641 has modified is considered to be TOE Software. The TOE Boundary does not include any of the environmental components shown above in the TOE deployment diagrams:

- The web browser used to access the GUI,
- The terminal client or SSH client used to access the CLI,
- The administrator workstation hosting the web browser, terminal client, and SSH client,
- The web services client,
- The web services server,
- The router/switch or any other network components, and
- The syslog server.

1.4.1.1 Guidance Documentation

Table 2 lists the TOE Guidance Documentation to install, configure, and maintain the TOE.

Table 2 Guidance Documentation

Document Name	Description
Forum Systems Sentry Version 8.1Command Line Reference	Provides a manifest of CLI commands available, the expected syntax, and the results of executing each command. Also provides sample return values for each command.
Forum Systems Sentry Version 8.1 Guide to Security Worlds	Gives an overview of how to work with the HSM cards on the TOE.
Forum Systems Sentry Version 8.1 Logging Guide	Gives an overview of the logging functionality available on the TOE.
Forum Systems Sentry Version 8.1 Network Policies Guide	Gives an overview of configuring HTTP policies on the TOE.
Forum Systems Sentry Version 8.1 Task Management Guide	Gives an overview of working with tasks on the GUI.
Forum Systems Sentry Version 8.1 Web-based Administration Guide	Gives an overview of how to use the web GUI.
Forum Systems Sentry Version 8.1 WSDL[13] Policies Guide	Gives an overview of how to configure WSDL policies on the TOE.
Forum Systems Sentry Version 8.1 XML Policies Guide	Gives an overview of how to configure XML policies on the TOE.
Forum Systems Sentry Version 8.1 System Management Guide	Gives an overview of general administrative tasks for the TOE.
Forum Systems Sentry Version 8.1Access Control Guide	Gives an overview of how to manage authentication and access control on the TOE.
Forum Systems Sentry Version 8.1Security Policies and PKI Guide	Gives instruction for managing TLS, encryption, and certificates on the TOE.
Forum Systems Sentry Version 8.1Software Installation Guide	Gives details on how to install the TOE software onto the TOE hardware.
Guidance Supplement v0.4	Contains information regarding specific configuration for the TOE evaluated configuration.

1.4.2 Logical Scope

The logical boundary of the TOE will be broken down into the following security classes which are further described in sections 6 and 7 of this ST. The logical scope also provides the description of the security features of the TOE. The security functional requirements implemented by the TOE are usefully grouped under the following Security Function Classes:

1.4.2.1 Security Audit

The TOE generates audit records for security-relevant actions of the authorized administrators within the GUI and CLI. The TOE provides an authorized administrator access to view the audit logs via the CLI and GUI. The TOE records the identity of the administrator or user responsible for logged events–where applicable. All logs are backed up to a syslog server via a secure channel.

[13] WSDL – Web Service Description Language

1.4.2.2 Cryptographic Support

The Cryptographic Support of the TSF[14] function provides cryptographic functions to secure communications for GUI and CLI sessions. TLS, HTTPS, and SSH are used to secure management communications sessions. The TOE uses encryption to protect locally stored passwords. In addition, the TOE provides a variety of cryptographic algorithms for its own use.

1.4.2.3 User Data Protection

The TOE stores network data (packets) within volatile memory while the data is being used by the TOE. Once the TOE finishes using the packet data, or if the TOE is rebooted, the memory space is de-allocated and zeroized to prevent third parties from being able to read that memory space.

1.4.2.4 Identification and Authentication

The TOE provides functionality that requires administrators to verify their claimed identity. The Identification and Authentication TSF ensures that only legitimate administrators can gain access to the configuration settings and management settings of the TOE. Administrators must log in with a valid user name and password before the TOE will allow administrators to manage the TOE. The TOE requires administrators to use strong passwords. No feedback is presented to administrators when they are entering their passwords at the login prompt of the local console.

1.4.2.5 Security Management

The TOE provides a CLI and GUI that administrators can use to configure the TOE and manage the security functionality. The Security Management function specifies how administrators can access management of the TOE components.

1.4.2.6 Protection of the TSF

The TOE protects cryptographic keys from being read by external entities by only providing access to them via the cryptographic code within the TOE. All passwords are stored encrypted to prevent unauthorized reading of passwords. At startup, the TOE runs a suite of self-tests that verify the correct operation of all cryptographic code.

The TOE has the capability to identify whether an update is from a trusted source by calculating a fingerprint of the update code.

The TOE also provides reliable time stamps for its own use.

1.4.2.7 TOE Access

The TOE terminates local and remote management sessions after an administrator-configurable time period of inactivity. Administrators can also end their session voluntarily. After an administrator's session is terminated, that user must log in again to regain access to TOE functionality. A login banner is displayed for users at the login screen of the GUI and at the login prompt for the CLI.

1.4.2.8 Trusted Path/Channels

The TOE implements a trusted TLS tunnel between itself and a remote syslog server in order to protect syslog traffic as it is being sent to the server. Additionally, the TOE provides trusted paths between administrators and the CLI via an SSH tunnel, and between administrators and the GUI via an HTTPS/TLS tunnel. All tunnels are encrypted with AES.

1.4.3 Product Physical and Logical Features and Functionality not included in the TOE

Features and Functionality that are not part of the evaluated configuration of the TOE are:

[14] TSF – TOE Security Functionality

- The application-firewall functionality provided by the TOE was not tested as part of the evaluated configuration of the TOE.
- XML encryption and decryption policies only allow the support of FIPS-approved algorithms. All non-approved algorithms are excluded.
- FTP[15] and OpenPGP[16]
- SSL[17]v2 and SSLv3 are disabled. Only TLS with FIPS-approved cryptography is allowed.
- Remote management via third-party WSDL management clients
- Global Device Management
- Sentry Virtual Appliance
- Use of an NTP (Network Time Protocol) server is not part of the evaluated configuration.

[15] FTP – File Transfer Protocol
[16] PGP – Pretty Good Privacy
[17] SSL – Secure Sockets Layer

2 Conformance Claims

This section provides the identification for any CC and Protection Profile (PP) conformance claims. Rationale is provided for any extensions or augmentations to the conformance claims. Rationale for CC and PP conformance claims can be found in Section 8.1.

Table 3 CC and PP Conformance

Common Criteria (CC) Identification and Conformance	Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 4, September 2012; CC Part 2 extended; CC Part 3 conformant; PP claim Network Devices Protection Profile conformant; Parts 2 and 3 Interpretations of the CEM[18] as of 2013-04-05 were reviewed, and no interpretations apply to the claims made in this ST.
PP Identification	Exact Conformance[19] to Protection Profile for Network Devices v1.1, June 8, 2012 (also referred to as the Network Devices Protection Profile (NDPP)).

[18] Common Evaluation Methodology

[19] Exact Conformance is a type of Strict Conformance such that the set of SFRs and the SPD Objectives are exactly as presented within the accepted NDPP without changes.

3 Security Problem

This section describes the security aspects of the environment in which the TOE will be used and the manner in which the TOE is expected to be employed. It provides the statement of the TOE security environment, which identifies and explains all:

- Known and presumed threats countered by either the TOE or by the security environment
- Organizational security policies with which the TOE must comply
- Assumptions about the secure usage of the TOE, including physical, personnel, and connectivity aspects

3.1 Threats to Security

This section identifies the threats to the IT[20] assets against which protection is required by the TOE or by the security environment. The threat agents are divided into two categories:

- Attackers who are not TOE users: They have public knowledge of how the TOE operates and are assumed to possess a low skill level, limited resources to alter TOE configuration settings or parameters and no physical access to the TOE.
- TOE administrative users: They have extensive knowledge of how the TOE operates and are assumed to possess a high skill level, moderate resources to alter TOE configuration settings or parameters and physical access to the TOE. (TOE administrative users are, however, assumed to operate in a trusted manner.)

Both are assumed to have a low level of motivation. The IT assets requiring protection are the TSF and user data saved on the TOE. Removal, diminution and mitigation of the threats are through the objectives identified in Section 4 Security Objectives. Table 4 below lists the applicable threats.

Table 4 Threats

Name	Description
T.ADMIN_ERROR	An administrator may unintentionally install or configure the TOE incorrectly, resulting in ineffective security mechanisms.
T.TSF_FAILURE	Security mechanisms of the TOE may fail, leading to a compromise of the TSF.
T.UNDETECTED_ACTIONS	Malicious remote users or external IT entities may take actions that adversely affect the security of the TOE. These actions may remain undetected and thus their effects cannot be effectively mitigated.
T.UNAUTHORIZED_ACCESS	A user may gain unauthorized access to the TOE data and TOE executable code. A malicious user, process, or external IT entity may masquerade as an authorized entity in order to gain unauthorized access to data or TOE resources. A malicious user, process, or external IT entity may misrepresent itself as the TOE to obtain identification and authentication data.
T.UNAUTHORIZED_UPDATE	A malicious party attempts to supply the end user with an update to the product that may compromise the security features of the TOE.
T.USER_DATA_REUSE	User data may be inadvertently sent to a destination not intended by the original sender.

[20] IT – Information Technology

3.2 Organizational Security Policies

An Organizational Security Policy (OSP) is a set of security rules, procedures, or guidelines imposed by an organization on the operational environment of the TOE. Table 5 below lists the OSPs that are presumed to be imposed upon the TOE or its operational environment by any organization implementing the TOE in the CC evaluated configuration.

Table 5 Organizational Security Policies

Name	Description
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.

3.3 Assumptions

This section describes the security aspects of the intended environment for the evaluated TOE. The operational environment must be managed in accordance with assurance requirement documentation for delivery, operation, and user guidance. Table 6 lists the specific conditions that are required to ensure the security of the TOE and are assumed to exist in an environment where this TOE is employed.

Table 6 Assumptions

Name	Description
A.NO_GENERAL_PURPOSE	It is assumed that 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.
A.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the environment.
A.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.

4 Security Objectives

Security objectives are concise, abstract statements of the intended solution to the problem defined by the security problem definition (see Section 3). The set of security objectives for a TOE form a high-level solution to the security problem. This high-level solution is divided into two part-wise solutions: the security objectives for the TOE, and the security objectives for the TOE's operational environment. This section identifies the security objectives for the TOE and its supporting environment.

4.1 Security Objectives for the TOE

The specific security objectives for the TOE are as follows:

Table 7 Security Objectives for the TOE

Name	Description
O.PROTECTED_COMMUNICATIONS	The TOE will provide protected communication channels for administrators, other parts of a distributed TOE, and authorized IT entities.
O.VERIFIABLE_UPDATES	The TOE will provide the capability to help ensure that any updates to the TOE can be verified by the Administrator to be unaltered and (optionally) from a trusted source.
O.SYSTEM_MONITORING	The TOE will provide the capability to generate audit data and send those data to an external IT entity.
O.DISPLAY_BANNER	The TOE will display an advisory warning regarding use of the TOE.
O.TOE_ADMINISTRATION	The TOE will provide mechanisms to ensure that only administrators are able to log in and configure the TOE, and provide protections for logged-in administrators.
O.RESIDUAL_INFORMATION_CLEARING	The TOE will ensure that any data contained in a protected resource is not available when the resource is reallocated.
O.SESSION_LOCK	The TOE shall provide mechanisms that mitigate the risk of unattended sessions being hijacked.
O.TSF_SELF_TEST	The TOE will provide the capability to test some subset of its security functionality to ensure it is operating properly.

4.2 Security Objectives for the Operational Environment

4.2.1 IT Security Objectives

The following IT security objectives are to be satisfied by the environment:

Table 8 IT Security Objectives

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

4.2.2 Non-IT Security Objectives

The following non-IT environment security objectives are to be satisfied without imposing technical requirements on the TOE. That is, they will not require the implementation of functions in the TOE hardware and/or software. Thus, they will be satisfied largely through application of procedural or administrative measures.

Table 9 Non-IT Security Objectives

Name	Description
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

5 Extended Components

This section defines the extended SFRs and extended SARs met by the TOE. These requirements are presented following the conventions identified in Section 6.1.

5.1 Extended TOE Security Functional Components

This section specifies the extended SFRs for the TOE. The extended SFRs are organized by class. Table 10 identifies all extended SFRs implemented by the TOE.

Table 10 Extended TOE Security Functional Requirements

Name	Description
FAU_STG_EXT.1	Extended: External audit trail storage
FCS_CKM_EXT.4	Extended: Cryptographic key destruction
FCS_HTTPS_EXT.1	Extended: HTTPS
FCS_RBG_EXT.1	Extended: Cryptographic operation (Random bit generation)
FCS_SSH_EXT.1	Extended: SSH
FCS_TLS_EXT.1	Extended: TLS
FIA_PMG_EXT.1	Extended: Password management
FIA_UAU_EXT.2	Extended: Password-based authentication mechanism
FIA_UIA_EXT.1	Extended: User identification and authentication
FPT_APW_EXT.1	Extended: Protection of administrator passwords
FPT_SKP_EXT.1	Extended: Management of TSF data (for reading of symmetric keys)
FPT_TST_EXT.1	Extended: TSF testing
FPT_TUD_EXT.1	Extended: Trusted update
FTA_SSL_EXT.1	Extended: TSF-initiated session locking

5.1.1 Class FAU: Security Audit

Families in this class address the requirements for functions to implement security audit as defined in CC Part 2.

5.1.1.1 Family FAU_STG_EXT: Extended: Security Audit Event Storage

Family Behaviour

This extended family FAU_STG_EXT is modeled after the FAU_STG family. This family defines the requirements for the TSF to be able to create and maintain a secure audit trail. Stored audit records refers to those records within the audit trail, and not the audit records that have been retrieved (to temporary storage) through selection. The requirements of the extended family are focused on the secure transmission of audit records to a remote logging server.

Components in this family address the requirements for protection audit data as defined in CC Part 2. This section defines the extended components for the FAU_STG_EXT family.

Component Leveling

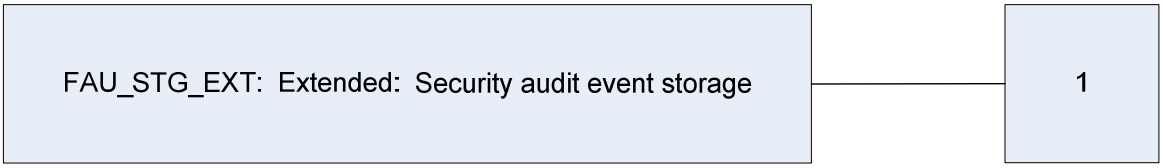

Figure 4 Extended: Security audit event storage family decomposition

FAU_STG_EXT.1 Extended: External Audit Trail Storage is the only component of this family. This component requires the TSF to use an external IT entity for audit data storage. It was modeled after FAU_STG.1.

Management: FAU_STG_EXT.1

a) There are no management activities foreseen.

Audit: FAU_STG_EXT.1

a) There are no audit activities foreseen.

FAU_STG_EXT.1 Extended: External audit trail storage
Hierarchical to: No other components.
Dependencies: FAU_GEN.1 Audit data generation
FTP_ITC.1 Inter-TSF trusted channel

FAU_STG_EXT.1.1

The TSF shall be able to [selection: transmit the generated audit data to an external IT entity, receive and store audit data from an external IT entity] using a trusted channel implementing the [selection: IPSec[21], SSH, TLS, TLS/HTTPS] protocol.

[21] IPSec – Internet Protocol Security

5.1.2 Class FCS: Cryptographic Support

Families in this class address the requirements for functions to implement cryptographic functionality as defined in CC Part 2.

5.1.2.1 Family FCS_CKM_EXT: Extended: Cryptographic Key Management

Family Behaviour

Cryptographic keys must be managed throughout their life cycle. The FCS_CKM family, after which this extended family is modeled, is intended to support that lifecycle and consequently defines requirements for the following activities: cryptographic key generation, cryptographic key distribution, cryptographic key access and cryptographic key destruction. This family should be included whenever there are functional requirements for the management of cryptographic keys. The extended family is designed to include CSPs[22] and further defines the requirements for plaintext secret and private cryptographic keys. The requirements also further define the key destruction methods allowed, per FIPS 140-2 requirements.

Components in this family address the requirements for managing cryptographic keys as defined in CC Part 2. This section defines the extended components for the FCS_CKM_EXT family.

Component Leveling

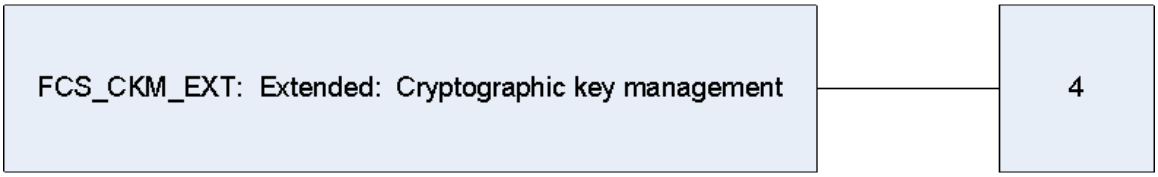

Figure 5 Extended: Cryptographic key management family decomposition

FCS_CKM_EXT.4 Extended: Cryptographic key zeroization is the only component of this family. This component requires cryptographic keys and cryptographic critical security parameters to be zeroized. It was modeled after FCS_CKM.4.

Management: FCS_CKM_EXT.4

a) There are no management activities foreseen.

Audit: FCS_CKM_EXT.4

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

a) There are no auditable events foreseen.

FCS_CKM_EXT.4 Extended: Cryptographic key zeroization
Hierarchical to: FCS_CKM.4.
Dependencies: [FDP_ITC.1 Import of user data without security attributes, or FDP_ITC.2 Import of user data with security attributes, or FCS_CKM.1 Cryptographic key generation]

FCS_CKM_EXT.4.1

The TSF shall zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required.

[22] Critical Security Parameters

5.1.2.3 Family FCS_HTTPS_EXT: Extended: HTTPS

Family Behaviour

Components in this family define the requirements for protecting remote management sessions between the TOE and an authorised administrator. This family describes how HTTPS will be implemented. This is a new family defined for the FCS Class.

Component Leveling

Figure 6 Extended: HTTPS family decomposition

FCS_HTTPS_EXT.1 Extended: HTTPS is the only component of this family. This component requires that HTTPS be implemented according to RFC[23] 2818 and supports TLS.

Management: FCS_HTTPS_EXT.1

a) There are no management activities foreseen.

Audit: FCS_ HTTPS _EXT.1

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

a) Failure of HTTPS session establishment.
b) HTTPS session establishment
c) HTTPS session termination

FCS_HTTPS_EXT.1 Extended: HTTPS
Hierarchical to: No other components.
Dependencies: FCS_TLS_EXT.1 Extended: TLS
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 the HTTPS protocol using TLS as specified in FCS_TLS_EXT.1.

[23] Request for Comment

5.1.2.4 Family FCS_RBG_EXT: Extended: Random Bit Generation

Family Behaviour

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

Component Leveling

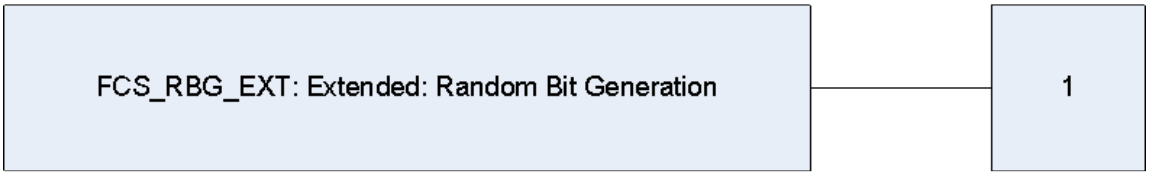

Figure 7 Extended: Random Bit Generation family decomposition

FCS_RBG_EXT.1 Extended: Random Bit Generation is the only component of this class. This component requires random bit generation to be performed in accordance with selected standards and seeded by an entropy source. It was modeled after FCS_COP.1 Cryptographic operation.

Management: FCS_RBG_EXT.1

a) There are no management activities foreseen.

Audit: FCS_RBG_EXT.1

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

a) There are no auditable events foreseen.

FCS_RBG_EXT.1 Extended: Cryptographic operation (Random bit generation)
Hierarchical to: No other components.
Dependencies: No dependencies.
FCS_RBG_EXT.1.1

The TSF shall perform all random bit generation (RBG) services in accordance with [selection, choose one of: NIST[24] Special Publication 800-90 using [selection: Hash DRBG[25] (any), HMAC[26] DRBG (any), CTR[27] DRBG (AES), Dual EC[28] DRBG (any)]; FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using AES] seeded by an entropy source that accumulated entropy from [selection, one or both of: a software-based noise source; a TSF-hardware-based noise source].

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded with a minimum of [selection, choose one of: 128 bits, 256 bits] of entropy at least equal to the greatest bit length of the keys and authorization factors that it will generate.

[24] NIST – National Institute of Standards and Technology
[25] DRBG – Deterministic Random Bit Generator
[26] HMAC – Hashed Message Authentication Code
[27] CTR – Counter Mode
[28] EC – Elliptical Curve

5.1.2.5 Family FCS_SSH_EXT: Extended: SSH

Family Behaviour

Components in this family address the requirements for protecting communications using SSH. This is a new family defined for the FCS Class.

Component Leveling

Figure 8 Extended: SSH family decomposition

FCS_SSH_EXT.1 Extended: SSH is the only component of this family. This component requires that SSH be implemented as specified.

Management: FCS_SSH_EXT.1

a) There are no management activities foreseen.

Audit: FCS_ SSH _EXT.1

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

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

FCS_SSH_EXT.1 Extended: SSH
Hierarchical to: No other components.
Dependencies: FCS_COP.1(1) Cryptographic operation (for data encryption/decryption).

FCS_SSH_EXT.1.1
The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, and 4254.

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

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

FCS_SSH_EXT.1.4
The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms: AES-CBC-128, AES-CBC-256, [selection: AEAD_AES_128_GCM, AEAD_AES_256_GCM, no other algorithms].

FCS_SSH_EXT.1.5
The TSF shall ensure that the SSH transport implementation uses SSH_RSA and [selection: PGP-SIGN-RSA, PGP-SIGN-DSS, no other public key algorithms] as its public key algorithm(s).

FCS_SSH_EXT.1.6
The TSF shall ensure that data integrity algorithms used in SSH transport connection is [selection: hmac-sha1, hmac-sha1-96, hmac-md5, hmac-md5-96].

FCS_SSH_EXT.1.7

The TSF shall ensure that diffie-hellman-group14-sha1 is the only allowed key exchange method used for the SSH protocol.

5.1.2.6 Family FCS_TLS_EXT: Extended: TLS

Family Behaviour

Components in this family address the requirements for protecting communications using TLS. This is a new family defined for the FCS Class.

Component Leveling

Figure 9 Extended: TLS family decomposition

FCS_TLS_EXT.1 Extended: TLS is the only component of this family. This component requires that TLS be implemented as specified.

Management: FCS_TLS_EXT.1

a) There are no management activities foreseen.

Audit: FCS_ TLS _EXT.1

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

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

FCS_TLS_EXT.1 Extended: TLS

Hierarchical to: No other components.

Dependencies: FCS_COP.1(1) Cryptographic operation (for data encryption/decryption)
FCS_COP.1(2) Cryptographic operation (for cryptographic signatures)
FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)

FCS_TLS_EXT.1.1

The TSF shall implement one or more of the following protocols [selection: TLS 1.0 (RFC 2346), TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246)] supporting the following ciphersuites:

Mandatory Ciphersuites:
TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA
Optional Ciphersuites:
[selection:
None
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_SHA256

TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
].

5.1.3 Class FIA: Identification and Authentication

Families in this class address the requirements for functions to establish and verify a claimed user identity as defined in CC Part 2.

5.1.3.1 Family FIA_PMG_EXT: Extended: Password Management

Family Behaviour

This family defines the password strength rules enforced by the TSF.

This section defines the extended components for the FIA_PMG_EXT family, which is modeled after FIA_SOS Specification of secrets.

Component Leveling

Figure 10 Extended: Password Management family decomposition

FIA_PMG_EXT.1 Extended: Password Management is the only component of this family. This component defines the password strength requirements that the TSF will enforce.

Management: FIA_PMG_EXT.1

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

a) Administrator configuration of strength requirements.

Audit: FIA_PMG_EXT.1

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

a) There are no auditable events foreseen.

FIA_PMG_EXT.1 Extended: Password Management
Hierarchical to: No other components.
Dependencies: No dependencies.
FIA_PMG_EXT.1.1

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

1. Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [selection: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [assignment: *other characters*]];
2. Minimum password length shall settable by the Security Administrator, and support passwords of 15 characters or greater.

5.1.3.2 Family FIA_UAU_EXT: Extended: User Authentication

Family Behaviour

This family defines the types of user authentication mechanisms supported by the TSF.

This section defines the extended components for the FIA_UAU_EXT family, which is modeled after the FIA_UAU User authentication family.

Component Leveling

Figure 11 Extended: User authentication family decomposition

FIA_UAU_EXT.2 Extended: Password-based authentication mechanism is the only component of this family. This component requires a local password-based authentication mechanism. In addition, other authentication mechanisms can be specified.

Management: FIA_UAU_EXT.2

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

a) Reset a user password by an administrator.

Audit: FIA_UAU_EXT.2

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

a) All use of the authentication mechanisms.

FIA_UAU_EXT.2 Extended: Password-based authentication mechanism
Hierarchical to: No other components.
Dependencies: No dependencies.
FIA_UAU_EXT.2.1

The TSF shall provide a local password-based authentication mechanism, [selection: [assignment: *other authentication mechanism(s)*], none] to perform administrative user authentication.

5.1.3.3 Family FIA_UIA_EXT: Extended: User Identification and Authentication

Family Behaviour

This family defines the types of user identification and authentication mechanisms supported by the TSF.

This section defines the components for the extended FIA_UIA_EXT family, which is modeled after the FIA_UAU and FIA_UID families.

Component Leveling

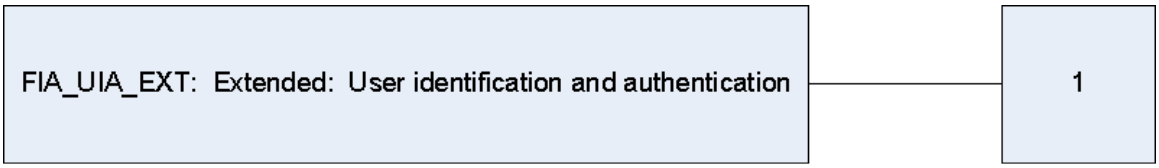

Figure 12 Extended: User identification and authentication family decomposition

FIA_UIA_EXT.1 Extended: User identification and authentication is the only component of this class, and is modeled after a combination of FIA_UAU.1 and FIA_UID.1. This component defines the actions available to users prior to initiating the identification and authentication process, and requires administrative users to be successfully identified and authenticated prior to interacting with the TSF.

Management: FIA_UIA_EXT.1

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

a) Management of the authentication data by an administrator;
b) Management of the authentication data by the associated user;
c) Managing the list of actions that can be taken before the user is identified and authenticated;
d) Management of the user identities;

Audit: FIA_UIA_EXT.1

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

a) All use of the identification and authentication mechanism.

FIA_UIA_EXT.1 Extended: User identification and authentication
Hierarchical to: FIA_UID.1 Timing of identification
FIA_UAU.1 Timing of authentication
Dependencies: FTA_TAB.1 Default TOE access banners

FIA_UIA_EXT.1.1

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

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

FIA_UIA_EXT.1.2

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

5.1.4 Class FPT: Protection of the TSF

Families in this class address the requirements for functions providing integrity and management of mechanisms that constitute the TSF and of the TSF data as defined in CC Part 2.

5.1.4.1 Family FPT_APW_EXT: Extended: Protection of Administrator Passwords

Family Behaviour

Components in this family address the requirements for managing and protecting TSF data, such as passwords. This is a new family modeled after the FPT_PTD family.

Component Leveling

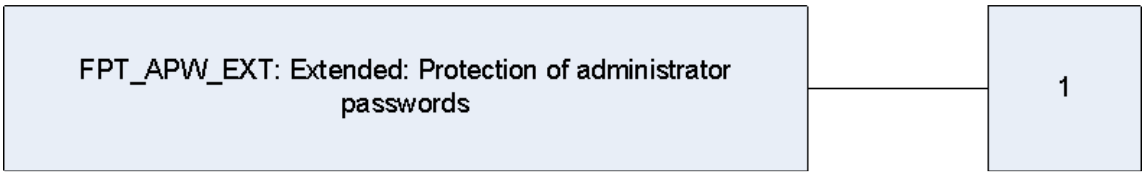

Figure 13 Extended: Protection of administrator passwords family decomposition

FPT_APW_EXT.1 Extended: Protection of administrator passwords, requires preventing selected TSF data from being read by any user or subject. It is the only component of this family.

Management: FPT_APW_EXT.1

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

a) There are no management activities foreseen.

Audit: FPT_APW_EXT.1

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

a) There are no auditable events foreseen.

FPT_APW_EXT.1 Extended: Protection of administrator passwords
Hierarchical to: No other components
Dependencies: No dependencies.
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.

5.1.4.2 Family FPT_SKP_EXT: Extended: Protection of TSF Data

Family Behaviour

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

Component Leveling

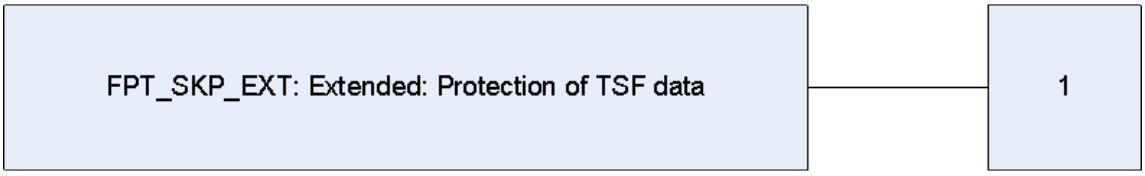

Figure 14 Extended: Protection of TSF data family decomposition

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

Management: FPT_SKP_EXT.1

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

b) There are no management activities foreseen.

Audit: FPT_SKP_EXT.1

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

b) There are no auditable events foreseen.

FPT_SKP_EXT.1 Extended: Protection of TSF data (for reading of all symmetric keys)
Hierarchical to: No other components
Dependencies: No dependencies.
FPT_SKP_EXT.1.1
The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

5.1.4.3 Family FPT_TST_EXT: Extended: TSF Self Test

Family Behaviour

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

The extended FPT_TST_EXT family is modeled after the FPT_TST family.

Component Leveling

Figure 15 Extended: TSF testing family decomposition

FPT_TST _EXT.1 Extended: TSF testing is the only component of this family. This component requires a suite of self tests to be run during initial start-up in order to demonstrate correct operation of the TSF.

Management: FPT_TST _EXT.1

a) There are no management activities foreseen.

Audit: FPT_TST _EXT.1

a) There are no auditable activities foreseen.

FPT_TST_EXT.1 Extended: TSF testing
Hierarchical to: No other components.
Dependencies: No dependencies.
FPT_TST_EXT.1.1
The TSF shall run a suite of self tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

5.1.4.4 Family FPT_TUD_EXT: Extended: Trusted Update

Family Behaviour

Components in this family address the requirements for updating the TOE firmware and/or software. This is a new family defined for the FPT Class.

Component Leveling

Figure 16 Extended: Trusted update family decomposition

FPT_TUD_EXT.1 Extended: Trusted update, requires management tools be provided to update the TOE firmware and software, including the ability to verify the updates prior to installation. It is the only component of this family.

Management: FPT_ TUD_EXT.1

a) There are no management activities foreseen.

Audit: FPT_ TUD_EXT.1

a) Initiation of the update process.

FPT_TUD_EXT.1 Extended: Trusted update
Hierarchical to: No other components.
Dependencies: [FCS_COP.1(2) Cryptographic operation (for cryptographic signature), or FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)]

FPT_TUD_EXT.1.1
The TSF shall provide security administrators the ability to query the current version of the TOE firmware/software.

FPT_TUD_EXT.1.2
The TSF shall provide security administrators the ability to initiate updates to TOE firmware/software.

FPT_TUD_EXT.1.3
The TSF shall provide a means to verify firmware/software updates to the TOE using a [selection: digital signature mechanism, published hash] prior to installing those updates.

5.1.5 Class FTA: TOE Access

Families in this class specify functional requirements for controlling the establishment of a user's session as defined in CC Part 2.

5.1.5.1 Family FTA_SSL_EXT: Extended: TSF-initiated Session Locking

Family Behaviour

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

The extended FTA_SSL_EXT family is based on the FTA_SSL family.

Component Leveling

Figure 17 Extended: TSF-initiated session locking family decomposition

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

Management: FTA_SSL_EXT.1

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

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

Audit: FTA_SSL_EXT.1

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

a) Any attempts at unlocking an interactive session.

FTA_SSL_EXT.1 Extended: TSF-initiated session locking
Hierarchical to: No other components.
Dependencies: FIA_UAU.1 Timing of authentication.
FTA_SSL_EXT.1.1
The TSF shall, for local interactive sessions, [selection:

- lock the session - disable any activity of the user's data access/display devices other than unlocking the session, and requiring that the administrator re-authenticate to the TSF prior to unlocking the session;
- terminate the session].

after a Security Administrator-specified time period of inactivity.

5.2 Extended TOE Security Assurance Components

There are no extended TOE Security Assurance Components.

6 Security Requirements

This section defines the SFRs and SARs met by the TOE. These requirements are presented following the conventions identified in Section 6.1.

6.1 Conventions

There are several font variations used within this ST. Selected presentation choices are discussed here to aid the Security Target reader.

The CC allows for assignment, refinement, selection and iteration operations to be performed on security functional requirements. All of these operations are used within this ST. These operations are performed as described in Part 2 of the CC, and are shown as follows:

- Completed assignment statements made by the ST author are identified using [*italicized text within brackets*].
- Completed selection statements are identified using [underlined text within brackets]. In keeping with these conventions, in the event an assignment is within a selection, it will be depicted as *italicized, underlined* text.
- Refinements are identified using **bold text**. Any text removed is stricken (Example: ~~TSF Data~~) and should be considered as a refinement. In keeping with these conventions, in the event a refinement is within an assignment, it will be depicted as ***bold italicized*** text, and when a refinement is within a selection, it will be depicted in **bold underlined** text.
- Extended Functional and Assurance Requirements are identified using "_EXT" at the end of the short name.
- Iterations are identified by appending a number in parentheses following the component title. For example, FAU_GEN.1(1) Audit Data Generation would be the first iteration and FAU_GEN.1(2) Audit Data Generation would be the second iteration.
- Operations such as assignments and selections performed by the PP author are identified as shown above; however, do not appear within brackets. This is done intentionally to delineate between selections/assignments made by the PP author and those made by the ST author. No refinements have been made by the ST author other than grammatical and/or formatting corrections, or in places where a table reference differs from that of the PP.

6.2 Security Functional Requirements

This section specifies the SFRs for the TOE. This section organizes the SFRs by CC class. Table 11 identifies all SFRs implemented by the TOE and indicates the ST operations performed on each requirement.

Table 11 TOE Security Functional Requirements

Name	Description	S	A	R	I
FAU_GEN.1	Audit data generation	✓	✓	✓	
FAU_GEN.2	User identity association				
FAU_STG_EXT.1	Extended: External audit trail storage	✓			
FCS_CKM.1	Cryptographic key generation	✓	✓	✓	
FCS_CKM_EXT.4	Extended: Cryptographic key destruction				
FCS_COP.1(1)	Cryptographic operation (for data encryption/decryption)	✓	✓	✓	✓

Name	Description	S	A	R	I
FCS_COP.1(2)	Cryptographic operation (for cryptographic signature)	✓	✓	✓	✓
FCS_COP.1(3)	Cryptographic operation (for cryptographic hashing)	✓	✓	✓	✓
FCS_COP.1(4)	Cryptographic operation (for keyed-hash message authentication)	✓	✓	✓	✓
FCS_HTTPS_EXT.1	Extended: HTTPS				
FCS_RBG_EXT.1	Extended: Cryptographic operation (Random bit generation)	✓			
FCS_SSH_EXT.1	Extended: SSH	✓	✓		
FCS_TLS_EXT.1	Extended: TLS	✓			
FDP_RIP.2	Full residual information protection	✓			
FIA_PMG_EXT.1	Extended: Password management	✓	✓		
FIA_UAU_EXT.2	Extended: Password-based authentication mechanism	✓	✓		
FIA_UAU.7	Protected authentication feedback		✓		
FIA_UIA_EXT.1	Extended: User identification and authentication	✓	✓		
FMT_MTD.1	Management of TSF data (for general TSF data)	✓	✓	✓	
FMT_SMF.1	Specification of management functions	✓	✓		
FMT_SMR.2	Restrictions on security roles		✓		
FPT_APW_EXT.1	Extended: Protection of administrator passwords				
FPT_SKP_EXT.1	Extended: Management of TSF data (for reading of all symmetric keys)				
FPT_STM.1	Reliable time stamps			✓	
FPT_TST_EXT.1	Extended: TSF testing				
FPT_TUD_EXT.1	Extended: Trusted update	✓			
FTA_SSL_EXT.1	Extended: TSF-initiated session locking	✓			
FTA_SSL.3	TSF-initiated termination		✓	✓	
FTA_SSL.4	User-initiated termination			✓	
FTA_TAB.1	Default TOE access banners			✓	
FTP_ITC.1	Inter-TSF trusted channel	✓	✓	✓	
FTP_TRP.1	Trusted path	✓	✓	✓	

Note: S=Selection; A=Assignment; R=Refinement; I=Iteration

6.2.1 Class FAU: Security Audit

FAU_GEN.1 Audit data generation
Hierarchical to: No other components.
Dependencies: FPT_STM.1 Reliable time stamps
FAU_GEN.1.1

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

a) Start-up of the audit functions;
b) All auditable events, for the not specified level of audit; and
c) *All administrative actions;*
d) *Specifically defined auditable events listed in* ***Table 12****.*

Table 12 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.
FCS_CKM.1	None.	None.
FCS_CKM_EXT.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. Establishment/Termination of a HTTPS session.	Reason for failure. Non-TOE endpoint of connection (IP[29] address) for both successes and failures.
FCS_TLS_EXT.1	Failure to establish a TLS Session. Establishment/Termination of a TLS session.	Reason for failure. Non-TOE endpoint of connection (IP address) for both successes and failures.
FCS_SSH_EXT.1	Failure to establish an SSH Session. Establishment/Termination of an SSH session.	Reason for failure. Non-TOE endpoint of connection (IP address) for both successes and failures.
FDP_RIP.2	None.	None.
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of the identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).

[29] IP – Internet Protocol

Requirement	Auditable Events	Additional Audit Record Contents
FIA_UAU_EXT.2	All use of the authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.
FMT_MTD.1	None.	None.
FMT_SMF.1	None.	None.
FMT_SMR.2	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_STM.1	Changes to the time.	The old and new values for the time. Origin of the attempt (e.g., IP address).
FPT_TUD_EXT.1	Initiation of update.	No additional information.
FPT_TST_EXT.1	None.	None.
FTA_SSL_EXT.1	Any attempts at unlocking of an interactive session.	No additional information.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	No additional information.
FTA_SSL.4	The termination of an active session.	No additional information.
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 attempts.
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.

FAU_GEN.1.2

The TSF shall record within each audit record at least the following information:

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

b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, *information specified in column three of* ***Table 12***.

FAU_GEN.2 User identity association

Hierarchical to: No other components.

Dependencies: FAU_GEN.1 Audit data generation
FIA_UID.1 Timing of identification

FAU_GEN.2.1

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

FAU_STG_EXT.1 Extended: External audit trail storage
Hierarchical to: No other components.
Dependencies: FAU_GEN.1 Audit data generation
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 implementing the [TLS] protocol.

6.2.2 Class FCS: Cryptographic Support

FCS_CKM.1 Cryptographic key generation
Hierarchical to: No other components.
Dependencies: FCS_COP.1 Cryptographic operation
FCS_CKM.4 Cryptographic key destruction

FCS_CKM.1.1

The TSF shall generate **asymmetric** cryptographic keys **used for key establishment** in accordance with ~~a specified cryptographic key generation algorithm~~ [***NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for elliptic curve-based key establishment schemes and implementing "NIST curves" P-256, P-384, and [P-521] (as defined in FIPS PUB[30] 186-3, "Digital Signature Standard")***] and specified cryptographic key sizes *equivalent to, or greater than, a symmetric key strength of 112 bits* ~~that meet the following: [assignment: *list of standards*]~~.

FCS_CKM_EXT.4 Extended: Cryptographic key destruction
Hierarchical to: No other components.
Dependencies: FCS_CKM.1 Cryptographic key generation

FCS_CKM_EXT.4.1

The TSF shall zeroize all plaintext secret and private cryptographic keys and CSP[31]s when no longer required.

FCS_COP.1(1) Cryptographic operation (for data encryption/decryption)
Hierarchical to: No other components.
Dependencies: FCS_CKM.1 Cryptographic key generation
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1(1).1

The TSF shall perform *encryption and decryption* in accordance with a specified cryptographic algorithm *AES operating in* [***CBC[32], ECB[33], GCM[34], and CMAC[35] modes***] and cryptographic key sizes *128-bits, 256-bits, and* [***192-bits***] that meet the following: [

- ***FIPS PUB 197, "Advanced Encryption Standard (AES)"***
- [***NIST SP[36] 800-38A, NIST SP 800-38B, NIST SP 800-38D***]].

[30] PUB – Publication
[31] CSP – Critical Security Parameter
[32] CBC – Cipher Block Chaining
[33] ECB – Electronic Codebook
[34] GCM – Galois/Counter Mode
[35] CMAC – Cipher-based Message Authentication Code
[36] SP – Special Publication

FCS_COP.1(2) Cryptographic operation (for cryptographic signature)
Hierarchical to: No other components.
Dependencies: FCS_CKM.1 Cryptographic key generation
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1(2).1

The TSF shall perform **cryptographic signature services** in accordance with a ~~specified cryptographic algorithm~~ **[**

- ***Digital Signature Algorithm (DSA) with a key size (modulus) of 2048 bits or greater,***
- ***RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater, or***
- ***Elliptic Curve Digital Signature Algorithm (ECDSA) with a key size of 256 bits or greater*****]**

~~and cryptographic key sizes [assignment: *cryptographic key* sizes]~~ that meets the following: [

Case: Digital Signature Algorithm

- ***FIPS PUB 186-3, "Digital Signature Standard"***

Case: RSA Digital Signature Algorithm

- ***FIPS PUB 186-2 or FIPS PUB 186-3, "Digital Signature Standard"***

Case: Elliptic Curve Digital Signature Algorithm

- ***FIPS PUB 186-3, "Digital Signature Standard"***
- ***The TSF shall implement "NIST curves" P-256, P-384, and [P-521] (as defined in FIPS PUB 186-3, "Digital Signature Standard").***]

FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)
Hierarchical to: No other components.
Dependencies: FCS_CKM.1 Cryptographic key generation
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1(3).1

The TSF shall perform *cryptographic hashing services* in accordance with a specified cryptographic algorithm [***SHA[37]-1, SHA-224, SHA-256, SHA-384, SHA-512***] **and** ~~cryptographic key~~ **message digest sizes** [***160, 224, 256, 384, 512***] **bits** that meet the following: *FIPS PUB 180-3, "Secure Hash Standard"*

FCS_COP.1(4) Cryptographic operation (for keyed-hash message authentication)
Hierarchical to: No other components.
Dependencies: FCS_CKM.1 Cryptographic key generation
FCS_CKM.4 Cryptographic key destruction

FCS_COP.1(4).1

The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm *HMAC*-**[*SHA-1, SHA-224, SHA-256, SHA-384, SHA-512*]**, ~~and cryptographic~~ key size **[*160, 224, 256, 384, 512*], and message digest sizes** [**160, 224, 256, 384, 512**] **bits** that meet the following: *FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-3, "Secure Hash Standard"*.

FCS_HTTPS_EXT.1 Extended: HTTPS
Hierarchical to: No other components.
Dependencies: FCS_TLS_EXT.1

FCS_HTTPS_EXT.1.1

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

FCS_HTTPS_EXT.1.2

The TSF shall implement HTTPS using TLS as specified in FCS_TLS_EXT.1.

FCS_RBG_EXT.1 Extended: Cryptographic operation (Random bit generation)
Hierarchical to: No other components.

[37] SHA – Secure Hash Algorithm

Dependencies: No dependencies.

FCS_RBG_EXT.1.1

The TSF shall perform all random bit generation (RBG) services in accordance with [NIST Special Publication 800-90 using [CTR_DRBG (AES)]] seeded by an entropy source that accumulated entropy from [a TSF-hardware-based noise source].

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded with a minimum of [256 bits] of entropy at least equal to the greatest bit length of the keys and authorization factors that it will generate.

FCS_SSH_EXT.1 Extended: SSH

Hierarchical to: No other components.

Dependencies: FCS_COP.1(1) Cryptographic operation (for data encryption/decryption).

FCS_SSH_EXT.1.1

The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, and 4254.

FCS_SSH_EXT.1.2

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

FCS_SSH_EXT.1.3

The TSF shall ensure that, as described in RFC 4253, packets greater than [*131,072*] bytes in an SSH transport connection are dropped.

FCS_SSH_EXT.1.4

The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms: AES-CBC-128, AES-CBC-256, [no other algorithms].

FCS_SSH_EXT.1.5

The TSF shall ensure that the SSH transport implementation uses SSH_RSA and [no other public key algorithms] as its public key algorithm(s).

FCS_SSH_EXT.1.6

The TSF shall ensure that data integrity algorithms used in SSH transport connection is [hmac-sha1].

FCS_SSH_EXT.1.7

The TSF shall ensure that diffie-hellman-group14-sha1 is the only allowed key exchange method used for the SSH protocol.

FCS_TLS_EXT.1 Extended: TLS

Hierarchical to: No other components.

Dependencies: FCS_COP.1(1) Cryptographic operation (for data encryption/decryption)
FCS_COP.1(2) Cryptographic operation (for cryptographic signatures)
FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)

FCS_TLS_EXT.1.1

The TSF shall implement one or more of the following protocols [TLS 1.0 (RFC 2346), TLS 1.1 (RFC 4346), TLS 1.2 (RFC 5246)] supporting the following ciphersuites:

Mandatory Ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_DHE_RSA_WITH_AES_128_CBC_SHA
TLS_DHE_RSA_WITH_AES_256_CBC_SHA

Optional Ciphersuites:

None.

6.2.3 Class FDP: User Data Protection

FDP_RIP.2 Full Residual Information Protection

Hierarchical to: No other components.

Dependencies: No dependencies.

FDP_RIP.2.1

The TSF shall ensure that any previous information content of a resource is made unavailable upon the [*deallocation of the resource from*] all objects.

6.2.4 Class FIA: Identification and Authentication

FIA_PMG_EXT.1 Extended: Password management
Hierarchical to: No other components.
Dependencies: No dependencies.
FIA_PMG_EXT.1.1

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

1. Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [*any character from the Unicode set*[38]];
2. Minimum password length shall settable by the Security Administrator, and support passwords of 15 characters or greater.

FIA_UAU.7 Protected authentication feedback
Hierarchical to: No other components.
Dependencies: FIA_UAU.1
FIA_UAU.7.1

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

FIA_UAU_EXT.2 Extended: Password-based authentication mechanism
Hierarchical to: No other components.
Dependencies: No dependencies.
FIA_UAU_EXT.2.1

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

FIA_UIA_EXT.1 User identification and authentication
Hierarchical to: No other components.
Dependencies: FTA_TAB.1 Default TOE access banners
FIA_UIA_EXT.1.1

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

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

FIA_UIA_EXT.1.2

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

[38] Space and tab can be entered as the first or last characters of the password, but all white space at the beginning and end of passwords is truncated by the TOE.

6.2.5 Class FMT: Security Management

FMT_MTD.1 Management of TSF data
Hierarchical to: No other components.
Dependencies: FMT_SMF.1 Specification of management functions
FMT_SMR.1 Security roles
FMT_MTD.1.1
The TSF shall restrict the ability to *manage* the *TSF data* to the *Security Administrators*.

FMT_SMF.1 Specification of management functions
Hierarchical to: No other components.
Dependencies: No dependencies.
FMT_SMF.1.1
The TSF shall be capable of performing the following management functions:
- *Ability to administer the TOE locally and remotely;*
- *Ability to update the TOE, and to verify the updates using [digital signature, published hash] capability prior to installing those updates;*
- *[Ability to configure the cryptographic functionality]*

FMT_SMR.2 Restrictions on security roles
Hierarchical to: No other components.
Dependencies: FIA_UID.1 Timing of identification
FMT_SMR.2.1
The TSF shall maintain the roles:
- *Authorized 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
- *Authorized Administrator role shall be able to administer the TOE locally;*
- *Authorized Administrator role shall be able to administer the TOE remotely;*

are satisfied.

6.2.6 Class FPT: Protection of the TSF

FPT_APW_EXT.1 Protection of administrator passwords
Hierarchical to: No other components.
Dependencies: No dependencies.
FPT_APW_EXT.1.1
The TSF shall store passwords in non-plaintext form.
FPT_APW_EXT.1.2
The TSF shall prevent reading of the plaintext passwords.

FPT_SKP_EXT.1 Extended: Management of TSF data (for reading of all symmetric keys)
Hierarchical to: No other components.
Dependencies: No dependencies.
FPT_SKP_EXT.1.1
The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

FPT_STM.1 Reliable time stamps
Hierarchical to: No other components.
Dependencies: No dependencies.
FPT_STM.1.1
The TSF shall be able to provide reliable time stamps **for its own use**.

FPT_TST_EXT.1 Extended: TSF testing
Hierarchical to: No other components.
Dependencies: No dependencies.
FPT_TST_EXT.1.1
The TSF shall run a suite of self tests during initial start-up (on power on) to demonstrate the correct operation of the TSF.

FPT_TUD_EXT.1 Extended: Trusted update
Hierarchical to: No other components.
Dependencies: FCS_COP.1(2) Cryptographic operation (for cryptographic signature),
FCS_COP.1(3) Cryptographic operation (for cryptographic hashing)
FPT_TUD_EXT.1.1
The TSF shall provide security administrators the ability to query the current version of the TOE firmware/software.
FPT_TUD_EXT.1.2
The TSF shall provide security administrators the ability to initiate updates to TOE firmware/software.
FPT_TUD_EXT.1.3
The TSF shall provide a means to verify firmware/software updates to the TOE using a [digital signature mechanism, published hash] prior to installing those updates.

6.2.7 Class FTA: TOE Access

FTA_SSL_EXT.1 Extended: TSF-initiated session locking
Hierarchical to: No other components.
Dependencies: No dependencies.
FTA_SSL_EXT.1.1

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

FTA_SSL.3 TSF-initiated termination
Hierarchical to: No other components.
Dependencies: No dependencies.
FTA_SSL.3.1

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

FTA_SSL.4 User-initiated termination
Hierarchical to: No other components.
Dependencies: No dependencies.
FTA_SSL.4.1

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

FTA_TAB.1 Default TOE access banners
Hierarchical to: No other components.
Dependencies: No dependencies.
FTA_TAB.1.1

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

6.2.8 Class FTP: Trusted Path/Channels

FTP_ITC.1 Inter-TSF trusted channel

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_ITC.1.1

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

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 [*syslog*].

FTP_TRP.1 Trusted path

Hierarchical to: No other components.

Dependencies: No dependencies.

FTP_TRP.1.1

The TSF shall **use [SSH, TLS/HTTPS] to** provide a **trusted** communication path between itself and **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 detection of modification of the communicated 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 administrative actions*.

6.3 Security Assurance Requirements

This section defines the assurance requirements for the TOE. Assurance requirements are taken from the CC Part 3 and are NDPP conformant.

Table 13 below summarizes the requirements.

Table 13 NDPP Assurance Requirements

Assurance Requirements	
Class ASE: Security Target evaluation	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.1 Security objectives for the operational environment
	ASE_REQ.1 Stated security requirements
	ASE_TSS.1 TOE summary specification
Class ALC : Life Cycle Support	ALC_CMC.1 Labeling of the TOE
	ALC_CMS.1 TOE CM Coverage
Class ADV: Development	ADV_FSP.1 Basic functional specification
Class AGD: Guidance documents	AGD_OPE.1 Operational user guidance
	AGD_PRE.1 Preparative procedures
Class ATE: Tests	ATE_IND.1 Independent testing – conformance
Class AVA: Vulnerability assessment	AVA_VAN.1 Vulnerability survey

7 TOE Summary Specification

This section presents information to detail how the TOE meets the functional requirements described in previous sections of this ST.

7.1 TOE Security Functions

Each of the security requirements and the associated descriptions correspond to the security functions. Hence, each function is described by how it specifically satisfies each of its related requirements. This serves to both describe the security functions and rationalize that the security functions satisfy the necessary requirements.

Table 14 Mapping of TOE Security Functions to Security Functional Requirements

TOE Security Functionality	SFR ID	Description
Security Audit	FAU_GEN.1	Audit data generation
	FAU_GEN.2	User identity association
	FAU_STG_EXT.1	Extended: External audit trail storage
Cryptographic Support	FCS_CKM.1	Cryptographic key generation
	FCS_CKM_EXT.4	Extended: Cryptographic key destruction
	FCS_COP.1(1)	Cryptographic operation (for data encryption/decryption)
	FCS_COP.1(2)	Cryptographic operation (for cryptographic signature)
	FCS_COP.1(3)	Cryptographic operation (for cryptographic hashing)
	FCS_COP.1(4)	Cryptographic operation (for keyed-hash message authentication)
	FCS_HTTPS_EXT.1	Extended: HTTPS
	FCS_RBG_EXT.1	Extended: Cryptographic operation (Random bit generation)
	FCS_SSH_EXT.1	Extended: SSH
	FCS_TLS_EXT.1	Extended: TLS
User Data Protection	FDP_RIP.2	Full residual information protection
Identification and Authentication	FIA_PMG_EXT.1	Extended: Password management
	FIA_UAU_EXT.2	Extended: Password-based authentication mechanism
	FIA_UAU.7	Protected authentication feedback

TOE Security Functionality	SFR ID	Description
	FIA_UIA_EXT.1	Extended: User identification and authentication
Security Management	FMT_MTD.1	Management of TSF data (for general TSF data)
	FMT_SMF.1	Specification of management functions
	FMT_SMR.2	Restrictions on security roles
Protection of the TSF	FPT_APW_EXT.1	Extended: Protection of administrator passwords
	FPT_SKP_EXT.1	Extended: Management of TSF data (for reading of all symmetric keys)
	FPT_STM.1	Reliable time stamps
	FPT_TST_EXT.1	Extended: TSF testing
	FPT_TUD_EXT.1	Extended: Trusted update
TOE Access	FTA_SSL_EXT.1	Extended: TSF-initiated session locking
	FTA_SSL.3	TSF-initiated termination
	FTA_SSL.4	User-initiated termination
	FTA_TAB.1	Default TOE access banners
Trusted path/channels	FTP_ITC.1	Inter-TSF trusted channel
	FTP_TRP.1	Trusted path

7.1.1 Security Audit

The TOE generates audit records within three different logs: the Audit Logs, System Logs, and Access Logs. The Audit Logs record all administrative user activities, such as login, logout, policy additions, etc. The System Logs record changes that occur in the life of and during movement of a document (a document is a web services element such as an XML page) through various processes on the TOE. The Access Logs capture metadata (such as timestamp, session ID, Client IP, URI[39], etc.) of documents being processed by the system. All three logs are stored in log files on the local file system. The resulting audit records can be examined to determine what security-relevant activities took place and who (i.e. which user) is responsible for those activities.

The default size of log files is 1024 Megabytes, but this value can be changed by an authorized administrator. Audit logs are stored as XML that can be compressed via Zip or GNU[40] zip compression. The TOE provides the capability to export log files into plain text or HTML[41] format. Once a log file

[39] Uniform Resource Identifier
[40] GNU – GNU's Not Unix
[41] HTML – Hypertext Markup Language

reaches the set capacity, log messages begin to wrap, replacing the oldest log entries with the newest log entries.

The CLI and GUI provide authorized administrators with access to view the logs. Within the logs, the TOE records the user name of any user who took an action when the action is initiated by a user (as opposed to automated actions performed by the system). In the CLI, the `show log audit`, `show log access`, and `show log system` commands allow administrators to review the logs. In the GUI, administrators can view the logs by navigating to the Diagnostics -> Logging page. All administrators have access to view the logs. The TOE does not display any GUI pages, CLI prompts, or logs to users who are not authenticated administrators.

The TOE allows administrators to configure a secure connection to the Syslog server. The TOE records an audit record for a failed login attempt via SSH, TLS, and HTTPS, including the reason for the failure and the IP address of the host.

TOE Security Functional Requirements Satisfied: FAU_GEN.1, FAU_GEN.2, FAU_STG_EXT.1

7.1.2 Cryptographic Support

Cryptographic operations on the TOE are provided by a FIPS 140-2 validated cryptographic module. The TOE uses TLS[42]/HTTPS and SSH to protect administrator communications during management sessions. SSH encrypts CLI traffic while TLS/HTTPS is used to encrypt traffic via the GUI. The SSH implementation complies with the standards identified in RFCs 4251, 4252, 4253, and 4254. The TOE uses AES to encrypt and decrypt management data. Additionally, the TOE implements the SHA and HMAC-SHA algorithms to support SSH and TLS. The TOE can use AES to protect data traveling between itself and a secure, trusted endpoint across the network as well. The TOE HSM is capable of providing DSA and ECDSA to requesting services.

The TOE detects large SSH packets by examining the header information for incoming packets. If the packet is an SSH packet, and the packet size is greater than 128 kilobytes, then the packet is dropped. SSH traffic can be encrypted with AES-CBC-128 and AES-CBC-256. SSH transport can be enforced by SSH_RSA. For data integrity during SSH sessions, the TOE uses HMAC-SHA1. Diffie-Hellman-group14-SHA1 is the only allowed key exchange method used for the SSH protocol.

The TOE provides public-key authentication capabilities in addition to password-based authentication. Administrative users can upload keys for client devices that are authorized to access the system. Then, every time that client connects to the TOE, the TOE sends a message encrypted with the client's public key to the client. The client uses the client private key to decrypt the message and prove to the server that it has successfully decrypted the message. Then the TOE allows the client to log in.

The TOE can use AES 128 and 256-bit when processing HTTPS/TLS requests, depending on the capabilities of the client. When establishing a session, the client and server use the standard TLS handshake protocol, which involves exchanging the server's certificate and then the client returning an encrypted pre-master secret. The client and server then use the pre-master secret to generate keys known only to the client and server. These keys are used to encrypt all future messages between the client and server. HTTPS/TLS is used for management sessions via the GUI and protecting communications with a remote syslog server. The TOE uses the following ciphersuites:

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA

[42] The TOE supports TLS versions 1.0, 1.1, and 1.2.

The TOE uses a DSA or RSA key in a digital certificate (stored in persistent memory on a hard drive) to perform key exchange with clients connecting via HTTPS and SSH. This key is loaded by default, but can be replaced with another certificate manually input by an administrator via the GUI. All symmetric keys are AES keys (stored in volatile memory in random access memory). No other keys or key-generating CSPs are used by the TOE. Certificate keys are only zeroized when the certificate expires or when the certificate is replaced. AES keys are zeroized after the session they are associated with ends. 3-key Triple-DES keys are zeroized only when the system is reinstalled. Zeroization is performed for all keys in all types of memory by overwriting all key data with zeros one time. Only password keys and SSH and TLS certificate keys are stored persistently, and these are also overwritten with zeros one time when zeroized

The TOE implements a NIST SP 800-56A Section 6.3 conformant Diffie-Hellman-based key agreement scheme for nonce-based key agreement.

Entropy for DRBG input is provided by a hardware-based Random Number Generator (RNG). This hardware device provide sufficient entropy to seed the hardware RNG directly (no entropy pools or buffers are used). This allows the TOE to keep a constant entropy pool of 256-bits that can be used for key generation.

TOE Security Functional Requirements Satisfied: FCS_CKM.1, FCS_CKM_EXT.4, FCS_DRBG_EXT.1, FCS_HTTPS_EXT.1, FCS_TLS_EXT.1, FCS_SSH_EXT.1.

7.1.3 User Data Protection

The TOE clears the memory space used for storing network data (buffers) by overwriting the memory space with zeros after the TOE finishes using that memory space (after each connection closes). The TOE ensures that no residual data remains prior to allocation of memory, ensuring that any attempt to reconstruct the content of the memory buffers after reallocation will result in the reconstruction of the zeros rather than actual packet data.

TOE Security Functional Requirements Satisfied: FDP_RIP.2.

7.1.4 Identification and Authentication

The CLI and GUI on the TOE are used in accessing this function. Administrators can view the login banner prior to authenticating to the TOE. The TOE must perform successful identification and authentication of the TOE administrator before the TSF grants the administrator access to other TOE security functions on the CLI or GUI. Administrator authentication is enforced through the use of a password. Passwords must meet the following criteria:

- composed of upper- and lower-case letters, numbers, and special characters from the Unicode set,
- minimum password length settable by an administrator and can be set to 15 characters or greater,

While authenticating via the local console, the TOE does not provide any visual feedback for the administrator's password.

Authentication via both methods (CLI and GUI) requires the use of a username and password combination. The CLI only accepts credentials via SSH or a serial connection and the GUI only accepts credentials via HTTPS. A login is considered successful if the username and the administrator's encrypted password match the stored username and encrypted password stored on the TOE.

TOE Security Functional Requirements Satisfied: FIA_PMG_EXT.1, FIA_UIA_EXT.1, FIA_UAU_EXT.2, FIA_UAU.7.

7.1.5 Security Management

Security management specifies how the TOE manages several aspects of the TSF including TSF data and security functions. TSF data includes configuration data of the TSF and audit data and cryptographic functionality. The TOE provides authorized administrators with a web GUI and CLI to easily manage the security functions and TSF data of the TOE. The TOE provides commands via the CLI and GUI pages that allow administrators to configure the cryptographic settings of the TOE, such as whether or not FIPS mode is enabled and configurations that affect the operation of the HSM cryptographic hardware accelerator.

The TOE only specifies one administrator role for the CLI, which can then gain "enable" privileges at any time by typing the `enable` command followed by the enable password. In the GUI, administrators can be set to have advanced privileges by checking the "Enable privileged access" option on the User Management -> User Details screen. Regular administrators have mostly read-only access and access to informational utilities such as `ping` and `traceroute`. Privileged administrators or enable users have access to all configuration settings on the TOE, including setting up policies, host settings, manage users, and manage cryptographic settings. Access to the GUI is remote only, but administrators can connect to the CLI remotely or locally.

Unauthenticated administrators only have access to read the displayed warning banner before authenticating successfully with the TOE. While the TOE access banner is displayed to all administrators before authentication, it is read-only and cannot be modified by an unauthenticated administrator (and is not modifiable from the login screen at all).

Administrators can initiate an update to the system firmware by using the SYSTEM > Configuration > Upgrade page of the Web GUI. Updates use a fingerprint for verification, described in more detail in Section 7.1.6 below.

TOE Security Functional Requirements Satisfied: FMT_MTD.1, FMT_SMF.1, FMT_SMR.2.

7.1.6 Protection of the TSF

The TOE does not allow any administrator to read plaintext passwords stored on the TOE, since all passwords are stored in an encrypted format using Triple-DES. The TOE also prevents pre-shared, symmetric, and private keys from being read by storing keys in internally-allocated data structures. This means that key data, which is stored in volatile memory in plaintext and on the hard drive encrypted via Triple-DES, can only be output via the cryptographic API[43], and no user- or administrator-accessible interfaces can be used to read keys. Session keys for SSH or TLS sessions are exchanged via Diffie-Hellman or RSA to ensure that only the intended recipient is able to read the key data. The OS safeguards memory and process space from unauthorized access. The TOE does not offer direct access to memory.

The TOE generates its own time stamps that originate from a system hardware clock. Time stamps are used by the logging function to record an accurate time for each auditable event, and also for licensing. The time can be changed via the `system config time` command in the CLI, which changes the system time for all uses including logs and licensing. Use of an NTP (Network Time Protocol) server is not part of the evaluated configuration.

Administrators can find the current version of TOE firmware by viewing the CLI startup screen, at any time by running the `show general` command from the CLI. Administrators can initiate an update to the system firmware from the SYSTEM > Configuration > Upgrade page of the Web GUI. The TOE uses a fingerprint of each update file to verify authenticity. The fingerprint uses a message digest and a signature. The signature uses SHA-1 with RSA[44], the message digest uses SHA-1, and the fingerprint is a

[43] API – Application Programming Interface

[44] The RSA key is part of a private key pair owned by Forum. The key is stored in a protected Java Keystore at Forum's headquarters. Only the public key is installed on the TOE.

combination of both. The key used for verifying fingerprints is hard coded into the TOE and stored in persistent memory on a hard disk drive.

During the upgrade process, the fingerprint of the upgrade file is determined and verified by comparing a newly-generated fingerprint with the one contained in the update file, thereby guaranteeing that the updates are both valid and trusted. If the comparison is successful (the values match) then the update occurs. If the comparison is unsuccessful (the values do not match) then the update fails and is not installed.

At power up, the TOE runs a suite of self-tests to check for the correct operation of the cryptographic functionality provided by the cryptographic module. A description of each self-test is given in Table 15 below. Each description provides an explanation on how the execution of these tests ensures the correct operation of the cryptographic functionality.

If any of these tests fail, then the TOE enters an error state where all external ports and interfaces are shut down and only an administrator can log in via the local console.

Table 15 Self-Test Descriptions

Self-Test	Description
AES KAT[45]	The AES KAT encrypts a known plaintext with known keys. It then compares the resultant ciphertext with the expected ciphertext hard-coded in the TOE. If the two values differ, then the KAT fails. If the two values agree, the AES KAT then decrypts the ciphertext with the known keys and compares the decrypted text with the known plaintext. If they differ, then the test fails. If they are the same, then the test passes.
Triple-DES KAT	The Triple-DES KAT takes known keys and plaintext value, which is encrypted and compared to the expected ciphertext value. If the values differ, the test is failed. The Triple-DES KAT then reverses this process by taking a known ciphertext value and keys and performing decryption. The result is compared to the known plaintext value. If the values differ, the test is failed. If they are the same, the test is passed.
SHA-1 KAT	The KAT takes a specific value and hashes it. This digest value is then compared to the known value. If the values differ, the test fails. If they are the same, the test passes.
SHA-224 KAT	The KAT takes a specific value and hashes it. This digest value is then compared to the known value. If the values differ, the test fails. If they are the same, the test passes.
SHA-256 KAT	The KAT takes a specific value and hashes it. This digest value is then compared to the known value. If the values differ, the test fails. If they are the same, the test passes.
SHA-384 KAT	The KAT takes a specific value and hashes it. This digest value is then compared to the known value. If the values differ, the test fails. If they are the same, the test passes.
SHA-512 KAT	The KAT takes a specific value and hashes it. This digest value is then compared to the known value. If the values differ, the test fails. If they are the same, the test passes.

[45] KAT – Known Answer Test

Self-Test	Description
HMAC SHA-1 KAT	The KAT creates a MAC[46] using a known message and known key. This MAC value is then compared to the expected MAC value. If the values differ, the test fails. If they are the same, the test passes.
HMAC SHA-224 KAT	The KAT creates a MAC using a known message and known key. This MAC value is then compared to the expected MAC value. If the values differ, the test fails. If they are the same, the test passes.
HMAC SHA-256 KAT	The KAT creates a MAC using a known message and known key. This MAC value is then compared to the expected MAC value. If the values differ, the test fails. If they are the same, the test passes.
HMAC SHA-384 KAT	The KAT creates a MAC using a known message and known key. This MAC value is then compared to the expected MAC value. If the values differ, the test fails. If they are the same, the test passes.
HMAC SHA-512 KAT	The KAT creates a MAC using a known message and known key. This MAC value is then compared to the expected MAC value. If the values differ, the test fails. If they are the same, the test passes.
Firmware Integrity Test	The TOE creates a SHA-1 hash of the firmware files and compares it against the stored copy of the hash. If the values differ, the test fails. If they are the same, the test passes.

The TOE also performs Power-On Self-Tests (POSTs) at startup to ensure the proper functioning of the system. If this test fails then the bootstrap process is halted and an error code is emitted.

TOE Security Functional Requirements Satisfied: FPT_APW_EXT.1, FPT_SKP_EXT.1, FPT_STM.1, FPT_TST_EXT.1, FPT_TUD_EXT.1.

7.1.7 TOE Access

The TOE terminates local and remote management sessions after an administrator-configurable time period of inactivity. Administrators may also terminate their sessions voluntarily via the logout button on each page of the GUI, or via the `exit` command from the CLI. Users must log in again to regain access to TOE management capabilities. At the login screen of the GUI and at the login prompt for the CLI, administrators are shown an advisory notice and consent warning message regarding unauthorized use of the TOE. The message is shown to users of both the GUI and CLI.

TOE Security Functional Requirements Satisfied: FTA_SSL.3, FTA_SSL.4, FTA_SSL_EXT.1, FTA_TAB.1.

7.1.8 Trusted Path/Channels

The TOE provides a trusted path between the TOE management interfaces and remote TOE administrators. These interfaces are the GUI and CLI. The GUI is protected via HTTPS, while the CLI is protected via SSH. These protocols and the cryptography they implement provide adequate defense against unauthorized disclosure and detection of modification of data being communicated. Additionally, the TOE protects syslog traffic by encrypting it with a secure TLS tunnel. This tunnel prevents unauthorized disclosure and detects if the audit data is modified while being transmitted to the remote syslog server. The TOE does not communicate with any other servers or network devices in the evaluated configuration.

[46] MAC – Message Authentication Code

TOE Security Functional Requirements Satisfied: FTP_ITC.1, FTP_TRP.1.

8 Rationale

8.1 Conformance Claims Rationale

This Security Target conforms to Part 2 extended and Part 3 conformant of the Common Criteria Standard for Information Technology Security Evaluations, Version 3.1 Revision 4. This ST conforms to the Protection Profile for Network Devices v1.1 dated June 8, 2012 (referred to as the NDPP).

8.1.1 Variance Between the PP and this ST

In some instances changes were made in this ST from the NDPP. All of these changes are documented below with a rationale for the change.

- Some SFRs have been modified from the NDPP text to include portions from the CEM that were stricken as part of the refinement operation. This text has been stricken within the SFR.
- Several inconsistencies are present with NDPP with respect to component naming conventions. In some cases, Extended SFRs are labeled appropriately, but not in others, e.g. “FCS_TLS_EXT.1 Explicit: TLS”, or “FIA_UIA_EXT.1 User identification and authentication”. To remain consistent, all Extended SFRs in this ST have been designated as such.
- Several SFRs in the NDPP include the word “refinement” to imply that they have been refined. SFRs included in this ST have been defined with the appropriate conventions to indicate refinements or other operations, as described in section 6.1. Therefore, the labeling of such is redundant, and as a result, the word “refinement” has been removed.
- The auditable events per FAU_GEN.1 contain events for FTP_TRP.1 Trusted path functions; however, the trusted path in certain cases is incorrectly identified as a trusted channel. Therefore, all instances of “channel” have been changed to “path” in the auditable events for FTP_TRP.1.

8.1.2 Security Assurance Requirements Rationale

This ST maintains exact conformance to NDPP v1.1, including the assurance requirements listed in section 4.3 of NDPP.

8.1.3 Dependency Rationale

This ST does satisfy all the requirement dependencies of the Common Criteria. Table 16 lists each requirement to which the TOE claims conformance with a dependency and indicates whether the dependent requirement was included. As the table indicates, all dependencies have been met.

Table 16 Functional Requirements Dependencies

SFR ID	Dependencies	Dependency Met	Rationale
FAU_GEN.1	FPT_STM.1	✓	
FAU_GEN.2	FAU_GEN.1	✓	
	FIA_UID.1	✓	Although FIA_UID.1 is not included, FIA_UIA_EXT.1 provides coverage for user identification and authentication which supersedes FIA_UID.1.
FAU_STG_EXT.1	FAU_GEN.1	✓	
FCS_CKM.1	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
	FCS_COP.1(1)	✓	
	FCS_COP.1(2)	✓	
	FCS_COP.1(3)	✓	
	FCS_COP.1(4)	✓	
FCS_CKM_EXT.4	FCS_CKM.1	✓	
FCS_COP.1(1)	FCS_CKM.1	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
FCS_COP.1(2)	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
	FCS_CKM.1	✓	
FCS_COP.1(3)	FCS_CKM.1	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
FCS_COP.1(4)	FCS_CKM.1	✓	
	FCS_CKM.4	✓	Although FCS_CKM.4 is not in the ST, FCS_CKM_EXT.4 provides coverage.
FCS_HTTPS_EXT.1	FCS_TLS_EXT.1	✓	

SFR ID	Dependencies	Dependency Met	Rationale
FCS_RBG_EXT.1	No dependencies	✓	
FCS_SSH_EXT.1	FCS_COP.1(1)	✓	
FCS_TLS_EXT.1	FCS_COP.1(2)	✓	
	FCS_COP.1(3)	✓	
	FCS_COP.1(1)	✓	
FDP_RIP.2	No dependencies	✓	
FIA_PMG_EXT.1	No dependencies	✓	
FIA_UAU_EXT.2	No dependencies	✓	
FIA_UAU.7	FIA_UAU.1	✓	Although FIA_UAU.1 is not included, FIA_UIA_EXT.1 provides coverage for user identification and authentication which supersedes FIA_UAU.1.
FIA_UIA_EXT.1	FTA_TAB.1	✓	
FMT_MTD.1	FMT_SMF.1	✓	
	FMT_SMR.1	✓	
FMT_SMF.1	No dependencies	✓	
FMT_SMR.2	FIA_UID.1	✓	Although FIA_UID.1 is not included, FIA_UIA_EXT.1 provides coverage for user identification and authentication which supersedes FIA_UID.1.
FPT_APW_EXT.1	No dependencies	✓	
FPT_SKP_EXT.1	No dependencies	✓	
FPT_STM.1	No dependencies	✓	
FPT_TST_EXT.1	No dependencies	✓	
FPT_TUD_EXT.1	FCS_COP.1(3)	✓	
FTA_SSL_EXT.1	FIA_UAU.1	✓	Although FIA_UAU.1 is not included, FIA_UIA_EXT.1 provides coverage for user identification and authentication which supersedes FIA_UAU.1.
FTA_SSL.3	No dependencies	✓	
FTA_SSL.4	No dependencies	✓	

SFR ID	Dependencies	Dependency Met	Rationale
FTA_TAB.1	No dependencies	✓	
FTP_ITC.1	No dependencies	✓	
FTP_TRP.1	No dependencies	✓	

9 Acronyms and Terms

This section describes the acronyms and terms.

9.1 Terminology

Table 17 Terms

Name	Definition
Authorized Administrator	A user with administrator TOE access that has been successfully identified and authenticated by the TOE.
Domain parameters	DSA requires that the private/public key pairs used for digital signature generation and verification be generated with respect to a particular set of domain parameters. These domain parameters may be common to a group of users and may be public. A user of a set of domain parameters (i.e., both the signatory and the verifier) shall have assurance of their validity prior to using them. Although domain parameters may be public information, they shall be managed so that the correct correspondence between a given key pair and its set of domain parameters is maintained for all parties that use the key pair. A set of domain parameters may remain fixed for an extended time period. The domain parameters for DSA are the integers p, q, and g, and optionally, the domain_parameter_seed and counter that were used to generate p and q (i.e., the full set of domain parameters is (p, q, g {, domain_parameter_seed, counter})).
Hardware-based noise source	A hardware random number generator is an apparatus that generates random numbers from a physical process. Such devices are often based on microscopic phenomena that generate a low-level, statistically random "noise" signal, such as thermal noise or the photoelectric effect or other quantum phenomena. These processes are, in theory, completely unpredictable, and the theory's assertions of unpredictability are subject to experimental test. A hardware random number generator typically consists of a transducer to convert some aspect of the physical phenomena to an electrical signal, an amplifier and other electronic circuitry to increase the amplitude of the random fluctuations to a macroscopic level, and some type of analog to digital converter to convert the output into a digital number, often a simple binary digit 0 or 1. By repeatedly sampling the randomly varying signal, a series of random numbers is obtained.
Target network	The domain of network and managed devices to be analyzed by the TOE.

9.2 Acronyms

Table 18 Acronyms

Acronym	Definition
AES	Advanced Encryption Standard
API	Application Programming Interface
CBC	Cipher Block Chaining
CC	Common Criteria
CEM	Common Evaluation Methodology
CLI	Command Line Interface
CM	Configuration Management
CMAC	Cipher-based Message Authentication Code
CSP	Critical Security Parameters
CTR	Counter Mode
DES	Data Encryption Standard
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
EC	Elliptical Curve
ECB	Electronic Code Book
ECDSA	Elliptical Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standard
FTP	File Transfer Protocol
GCM	Galois/Counter Mode
GNU	GNU's Not Unix
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
HSM	Hardware Security Module
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ID	Identifier
IP	Internet Protocol
IT	Information Technology
KAT	Known Answer Test
LDAP	Lightweight Directory Access Protocol

Acronym	Definition
MAC	Message Authentication Code
NDPP	Network Devices Protection Profile
NIST	Institute of Standards and Technology
NTP	Network Time Protocol
OS	Operating System
OSP	Organizational Security Policy
PCIe	Peripheral Component Interconnect Express
PGP	Pretty Good Privacy
PP	Protection Profile
PUB	Publication
RFC	Request for Comment
RNG	Random Number Generator
RSA	Rivest, Shamir, and Adelman (cryptographic algorithm)
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SFTP	Secure File Transfer Protocol
SHA	Secure Hash Algorithm
SP	Special Publication
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functionality
URI	Uniform Resource Identifier
WSDL	Web Services Description Language
XML	Extensible Markup Language

Prepared by:
Corsec Security, Inc.

13135 Lee Jackson Memorial Highway
Suite 220
Fairfax, VA 22033
United States of America

Phone: +1 703 267 6050
Email: info@corsec.com
http://www.corsec.com