Arista Networks 7050X and 7150 Series Security Target

Document Version

13-2624-R-0001

Version: 1.11

June 20, 2014

Prepared For:

Arista Networks

5453 Great America Parkway

Santa Clara, CA, 95054

Prepared By:

Scott Cutler and Kenji Yoshino

InfoGard Laboratories

709 Fiero Ln., Suite 25

San Luis Obispo, CA, 93401

Notices:

Table of Contents

Tables

1 Security Target (ST) Introduction

- The ST introduction shall contain an ST reference, a TOE reference, a TOE overview and a TOE description.
- The ST reference shall uniquely identify the ST.
- The TOE reference shall identify the TOE.

The structure of this document is defined by CC v3.1r3 Part 1 Annex A.2, "Mandatory contents of an ST":

- Section 1 contains the ST Introduction, including the ST reference, Target of Evaluation (TOE) reference, TOE overview, and TOE description.
- Section 2 contains conformance claims to the Common Criteria (CC) version, Protection Profile (PP) and package claims, as well as rationale for these conformance claims.
- Section 3 contains the security problem definition, which includes threats, Organizational Security Policies (OSP), and assumptions that must be countered, enforced, and upheld by the TOE and its operational environment.
- Section 4 contains statements of security objectives for the TOE, and the TOE operational environment as well as rationale for these security objectives.
- Section 5 contains definitions of any extended security requirements claimed in the ST.
- Section 6 contains the security function requirements (SFR), the security assurance requirements (SAR), as well as the rationale for the claimed SFR and SAR.
- Section 7 contains the TOE summary specification, which includes the detailed specification of the IT security functions

1.1 Security Target Reference

The Security Target reference shall uniquely identify the Security Target.

ST Title:	Arista Networks 7050X and 7150 Series Security Target
ST Version Number:	Version 1.11
ST Author(s):	Scott Cutler and Kenji Yoshino
ST Publication Date:	June 20, 2014
Keywords:	Network Device, Switch, LAN, Router

1.2 Target of Evaluation Reference

The Target of Evaluation reference shall identify the Target of Evaluation.

TOE Developer	Arista Networks
	5453 Great America Parkway
	Santa Clara, CA, 95054
TOE Name	Arista 7050QX Series: DCS-7050QX-32-F, DCS-7050QX-32-R; EOS V4.13.3.4
	Arista 7150 Series: DCS-7150S-24-F, DCS-7150S-24-R, DCS-7150S-24#, DCS-7150S-24-CL#, DCS-7150S-24-CLD#, DCS-7150S-52-CL-F, DCS-7150S-52-CL-R,

	DCS-7150S-52-CL#, DCS-7150S-52-CLD#, DCS-7150S-64-CL-F, DCS-7150S-64-CL-R, DCS-7150S-64-CL#, DCS-7150S-64-CLD#; EOS V4.13.3.4
TOE Version	Arista 7050QX Series: CPU: AMD Turion(tm) II Neo N41H Dual-Core Processor, Security Chip: R5H30211, Forwarding ASIC: Linecard0/0: Chip: BCM56850 Arista 7150 Series: CPU: AMD Turion(tm) II NEO N41H Dual-Core Processor, Forwarding ASIC: Alta, Security Chip: R5H30211

1.3 Target of Evaluation Overview

1.3.1 TOE Product Type

The TOE is classified as a Network Device (a generic infrastructure device that can be connected to a network).

1.3.2 TOE Usage

The TOE is a Network Device that provides layer 2, 3, and 4 Ethernet network management and interconnectivity. The Ethernet management layers refer to the Open Systems Interconnection (OSI) model layers. They refer to the data link, network, and transport layers respectively. It also contains a modern Linux-based operating system that allows for complex management solutions. It is designed with high performance electronics to meet the needs of latency-critical applications such as financial Electronic Communication Networks (ECNs) or High Performance Computing (HPC) clusters.

The TOE can direct and filter network packets based on the contents within each of these layers. It is also capable of supporting many modern layer-specific traffic management features including the following *unevaluated* features:

- 802.1w, 802.1s Spanning Tree Protocol (STP)
- 802.3ad and Multi-Chassis Link Aggregation
- 802.3x Flow Control
- Virtual Local Area Networks (VLANs)
- IPv4/IPv6 routing and Network Address Translation (NAT)
- Access Control Lists (ACLs)
- Virtualization support (VXLAN and VMware)
- Quality of Service (QoS) rate limiting and queuing
- Congestion monitoring and management

The TOE supports remote administration over the Secure Shell v2 (SSHv2) protocol that supports cryptographic encryption and authentication using FIPS-certified algorithms. Remote administration is configured using an internal role-based access control system that allows for flexible administrator permissions and capabilities.

The TOE also supports storage and forwarding of detailed audit logs. The process that manages audit messages is capable of forwarding audit messages, encrypted using SSHv2, to any syslog-compatible network entity.

1.3.3 TOE Major Security Features

- Audit
  - Generates audit events and stores them locally and/or remotely.
  - Supports secure communication to remote syslog-compatible audit servers.

- Cryptography
  - Several NIST-recommended cryptographic algorithms and data zeroization help support secure communications and device functionality.
- User Data Protection
  - User information is stored and forwarded in a secure manner to ensure data is not leaked.
- Identification and Authentication
  - Password and user access polices are enforced by the TOE.
- Security Management
  - Administrators may locally or remotely log into the TOE to administer the TSF functionality.
- Protection of the TSF
  - Critical security parameters are protected from disclosure.
  - Self-tests are run each power-up to help ensure the correct functionality of the TSF.
- TOE Access
  - Administrative sessions are protected using timeout and log-out mechanisms.
- Trusted Path/Channels
  - The TOE provides cryptographically secure communication channels and paths between administrators and authorized IT entities.

1.3.4 TOE IT environment hardware/software/firmware requirements

1.3.4.1 Network Requirements

For remote storage of audit logs, a remote syslog server capable of SSHv2 tunneled connections is required. It must be reachable over the Ethernet administrative network interface. The SSHv2 implementation on the audit server must be capable of the following cryptographic algorithms: RSA-2048 for public-key authentication, AES-128/256 CBC for data encryption, HMAC-SHA1 for data integrity, and diffie-hellman-group14-sha1 for key exchange.

For time synchronization a remote server capable of NTPv4 (RFC 5095) is required. The server must be reachable over the Ethernet administrative network interface.

1.3.4.2 Software Requirements

For management over the console port a RS232 terminal application for a Personal Computer (PC) is required.

For management over SSH a client capable of connecting over SSHv2 is required. The client must be capable of the algorithms detailed in the Network Requirements section above.

1.3.4.3 Hardware Requirements

For each non-administrative network interface a separate Small Form Factor Pluggable (SFP+), or Quad Small Form-factor Pluggable (QSFP+) module is required to provide physical Ethernet connectivity. Further explanations of SFPs and QSFPs are provided in section 1.4. The TOE supports the following SFP and QSFP interfaces:

Table 1: 7150 SFP Interfaces		
SFP Interface	SFP Type	Speed (gigabit)

Table 1: 7150 SFP Interfaces		
SFP Interface	SFP Type	Speed (gigabit)
40GBASE-CR4	QSFP+	40
40GBASE-SR4	QSFP+	40
40GBASE-LR4	QSFP+	40
10GBASE-CR	QSFP+ / SFP+	10
10GBASE-SRL	SFP+	10
10GBASE-SR	SFP+	10
10GBASE-LR	SFP+	10
10GBASE-ER	SFP+	10
10GBASE-DWDM	SFP+	10
1GbE-SX	SFP+	1
1GbE-LX	SFP+	1
1GbE-TX	SFP+	1
100Mb-TX	SFP+	0.1

Table 2: 7050QX SFP Interfaces		
SFP Interface	SFP Type	Speed (gigabit)
40GBASE-CR4	QSFP+	40
40GBASE-SR4	QSFP+	40
AOC-40G-Q-Q	QSFP+	40
40GBASE-XSR4	QSFP+	40
40G-LRL4	QSFP+	40
40GBASE-LR4	QSFP+	40
10GBASE-CR	QSFP+ / SFP+	10

For management over the console port a PC with a serial communications port is required.

The following models of the TOE do not include fans or power supplies: DCS-7150S-24#, DCS-7150S-24-CL#, DCS-7150S-24-CLD#, DCS-7150S-52-CL#, DCS-7150S-52-CLD#, DCS-7150S-64-CL#, DCS-7150S-64-CLD#. For these models, the user must install at least one fan and one power supply, although the typical configuration contains two of each.

The TOE supports the following models of fans and power supplies:

Table 3: Supported Fans and Power Supplies	
Product Number	Description
FAN-7000-F	Front-to-rear airflow fan module
FAN-7000-R	Rear-to-front airflow fan module
PWR-460AC-F	460 Watt AC PSU with front-to-rear airflow
PWR-460AC-R	460 Watt AC PSU with rear-to-front airflow
PWR-460DC-F	460 Watt DC PSU with front-to-rear airflow
PWR-460DC-R	460 Watt DC PSU with rear-to-front airflow

1.4 Target of Evaluation Description

The TOE is a network switch that is intended to connect many Ethernet-based network devices together in an enterprise environment while maintaining security.

Each non-administrative network interface uses a small form-factor pluggable (SFP) transceiver, to provide connectivity between the network device motherboard and a fiber optic or copper cable. This allows the customer to use several different types of network cables with the network device. The list of compatible SFPs is provided in Table 1 and the user guidance.

Each model of the TOEs under evaluation varies by the amount and type of SFPs the hardware. The 7150S-24 supports 24 separate SFP+ modules, the 7150S-52 supports 52 separate SFP+ modules, and the 7150S-64 supports 48 separate SFP+ modules and 4 Quad Small Form-factor Pluggable (QSFP+) modules. The 7050QX supports 32 separate QSFP+ modules. The TOE security chip refers to the Renesas chip of the same model number that serves as an entropy source.

The Arista Extensible Operating System, or Arista EOS, is built upon the mainline Linux kernel (www.kernel.org) and an x86 dual-core CPU.

1.4.1 Target of Evaluation Physical Boundaries

The TOE consists of the following hardware:

Arista 7150 Series DCS-7150S-24
Arista 7150 Series DCS-7150S-52
Arista 7150 Series DCS-7150S-64
Arista 7050X Series DCS-7050QX-32

Which runs the following software:

Arista Extensible Operating System (EOS) 4.13.3.4

And includes the following documentation that are available online:

Common Criteria Guidance Supplement Arista 7150, 7050X, 7250X, 7300X and 7500E Series Switches Guidance Document AGD_OPE.1, AGD_PRE.1, 1.4, June 10, 2014
Arista User Manual, EOS version 4.13.3.4, June 9, 2014
Arista EOS System Message Guide, Software Release 4.13.3.4, April 18, 2014
Arista Quick Start Guide 7000 Series 1 RU – Gen 2Data Center Switches, PDOC-00019-13

1.4.2 Target of Evaluation Logical Boundaries

The logical boundaries of the TOE include those security functions implemented exclusively by the TOE. These security functions are summarized in Section 1.3.3 above and are further described in the following subsections. A more detailed description of the implementation of these security functions is provided in Section 7, “TOE Summary Specification.”

1.4.2.1 Audit

The Arista EOS uses an internal syslog process that receives, stores, and forwards auditable events from all system processes. When a user or system process triggers applicable TSF functionality an audit message is generated, and sent to the internal syslog process. These events are then sent to an external audit server for storage and review by an administrator. The communication between the TOE and external audit server is protected by tunneling the syslog protocol through an encrypted SSH tunnel.

1.4.2.2 Cryptographic Operations

The TSF performs the following cryptographic operations:

- SSH with the following algorithms:

    - RSA-2048 for public-key authentication (FIPS algorithm cert# 1315)
    - AES-128/256 CBC for data encryption (FIPS algorithm cert# 2567)
    - HMAC-SHA1 for data integrity (FIPS algorithm cert# 1584)
    - diffie-hellman-group14-sha1 for key exchange
- SHA-512 for the following purposes: (FIPS algorithm cert# 2163)
    - Local administrator password storage and authentication
    - CLI "verify" function which allows the SHA-512 hash calculation of any file
- SHA-1 for the following purposes: (FIPS algorithm cert# 2163)
    - Used within HMAC-SHA1 and diffie-hellman-group14-sha1
- HMAC-SHA1 for the following purposes: (FIPS algorithm cert# 1584)
    - SSH data integrity
- Random bit generation using FIPS 140-2 X9.31-AES (FIPS algorithm cert# 1218)

The TSF uses the OpenSSL library to manage session-based plaintext secret and private cryptographic keys and Critical Security Parameters, (CSPs). During an SSH session, CSPs are stored within a central data structure and all functions are given pointers to the single instance of that data structure. After the CSPs are no longer needed and the session has terminated, a function call overwrites the entire data structure that contains all private keys and CSPs with zeros.

The TOE requires the user to perform a separate procedure in order to zeroize and replace the private RSA keys that are no longer used for SSH authentication.

1.4.2.3 User Data Protection

The TOE uses various software and hardware mechanisms to ensure that network packets traveling through the TOE are not re-used or accessible once they have finished being used by the TOE. The hardware packet-routing architecture is built without the use of padding to ensure that all data is passed between components exactly as-is. Therefore, when an Ethernet packet is received by the switch, the exact size of the packet is known and allocated for in global memory. When a packet is stored within global memory it is stored along with metadata to ensure packet integrity.

The Linux kernel API, which handles padding in a safe manner, is leveraged to generate packets internally. If the kernel is given a payload that does not meet the minimum payload size requirement it will pad the payload with zeros. In addition, the kernel will not accept payloads with a bit length non-divisible by eight. Therefore, each individual system process is responsible for creating a payload that does not require padding past the minimum length requirement. These features together protect user data from being disclosed.

1.4.2.4 Identification and Authentication

The TOE supports password authentication for administrative users over console and SSH. The TOE also supports RSA key-based authentication for administrative users over SSH. The TOE stores administrator passwords locally using SHA-512 hashing and allows special characters and passwords in excess of 15 characters. The TOE contains an AAA (authentication, authorization and accounting) module that stores and manages permissions for users. The AAA module stores the privilege level of each user along with all other information required to access the TOE. The TOE enforces that administrative users authenticate through this mechanism before performing any administrative actions.

This document uses the terms administrator as a term for a user with the ability to log into the TOE locally or remotely and administer it. The root account is an administrator account, disabled by default, that fits the above description but falls outside the CC-evaluation because of its ability to access another

CLI interface, Bash. The bash interface is undocumented and treated as a maintenance tool that is outside the scope of the evaluation. The use of the bash interface, along with the root account, creates security impacts that are discussed in Section 7.6.1.

1.4.2.5 Security Management

The TOE allows a remote administrator to manage the TOE using a local RS-232 console, or remotely using an SSHv2 session. The TOE provides a custom CLI interface to administer the TOE, which provides authentication and restricts the ability to manage the TOE to security administrators. The TOE also provides administrators with the ability to update the TOE and verify their integrity using SHA-512 hashing algorithm.

1.4.2.6 Protection of the TSF

The TOE protects TSF data from disclosure using different cryptographic methods and security-functionality. The TOE provides administrative access to users through a CLI that enforces user and group profiles established by the local role-based access control provided by EOS. The administrator configures user profiles that specify varying degrees of access to the system. The limited CLI, user account system, and underlying file system permissions serve to restrict access to TSF data such as private keys. Plaintext private keys used for SSH authentication are stored on internal flash which is only accessible through CLI commands performed by the local root account. Local administrator passwords are stored by the TOE and kept in a hashed form so that they cannot be read in plaintext format.

The TOE derives a reliable time source for logging and other system processes through the local NTP service. The exact time can be provided by setting the value locally, or through synchronizing the time from an external server via NTP.

When updating TSF functionality, a published cryptographic hash of the updated software is provided to the user to ensure the integrity of the software.

The TOE is also able to verify that TSF protection is functioning properly by running a memory test at boot-time and several diagnostic tools throughout the operation of the TOE. During the EOS boot sequence the TOE also initializes the OpenSSL FIPS self-tests against each cryptographic algorithm supported by SSH.

1.4.2.7 TOE Access

In order to mitigate unauthorized access to the TOE, administrative sessions can be terminated manually or automatically. If an administrator accesses the TOE, the session may be terminated by the administrator's own actions or automatically after a specified time of inactivity. These termination features apply to both local and remote connections to the TOE.

The TOE will also display a customizable warning message that is displayed to the user during each administrative logon. The message is designed to serve as an advisory notice and consent warning regarding use of the TOE.

1.4.2.8 Trusted Path/Channels

The TOE implements and requires a secured method of communication between itself, audit servers, and remote administrators. In order to accomplish a secure connection to external devices, the TOE uses an SSHv2 connection with RSA based authentication and AES-based encryption. A private/public key pair can either be generated by the TOE or imported from another device and imported into the TOE. After an SSHv2 connection is authenticated via RSA key pairs, AES encryption keys are exchanged via diffie-

hellman-group14-sha1 key exchange algorithm. After these steps, all further traffic between the TOE and the external device is encrypted via AES-128/256-CBC encryption. This method provides assured identification of the external device and prevents disclosure or undetected modification of data across the communication channel. Communications between the TOE and IT entities may be initiated from either the TOE or the IT entity.

Remote administrators may also create a secured connection to the TOE that provides cryptographic authentication and protection of data. Remote administrators connecting to the TOE via SSHv2 have the option of using password-based authentication or RSA key-based authentication

1.5 Notation, formatting, and conventions

The notation, formatting, and conventions used in this Security Target are defined below; these styles and clarifying information conventions were developed to aid the reader.

Where necessary, the ST author has added application notes to provide the reader with additional details to aid understanding; they are italicized and usually appear following the element needing clarification. Those notes specific to the TOE are marked "TOE Application Note;" those taken from the ND Protection Profile are marked "PP Application Note."

The notation conventions that refer to iterations, assignments, selections, and refinements made in this Security Target are in reference to SARs and SFRs taken directly from CC Part 2 and Part 3 as well as any SFRs and SARs taken from a protection profile.

The notation used in the PP to indicate iterations, assignments, selections, and refinements of SARs and SFRs taken from CC Part 2 and Part 3 is not carried forward into this document. Additionally, obvious errors in the PP are corrected and noted as such.

The CC permits four component operations (assignment, iteration, refinement, and selection) to be performed on requirement components. These operations are defined in Common Criteria, Part 1; paragraph 6.4.1.3.2, "Permitted operations on components" as:

- Iteration: allows a component to be used more than once with varying operations;
- Assignment: allows the specification of parameters;
- Selection: allows the specification of one or more items from a list; and
- Refinement: allows the addition of details.

Iterations made by the ST and PP author are indicated by a number in parenthesis following the requirement number, e.g., FIA_UAU.1.1(1); the iterated requirement titles are similarly indicated, e.g., FIA_UAU.1(1).

Assignments made by the ST author are identified with **bold text.**

Selections are identified with <u>underlined text</u>**.**

Refinements that add text use ***bold and italicized text*** to identify the added text**.** Refinements that performs a deletion, identifies the deleted text with ~~***strikeout, bold, and italicized text***~~.

2 Conformance Claims

2.1 Common Criteria Conformance Claims

This Security Target is conformant to the Common Criteria Version 3.1r3.

2.2 Conformance to Protection Profiles

This Security Target claims exact conformance to the Network Device Protection Profile, Version 1.1, dated June 8, 2012 [10], including the Security Requirements for Network Devices Errata #1 [11].

2.3 Conformance to Security Packages

This Security Target does not claim conformance to any security function requirements package, neither as package-conformant or package-augmented.

2.4 Conformance Claims Rationale

The ST claims exact conformance to the Network Device Protection Profile, Version 1.1, June 8, 2012. This Protection Profile will be called NDPP or PP for convenience through this Security Target.

The following address the completeness of the threats, OSP, objectives, and limitations on the assumptions:

- Threats
  - All threats defined in the NDPP are carried forward to this ST
  - No additional threats have been defined in this ST
- Organizational Security Policies
  - All OSP defined in the NDPP are carried forward to this ST;
  - No additional OSP have been defined in this ST.
- Assumptions
  - All assumptions defined in the NDPP are carried forward to this ST;
  - No additional assumptions for the operational environment have been defined in this ST
- Objectives
  - All objectives defined in the NDPP are carried forward to this ST

All SFRs and SARs defined in the NDPP are carried forward to this Security Target.

Rationale presented in the body of this ST shows all assumptions on the operational environment have been upheld, all the OSP are enforced, all defined objectives have been met and these objectives counter the defined threats.

Additionally, all SFRs and SARs defined in the NDPP have been properly instantiated in this Security Target; therefore, this ST shows exact conformance to the NDPP.

3 Security Problem Definition

3.1 Threats

The following table defines the security threats for the TOE, characterized by a threat agent, an asset, and an adverse action of that threat agent on that asset. These threats are taken directly from the PP unchanged.

Table 4: Threats	
Threat	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.

3.2 Organizational Security Policies

The following table defines the organizational security policies which are a set of rules, practices, and procedures imposed by an organization to address its security needs. These threats are taken directly from the PP unchanged.

Table 5: Organizational Security Policies	
OSP	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 assumptions that are made on the operational environment in which the TOE is intended to be used in order to be able to provide security functionality. It includes information about the physical, personnel, and connectivity aspects of the environment. The operational environment must be managed in accordance with the provided guidance documentation. The following table defines specific conditions that are assumed to exist in an environment where the TOE is deployed. These assumptions are taken directly from the PP unchanged.

Table 6: Assumptions	
Assumption	Description
A.NO_GENERAL_PURPOSE	It is assumed that there are no general-purpose computing capabilities (e.g., compilers or user applications) available to the TOE, other than those services

Table 6: Assumptions	
Assumption	Description
	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

4.1 Security Objectives for the TOE

Table 7: Security Objectives for the TOE	
TOE Objective	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

Table 8: Security Objectives for the Operational Environment	
Objective	Description
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available to the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is assumed to be provided by the IT environment.
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all administrator guidance in a trusted manner.

5 Extended Components Definition

This section provides definition of the extended security functional and assurance requirements; the components that are CC Part 2 extended, and CC Part 3 extended, i.e., NIAP interpreted requirements, and extended requirements.

5.1 Extended Security Functional Requirements Definitions

There are no extended Security Functional Requirements defined in this Security Target. All extended SFRs were taken from the NDPP.

5.2 Extended Security Assurance Requirement Definitions

There are no extended Security Assurance Requirements defined in this Security Target. All extended SARs were taken from the NDPP.

6 Security Requirements

This section describes the security functional and assurance requirements for the TOE.

6.1 Security Function Requirements

This section describes the functional requirements for the TOE. The security functional requirement components in this security target are CC Part 2 conformant or CC Part 2 extended as defined in Section 2, Conformance Claims. Operations that were performed in the NDPP are not signified in this section. Operations performed by the ST are denoted according to the formatting conventions in section 1.5.

6.1.1 Security Audit (FAU)

6.1.1.1 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record for 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 9.

Table 9: Auditable Events

#	SFR	Auditable Events	Additional Audit Record Contents
1.	FAU_GEN.1	None.	
2.	FAU_GEN.2	None.	
3.	FAU_STG_EXT.1	None.	
4.	FCS_CKM_EXT.4	None.	
5.	FCS_COP.1(1)	None.	
6.	FCS_COP.1(2)	None.	
7.	FCS_COP.1(3)	None.	
8.	FCS_COP.1(4)	None.	
9.	FCS_RGB_EXT.1	None.	
10.	FDP_RIP.2	None.	
11.	FIA_PMG_EXT.1	None.	
12.	FIA_UIA_EXT.1	All use of the identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
13.	FIA_UAU_EXT.1	All use of the authentication mechanism.	Origin of the attempt (e.g., IP address).
14.	FIA_UAU.7	None.	
15.	FMT_MTD.1	None.	
16.	FMT_SMF.1	None.	
17.	FMT_SMR.2	None.	
18.	FPT_SKP_EXT.1	None.	

Table 9: Auditable Events			
#	SFR	Auditable Events	Additional Audit Record Contents
19.	FPT_APW_EXT.1	None.	
20.	FPT_STM.1	Changes to the time.	The old and new values for the time. Origin of the attempt (e.g., IP address).
21.	FPT_TUD_EXT.1	Initiation of update.	No additional information.
22.	FPT_TST_EXT.1	None.	
23.	FTA_SSL_EXT.1	Any attempts at unlocking of an interactive session.	No additional information.
24.	FTA_SSL.3	The termination of a remote session by the session locking mechanism.	No additional information.
25.	FTA_SSL.4	The termination of an interactive session.	No additional information.
26.	FTA_TAB.1	None.	
27.	FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
28.	FTP_TRP.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the claimed user identity.
29.	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.

PP Application Note: *The ST author can include other auditable events directly in the table; they are not limited to the list presented.*

Many auditable aspects of the SFRs included in this document deal with administrative actions. Item c above requires all administrative actions to be auditable, so no additional specification of the auditability of these actions is specified in Table 9.

Assurance Activity: The evaluator shall check the administrative guide and ensure that it lists all of the auditable events and provides a format for audit records. Each audit record format type must be covered, along with a brief description of each field. The evaluator shall check to make sure that every audit event type mandated by the PP is described and that the description of the fields contains the information required in FAU_GEN.1.2, and the additional information specified in Table 8

The evaluator shall also make a determination of the administrative actions that are relevant in the context of the NDPP. The evaluator shall examine the administrative guide and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the

requirements specified in the PP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security relevant with respect to the NDPP. The evaluator may perform this activity as part of the activities associated with ensuring the AGD_OPE guidance satisfies the requirements.

The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records for the events listed in Table 9 and administrative actions. This should include all instances of an event--for instance, if there are several different I&A mechanisms for a system, the FIA_UIA_EXT.1 events must be generated for each mechanism. The evaluator shall test that audit records are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. If HTTPS is implemented, the test demonstrating the establishment and termination of a TLS session can be combined with the test for an HTTPS session. For administrative actions, the evaluator shall test that each action determined by the evaluator above to be security relevant in the context of the NDPP is auditable. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly. For example, testing performed to ensure that the administrative guidance provided is correct verifies that AGD_OPE.1 is satisfied and should address the invocation of the administrative actions that are needed to verify the audit records are generated as expected.

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

1. Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
2. 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 9.

PP Application Note: *As with the previous component, the ST author should update Table 9 above with any additional information generated. "Subject identity" in the context of this requirement could either be the administrator's user id or the affected network interface, for example.*

Assurance Activity: This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

6.1.1.2 FAU_GEN.2 User Identity Association

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

Assurance Activity: This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

6.1.1.3 FAU_STG_EXT.1 External Audit Trail Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel implementing the SSH protocol.

PP Application Note: *For applications of the NDPP to TOEs that do not act as audit servers, the TOE relies on a non-TOE audit server for storage and review of audit records. Although the TOE generates audit records, the storage of these audit records and the ability to allow the administrator to review these audit records is provided by the operational environment. The ST author chooses the first clause of the first selection in these cases. The NDPP can also be used to specify requirements for an audit server; in this case the second clause of the first selection is used.*

In the second selection, the ST author chooses the means by which this connection is protected. The ST author also ensures that the supporting protocol requirement matching the selection is included in the ST.

Assurance Activity: For both types of TOEs (those that act as an audit server and those that send data to an external audit server), there is some amount of local storage. The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally; what happens when the local audit data store is full; and how these records are protected against unauthorized access. The evaluator shall also examine the operational guidance to determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server (for TOEs that are not acting as an audit log server). For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and "cleared" periodically by sending the data to the audit server.

TOE acts as audit server: The evaluator shall examine the TSS to ensure it describes the connection supported from non-TOE entities to send the audit data to the TOE, and how the trusted channel is provided. Testing of the trusted channel mechanism will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall also examine the operational guidance to ensure it describes how to establish the trusted channel with the TOE, as well as describe any requirements for other IT entities to connect and send audit data to the TOE (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with other IT entities. The evaluator shall perform the following test for this requirement:

Test 1: The evaluator shall establish a session between an external IT entity and the TOE according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the IT entity and the TOE during several activities of the TOE. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer,

and that they are successfully received by the TOE. The evaluator shall perform this test for each protocol selected in the second selection.

TOE is not an audit server: The evaluator shall examine the TSS to ensure it describes the means by which the audit data are transferred to the external audit server, and how the trusted channel is provided. Testing of the trusted channel mechanism will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall also examine the operational guidance to ensure it describes how to establish the trusted channel to the audit server, as well as describe any requirements on the audit server (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with the audit server. The evaluator shall perform the following test for this requirement:

Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator's choice designed to generate audit data to be transferred to the audit server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing.

6.1.2 Cryptographic Support (FCS)

6.1.2.1 FCS_CKM.1 Cryptographic Key Generation (for asymmetric keys)

FCS_CKM.1.1(1) The TSF shall generate asymmetric cryptographic keys used for key establishment in accordance with

1. NIST Special Publication 800-56A. "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" for finite field-based key establishment schemes;

and specified cryptographic key sizes equivalent to, or greater than, a symmetric key strength of 112 bits.

FCS_CKM.1.1(2) The TSF shall generate asymmetric cryptographic keys used for key establishment in accordance with

2. NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography" for RSA-based key establishment schemes

and specified cryptographic key sizes equivalent to, or greater than, a symmetric key strength of 112 bits.

PP Application Note: *This component requires that the TOE be able to generate the public/private key pairs that are used for key establishment purposes for the various cryptographic protocols used by the TOE (e.g., IPsec). If multiple schemes are supported, then the ST author should iterate this requirement to capture this*

capability. The scheme used will be chosen by the ST author from the selection.

Since the domain parameters to be used are specified by the requirements of the protocol in the NDPP, it is not expected that the TOE will generate domain parameters, and therefore there is no additional domain parameter validation needed when the TOE complies to the protocols specified in the NDPP.

SP 800-56B references (but does not mandate) key generation according to FIPS 186-3. For purposes of compliance in this version of the NDPP, RSA key pair generation according to FIPS 186-2 or FIPS 186-3 is allowed in order for the TOE to claim conformance to SP 800-56B.

The generated key strength of 2048-bit DSA and rDSA keys need to be equivalent to, or greater than, a symmetric key strength of 112 bits. See NIST Special Publication 800-57, "Recommendation for Key Management" for information about equivalent key strengths.

Assurance Activity: The evaluator shall use the key pair generation portions of "The FIPS 186-3 Digital Signature Algorithm Validation System (DSA2VS)", "The FIPS 186-3 Elliptic Curve Digital Signature Algorithm Validation System (ECDSA2VS)", and either "The RSA Validation System (RSAVS)" (for FIPS 186-2) or "The 186-3 RSA Validation System (RSA2VS)" (for FIPS 186-3) as a guide in testing the requirement above, depending on the selection performed by the ST author. This will require that the evaluator have a trusted reference implementation of the algorithms that can produce test vectors that are verifiable during the test.

The evaluator shall ensure that the TSS contains a description of how the TSF complies with 800-56A and/or 800-56B, depending on the selections made. This description shall indicate the sections in 800-56A and/or 800-56B that are implemented by the TSF, and the evaluator shall ensure that key establishment is among those sections that the TSF claims to implement.

Any TOE-specific extensions, processing that is not included in the documents, or alternative implementations allowed by the documents that may impact the security requirements the TOE is to enforce shall be described.

6.1.2.2 FCS_CKM_EXT.4 Cryptographic Key Zeroization

FCS_CKM_EXT.4.1 The TSF shall zeroize all plaintext secret and private cryptographic keys and CSPs when no longer required.

PP Application Note: *"Cryptographic Critical Security Parameters" are defined in FIPS 140-2 as "security-related information (e.g., secret and private cryptographic keys, and authentication data such as passwords and PINs) whose disclosure or modification can compromise the security of a cryptographic module."*

The zeroization indicated above applies to each intermediate storage area for plaintext key/cryptographic critical security parameter (i.e., any storage,

such as memory buffers, that is included in the path of such data) upon the transfer of the key/cryptographic critical security parameter to another location.

Assurance Activity

The evaluator shall check to ensure the TSS describes each of the secret keys (keys used for symmetric encryption), private keys, and CSPs used to generate key; when they are zeroized (for example, immediately after use, on system shutdown, etc.); and the type of zeroization procedure that is performed (overwrite with zeros, overwrite three times with random pattern, etc.). If different types of memory are used to store the materials to be protected, the evaluator shall check to ensure that the TSS describes the zeroization procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are zeroized by overwriting once with zeros, while secret keys stored on the internal hard drive are zeroized by overwriting three times with a random pattern that is changed before each write").

6.1.2.3 FCS_COP.1(1) Cryptographic Operation (for data encryption/decryption)

FCS_COP.1.1(1)

The TSF shall perform encryption and decryption in accordance with a specified cryptographic algorithm AES operating in **CBC mode** and cryptographic key sizes 128-bits, 256-bits, and no other key sizes that meets the following:

1. FIPS PUB 197, “Advanced Encryption Standard (AES)”
2. NIST SP 800-38A

PP Application Note:

Application Note: For the assignment, the ST author should choose the mode or modes in which AES operates. For the first selection, the ST author should choose the key sizes that are supported by this functionality. For the second selection, the ST author should choose the standards that describe the modes specified in the assignment.

Assurance Activity:

The evaluator shall use tests appropriate to the modes selected in the above requirement from "The Advanced Encryption Standard Algorithm Validation Suite (AESAVS)", "The XTS-AES Validation System (XTSVS)", The CMAC Validation System (CMACVS)", "The Counter with Cipher Block Chaining-Message Authentication Code (CCM) Validation System (CCMVS)", and "The Galois/Counter Mode (GCM) and GMAC Validation System (GCMVS)" (these documents are available from http://csrc.nist.gov/groups/STM/cavp/index.html) as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test

6.1.2.4 FCS_COP.1(2) Cryptographic Operations (for cryptographic signature)

FCS_COP.1.1(2)

The TSF shall perform cryptographic signature services in accordance with a

a) RSA Digital Signature Algorithm (rDSA) with a key size (modulus) of 2048 bits or greater

that meets the following:

Case: RSA Digital Signature Algorithm

FIPS PUB 186-2 or FIPS PUB 186-3, “Digital Signature Standard”

PP Application Note: *Application Note: The ST Author should choose the algorithm implemented to perform digital signatures; if more than one algorithm is available, this requirement (and the corresponding FCS_CKM.1 requirement) should be iterated to specify the functionality. For the algorithm chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.*

For elliptic curve-based schemes, the key size refers to the log_2 of the order of the base point. As the preferred approach for digital signatures, ECDSA will be required in future publications of the NDPP.

Assurance Activity: The evaluator shall use the signature generation and signature verification portions of "The Digital Signature Algorithm Validation System” (DSA2VS), "The Elliptic Curve Digital Signature Algorithm Validation System” (ECDSA2VS), and "The RSA Validation System” (RSAVS (for 186-2) or RSA2VS (for 1806-3)) as a guide in testing the requirement above. The Validation System used shall comply with the conformance standard identified in the ST (i.e., FIPS PUB 186-2 or FIPS PUB 186-3). This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.1.2.5 FCS_COP.1(3) Cryptographic Operation (for cryptographic hashing)

FCS_COP.1.1(3) The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm SHA-1 and SHA-512 and message digest sizes 160 and 512 bits that meet the following: FIPS Pub 180-3, “Secure Hash Standard.”

PP Application Note: *The selection of the hashing algorithm must correspond to the selection of the message digest size; for example, if SHA-1 is chosen, then the only valid message digest size selection would be 160 bits.*

In subsequent publications of the NDPP, it is likely that SHA-1 will no longer be an approved algorithm for cryptographic hashing.

Assurance Activity: The evaluator shall use "The Secure Hash Algorithm Validation System (SHAVS)" as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.1.2.6 FCS_COP.1(4) Cryptographic Operation (for keyed hash message authentication)

FCS_COP.1.1(4) The TSF shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC-SHA-1 key size **160** and message digest sizes 160 bits that meet the following: FIPS Pub 198-1, "The Keyed-Hash Message Authentication Code, and FIPS Pub 180-3, “Secure Hash Standard.”

PP Application Note: *In future versions of the NDPP, SHA-1 may be removed as a valid hash algorithm. Developers are encouraged to transition to the other listed hash algorithms.*

Assurance Activity: The evaluator shall use "The Keyed-Hash Message Authentication Code (HMAC) Validation System (HMACVS)" as a guide in testing the requirement above. This will require that the evaluator have a reference implementation of the algorithms known to be good that can produce test vectors that are verifiable during the test.

6.1.2.7 FCS_SSH_EXT.1 SSH

FCS_SSH_EXT.1.1 The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and no other RFCs.

PP Application Note: *The ST author selects which of the additional RFCs to which conformance is being claimed. Note that these need to be consistent with selections in later elements of this component (e.g., cryptographic algorithms permitted).*

In the next version of the NDPP, a requirement will be added regarding rekeying. The requirement will read "The TSF shall ensure that the SSH connection be rekeyed after no more than 2^{28} packets have been transmitted using that key."

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.

Assurance Activity: The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication, that this list conforms to FCS_SSH_EXT.1.5, and ensure that password-based authentication methods are also allowed. The evaluator shall also perform the following tests:

1. Test 1: The evaluator shall, for each public key algorithm supported, show that the TOE supports the use of that public key algorithm to authenticate a user connection. Any configuration activities required to support this test shall be performed according to instructions in the operational guidance.
2. Test 2: Using the operational guidance, the evaluator shall configure the TOE to accept password-based authentication, and demonstrate that a user can be successfully authenticated to the TOE over SSH using a password as an authenticator.

FCS_SSH_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than 262144 bytes in an SSH transport connection are dropped.

PP Application Note: *RFC 4253 provides for the acceptance of "large packets" with the caveat that packets should be of "reasonable length" or dropped. The assignment should be filled in by the ST author with the maximum packet size accepted, thus defining "reasonable length" for the TOE.*

Assurance Activity: The evaluator shall check that the TSS describes how “large packets” in terms of RFC 4253 are detected and handled. The evaluator shall also perform the following test:

1. Test 1: The evaluator shall demonstrate that if the TOE receives a packet larger than that specified in this component, that packet is 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.

PP Application Note: *In the assignment, the ST author can select the AES-GCM algorithms, or "no other algorithms" if AES-GCM is not supported. If AES-GCM is selected, there should be corresponding FCS_COP entries in the ST. Since the Dec. 2010 publication of NDPP v1.0, there has been consider progress with respect to the prevalence of AES-GCM support in commercial network devices. It is likely that an NDPP v2.0 will be published in late 2012 which will require AES-GCM and AES-CBC will become optional.*

Assurance Activity: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics are specified, and the encryption algorithms supported are specified as well. The evaluator shall check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component. The evaluator shall also check the operational guidance to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements). The evaluator shall also perform the following test:

1. Test 1: The evaluator shall establish a SSH connection using each of the encryption algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

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).

PP Application Note: *RFCs 4253 and 5656 specify required and allowable public key algorithms. This requirement makes SSH-RSA “required” and allows the others to be claimed in the ST. The ST author should make the appropriate selection, selecting "no other public key algorithms" if only SSH_RSA is implemented.*

Assurance Activity: The assurance activity associated with FCS_SSH_EXT.1.4 verifies this requirement.

FCS_SSH_EXT.1.6 The TSF shall ensure that data integrity algorithms used in SSH transport connection is hmac-sha1.

PP Application Note: *RFC 6668 specifies the use of the sha2 algorithms in SSH.*

Assurance Activity: The evaluator shall check the TSS to ensure that it lists the supported data integrity algorithms, and that that list corresponds to the list in this

component. The evaluator shall also check the operational guidance to ensure that it contains instructions to the administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the "none" MAC algorithm is not allowed). The evaluator shall also perform the following test:

1. Test 1: The evaluator shall establish a SSH connection using each of the integrity algorithms specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

FCS_SSH_EXT.1.7 The TSF shall ensure that diffie-hellman-group14-sha1 and no other methods are the only allowed key exchange method used for the SSH protocol.

Assurance Activity: The evaluator shall ensure that operational guidance contains configuration information that will allow the security administrator to configure the TOE so that all key exchanges for SSH are performed using DH group 14 and any groups specified from the selection in the ST. If this capability is "hard-coded" into the TOE, the evaluator shall check the TSS to ensure that this is stated in the discussion of the SSH protocol. The evaluator shall also perform the following test:

1. Test 1: The evaluator shall attempt to perform a diffie-hellman-group1-sha1 key exchange, and observe that the attempt fails. For each allowed key exchange method, the evaluator shall then attempt to perform a key exchange using that method, and observe that the attempt succeeds.

6.1.2.8 FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

FCS_RBG_EXT.1.1 The TSF shall perform all random bit generation (RBG) services in accordance with FIPS Pub 140-2 Annex C: X9.31 Appendix 2.4 using 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 security strength of the keys and hashes that it will generate.

PP Application Note: *NIST Special Pub 800-90B describes the minimum entropy measurement that will probably be required future versions of FIPS-140. If possible this should be used immediately and will be required in future versions of the NDPP.*

For the first selection in FCS_RBG_EXT.1.1, the ST author should select the standard to which the RBG services comply (either 800-90B or 140-2 Annex C).

SP 800-90B contains four different methods of generating random numbers; each of these, in turn, depends on underlying cryptographic primitives (hash functions/ciphers). The ST author will select the function used (if 800-90B is selected), and include the specific underlying cryptographic primitives used

in the requirement or in the TSS. While any of the identified hash functions (SHA-1, SHA-224, SHA-256, SHA-384, SHA-512) are allowed for Hash_DRBG or HMAC_DRBG, only AES-based implementations for CT_DRBG are allowed. While any of the curves defined in 800-90B are allowed for Dual_EC_DRBG, the ST author not only must include the curve chosen, but also the hash algorithm used.

For the second selection in FCS_RBG_EXT.1.1, the ST author indicates whether the sources of entropy are software-based, hardware-based, or both. If there are multiple sources of entropy, the ST will elaborate each entropy sources and whether it is hardware- or software-based. Hardware-based noise sources are preferred.

Note that for FIPS Pub 140-2 Annex C, currently only the method described in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3 is valid. If the key length for the AES implementation used here is different than that used to encrypt the user data, then FCS_COP.1 may have to be adjusted or iterated to reflect the different key length. For the selection in FCS_RBG_EXT.1.2, the ST author selects the minimum number of bits of entropy that is used to seed the RBG.

The ST author also ensures that any underlying functions are included in the baseline requirements for the TOE.

For the selection in FCS_RBG_EXT.1.2, the ST author selects the appropriate number of bits of entropy that corresponds to the greatest security strength of the algorithms included in the ST. Security strength is defined in Tables 2 and 3 of NIST SP 800-57A. For example, if the implementation includes 2048-bit RSA (security strength of 112 bits), AES 128 (security strength 128 bits), and HMAC-512 (security strength 256 bits), then the ST author would select 256 bits.

Assurance Activity:

Documentation shall be produced – and the evaluator shall perform the activities – in accordance with Annex D, Entropy Documentation and Assessment.

The evaluator shall also perform the following tests, depending on the standard to which the RBG conforms.

Implementations Conforming to FIPS 140-2, Annex C

The reference for the tests contained in this section is The Random Number Generator Validation System (RNGVS) [RNGVS]. The evaluator shall conduct the following two tests. Note that the "expected values" are produced by a reference implementation of the algorithm that is known to be correct. Proof of correctness is left to each Scheme.

The evaluator shall perform a Variable Seed Test. The evaluator shall provide a set of 128 (Seed, DT) pairs to the TSF RBG function, each 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant for all 128 (Seed, DT) pairs. The DT value is incremented by 1 for each set. The seed values shall have no repeats within

the set. The evaluator ensures that the values returned by the TSF match the expected values.

The evaluator shall perform a Monte Carlo Test. For this test, they supply an initial Seed and DT value to the TSF RBG function; each of these is 128 bits. The evaluator shall also provide a key (of the length appropriate to the AES algorithm) that is constant throughout the test. The evaluator then invokes the TSF RBG 10,000 times, with the DT value being incremented by 1 on each iteration, and the new seed for the subsequent iteration produced as specified in NIST-Recommended Random Number Generator Based on ANSI X9.31 Appendix A.2.4 Using the 3-Key Triple DES and AES Algorithms, Section 3. The evaluator ensures that the 10,000th value produced matches the expected value.

Implementations Conforming to NIST Special Publication 800-90

The evaluator shall perform 15 trials for the RBG implementation. If the RBG is configurable, the evaluator shall perform 15 trials for each configuration. The evaluator shall also confirm that the operational guidance contains appropriate instructions for configuring the RBG functionality.

If the RBG has prediction resistance enabled, each trial consists of (1) instantiate drbg, (2) generate the first block of random bits, (3) generate a second block of random bits, (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP 800-90).

If the RBG does not have prediction resistance, each trial consists of (1) instantiate drbg, (2) generate the first block of random bits, (3) reseed, (4) generate a second block of random bits, (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator.

1. Entropy input: the length of the entropy input value must equal the seed length.

2. Nonce: If a nonce is supported (CTR_DRBG with no df does not use a nonce), the nonce bit length is one-half the seed length.

3. Personalization string: The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

4. Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

6.1.3 User Data Protection (FDP)

6.1.3.1 FDP_RIP.2 Full Residual Information Protection

FDP_RIP.2.1 The TSF shall ensure that any previous information content of a resource is made unavailable upon the allocation of the resource to all objects.

Assurance Activity: "Resources" in the context of this requirement are network packets being sent through (as opposed to "to", as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet. The evaluator shall check to ensure that the TSS describes packet processing to the extent that they can determine that no data will be reused when processing network packets. The evaluator shall ensure that this description at a minimum describes how the previous data are zeroized/overwritten, and at what point in the buffer processing this occurs.

6.1.4 Identification and Authentication (FIA)

6.1.4.1 FIA_PMG_EXT.1 Password Management

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: "!", "@", "$", "%", "^", "&", "*", "#", **"-", "_", "=", "+", "\", ";", ":", "'", "'", "<", ">", ",", ".", "?", "/", "`", "~"**

2. Minimum password length shall settable by the Security Administrator, and support passwords of 15 characters or greater;

PP Application Note: *The ST author selects the special characters that are supported by TOE; they may optionally list additional special characters supported using the*

assignment. "Administrative passwords" refers to passwords used by administrators at the local console or over protocols that support passwords, such as SSH and HTTPS.

Assurance Activity: The evaluator shall examine the operational guidance to determine that it provides guidance to security administrators on the composition of strong passwords, and that it provides instructions on setting the minimum password length. The evaluator shall also perform the following tests. Note that one or more of these tests can be performed with a single test case.

1. Test 1: The evaluator shall compose passwords that either meet the requirements, or fail to meet the requirements, in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that all characters, rule characteristics, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.

6.1.4.2 FIA_UIA_EXT.1 User Identification and Authentication

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

- Display the warning banner in accordance with FTA_TAB.1;
- ICMP replies

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.

PP Application Note: *This requirement applies to users (administrators and external IT entities) of services available from the TOE directly, and not services available by connecting through the TOE. While it should be the case that few or no services are available to external entities prior to identification and authentication, if there are some available (perhaps ICMP echo) these should be listed in the assignment statement; otherwise "no other actions" should be selected.*

Authentication can be password-based through the local console or through a protocol that supports passwords (such as SSH), or be certificate based (SSH, TLS).

For communications with external IT entities (e.g., an audit server or NTP server, for instance), such connections must be performed in accordance with FTP_ITC.1, whose protocols perform identification and authentication. This means that such communications (e.g., establishing the IPsec connection to the authentication server) would not have to be specified in the assignment, since establishing the connection "counts" as initiating the identification and authentication process.

Assurance Activity: The evaluator shall examine the TSS to determine that it describes the logon process for each logon method (local, remote (HTTPS, SSH, etc.)) supported for the product. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a "successful logon". The evaluator shall examine the operational guidance to determine that any necessary preparatory steps (e.g., establishing credential material such as pre-shared keys, tunnels, certificates, etc.) to logging in are described. For each supported the login method, the evaluator shall ensure the operational guidance provides clear instructions for successfully logging on. If configuration is necessary to ensure the services provided before login are limited, the evaluator shall determine that the operational guidance provides sufficient instruction on limiting the allowed services.

The evaluator shall perform the following tests for each method by which administrators access the TOE (local and remote), as well as for each type of credential supported by the login method:

1. Test 1: The evaluator shall use the operational guidance to configure the appropriate credential supported for the login method. For that credential/login method, the evaluator shall show that providing correct I&A information results in the ability to access the system, while providing incorrect information results in denial of access.
2. Test 2: The evaluator shall configure the services allowed (if any) according to the operational guidance, and then determine the services available to an external remote entity. The evaluator shall determine that the list of services available is limited to those specified in the requirement.
3. Test 3: For local access, the evaluator shall determine what services are available to a local administrator prior to logging in, and make sure this list is consistent with the requirement.

6.1.4.3 FIA_UAU_EXT.2 Password-based Authentication Mechanism

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

Assurance Activity: Assurance activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

6.1.4.4 FIA_UAU.7 Protected Authentication Feedback

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

PP Application Note: *"Obscured feedback" implies the TSF does not produce a visible display of any authentication data entered by a user (such as the echoing of a password), although an obscured indication of progress may be provided (such as an asterisk for each character). It also implies that the TSF does not*

return any information during the authentication process to the user that may provide any indication of the authentication data.

Assurance Activity: The evaluator shall perform the following test for each method of local login allowed:

1. Test 1: The evaluator shall locally authenticate to the TOE. While making this attempt, the evaluator shall verify that at most, obscured feedback is provided while entering the authentication information.

6.1.5 Security Management (FMT)

6.1.5.1 FMT_MTD.1 Management of TSF Data (for general TSF data)

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

PP Application Note: *The word "manage" includes but is not limited to create, initialize, view, change default, modify, delete, clear, and append. This requirement is intended to be the "default" requirement for management of TSF data; other iterations of FMT_MTD should place different restrictions or operations available on the specifically-identified TSF data. TSF data includes cryptographic information as well; managing these data would include the association of a cryptographic protocol with an interface, for instance.*

Assurance Activity: The evaluator shall review the operational guidance to determine that each of the TSF-data-manipulating functions implemented in response to the requirements of the NDPP is identified, and that configuration information is provided to ensure that only administrators have access to the functions. The evaluator shall examine the TSS to determine that, for each administrative function identified in the operational guidance; those that are accessible through an interface prior to administrator log-in are identified. For each of these functions, the evaluator shall also confirm that the TSS details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.

6.1.5.2 FMT_SMF.1 Specification of Management Functions

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

1. Ability to administer the TOE locally and remotely;
2. Ability to update the TOE, and to verify the updates using published hash capability prior to installing those updates;
3. Ability to configure the cryptographic functionality;

PP Application Note: *The TOE must provide functionality for both local and remote administration, as well as the capability for the administrator to verify that updates received came from a trusted source. They must be capable of performing this action using digital signatures, and optionally a published hash. The ST author chooses whether the published hash verification option is available using the first selection, which must match the corresponding*

selection in FPT_TUD_EXT.1.3. If the TOE offers the ability for the administrator to configure the services available prior to identification or authentication, or if any of the cryptographic functionality on the TOE can be configured, then the ST author makes the appropriate choice or choices in the second selection, otherwise select "no other capabilities."

Assurance Activity: The security management functions for FMT_SMF.1 are distributed throughout the PP and are included as part of the requirements in FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

6.1.5.3 FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

1. 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

1. Authorized Administrator role shall be able to administer the TOE locally;
2. Authorized Administrator role shall be able to administer the TOE remotely;

are satisfied.

PP Application Note: *FMT_SMR.2.2 requires that user accounts be associated with only one role. However, note that multiple users may have the same role, and the TOE is not required to restrict roles to a single person.*

FMT_SMR.2.3 requires that an authorized administrator be able to administer the TOE through the local console and through a remote mechanism (IPsec, SSH, TLS, TLS/HTTPS). For multiple component TOEs, only the TOE components providing the management control and configuration of the other TOE components require a local administration interface.

Assurance Activity: The evaluator shall review the operational guidance to ensure that it contains instructions for administering the TOE both locally and remotely, including any configuration that needs to be performed on the client for remote administration. In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of the NDPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

6.1.6 Protection of the TSF (FPT)

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

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

PP Application Note: *The intent of the requirement is that an administrator is unable to read or view the identified keys (stored or ephemeral) through "normal" interfaces. While it is understood that the administrator could directly read memory to view these keys, do so is not a trivial task and may require substantial work on the part of an administrator. Since the administrator is considered a trusted agent, it is assumed they would not endeavor in such an activity.*

Assurance Activity: The evaluator shall examine the TSS to determine that it details how any pre-shared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

6.1.6.2 FPT_APW_EXT.1 Protection of Administrator Passwords

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

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

PP Application Note: *The intent of the requirement is that raw password authentication data are not stored in the clear, and that no user or administrator is able to read the plaintext password through "normal" interfaces. An all-powerful administrator of course could directly read memory to capture a password but is trusted not to do so.*

In this version of the PP there are no requirements on the method used to store the passwords in non-plaintext form, but cryptographic methods based on the requirements in FCS_COP are preferred. In future versions of the NDPP, FCS_COP-based cryptographic methods that conform to the Level 2 Credential Storage requirements from NIST SP 800-63 will be required.

Assurance Activity: The evaluator shall examine the TSS to determine that it details all authentication data that are subject to this requirement, and the method used to obscure the plaintext password data when stored. The TSS shall also detail passwords are stored in such a way that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note.

6.1.6.3 FPT_STM.1 Reliable Time Stamps

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

Assurance Activity: The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time. The TSS provides a description of how the time is maintained and considered reliable in the context of each of the time related functions.

The evaluator examines the operational guidance to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the operational guidance instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

1. Test 1: The evaluator uses the operational guide to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

2. Test2: [conditional] If the TOE supports the use of an NTP server; the evaluator shall use the operational guidance to configure the NTP client on the TOE, and set up a communication path with the NTP server. The evaluator will observe that the NTP server has set the time to what is expected. If the TOE supports multiple protocols for establishing a connection with the NTP server, the evaluator shall perform this test using each supported protocol claimed in the operational guidance.

6.1.6.4 FPT_TUD_EXT.1 Trusted Update

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 published hash prior to installing those updates.

PP Application Note: *The digital signature mechanism referenced in the third element is the one specified in FCS_COP.1(2). The published hash referenced is generated by one of the functions specified in FCS_COP.1(3). The ST author should choose the mechanism implemented by the TOE; it is acceptable to implement both mechanisms.*

Assurance Activity: Updates to the TOE either have a hash associated with them, or are signed by an authorized source. If digital signatures are used, the definition of an authorized source is contained in the TSS, along with a description of how the certificates used by the update verification mechanism are contained on the device. The evaluator ensures this information is contained in the TSS. The evaluator also ensures that the TSS (or the operational guidance) describes how the candidate updates are obtained; the processing associated with verifying the digital signature or calculating the hash of the updates; and the actions that take place for successful (hash or signature was verified) and unsuccessful (hash or signature could not be verified) cases. The evaluator shall perform the following tests:

1. Test 1: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains a legitimate update using procedures described in the operational guidance and verifies that it is successfully installed on the TOE.

Then, the evaluator performs a subset of other assurance activity tests to demonstrate that the update functions as expected. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update.

2. Test 2: The evaluator performs the version verification activity to determine the current version of the product. The evaluator obtains or produces an illegitimate update, and attempts to install it on the TOE. The evaluator verifies that the TOE rejects the update.

6.1.6.5 FPT_TST_EXT.1 TSF Testing

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.

Assurance Activity: The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF on start-up; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

The evaluator shall also ensure that the operational guidance describes the possible errors that may result from such tests, and actions the administrator should take in response; these possible errors shall correspond to those described in the TSS.

6.1.7 TOE Access (FTA)

6.1.7.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

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

a) terminate the session

after a Security Administrator-specified time period of inactivity.

Assurance Activity: The evaluator shall perform the following test:

1. Test 1: The evaluator follows the operational guidance to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a local interactive session with the TOE. The evaluator then observes that the session is either locked or terminated after the configured time period. If locking was selected from the component, the evaluator then ensures that re-authentication is needed when trying to unlock the session.

6.1.7.2 FTA_SSL.3 TSF-initiated Termination

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

Assurance Activity: The evaluator shall perform the following test:

1. Test 1: The evaluator follows the operational guidance to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a remote interactive session with the TOE. The evaluator then observes that the session is terminated after the configured time period.

6.1.7.3 FTA_SSL.4 User-initiated Termination

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

Assurance Activity: The evaluator shall perform the following test:

1. Test 1: The evaluator initiates an interactive local session with the TOE. The evaluator then follows the operational guidance to exit or log off the session and observes that the session has been terminated.
2. Test 2: The evaluator initiates an interactive remote session with the TOE. The evaluator then follows the operational guidance to exit or log off the session and observes that the session has been terminated.

6.1.7.4 FTA_TAB.1 Default TOE Access Banners

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

PP Application Note: *This requirement is intended to apply to interactive sessions between a human user and a TOE. IT entities establishing connections or programmatic connections (e.g., remote procedure calls over a network) are not required to be covered by this requirement.*

Assurance Activity: The evaluator shall check the TSS to ensure that it details each method of access (local and remote) available to the administrator (e.g., serial port, SSH, HTTPS). The evaluator shall also perform the following test:

1. Test 1: The evaluator follows the operational guidance to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

6.1.8 Trusted Path/Channels (FTP)

6.1.8.1 FTP_ITC.1 Inter-TSF-trusted channel

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

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

FTP_ ITC.1.3 The TSF shall initiate communication via the trusted channel for **syslog**.

PP Application Note: *The intent of the above requirement is to use a cryptographic protocol to protect external communications with authorized IT entities that the TOE interacts with to perform its functions. This is not, however, to be used to specify VPN Gateway functionality; a separate VPN Protection Profile should be used in these instances. Protection (by either (or both) of the IPsec or SSH protocols) is required at least for communications with the server that collects the audit information. If it communicates with an authentication server (e.g., RADIUS), then the ST author chooses "authentication server" in FTP_ITC.1.1 and this connection must be protected by either IPsec or SSH. If other authorized IT entities (e.g., NTP server) are protected, the ST author makes the appropriate assignments (for those entities) and selections (for the protocols that are used to protect those connections). After the ST author has made the selections, they are to select the detailed requirements in Annex C corresponding to their protocol selection to put in the ST. TLS or HTTPS can only be used if tunneled through IPsec or SSH. To summarize, the connection to an external audit collection server is required to be protected by either IPsec or SSH protocols. If an external authentication server is supported, then it is required to protect that connection with either IPsec or SSH. For any other external server, external communications are not required to be protected, but if protection is claimed, then it must be protected with either IPsec or SSH.*

While there are no requirements on the party initiating the communication, the ST author lists in the assignment for FTP_ITC.1.3 the services for which the TOE can initiate the communication with the authorized IT entity.

The requirement implies that not only are communications protected when they are initially established, but also on resumption after an outage. It may be the case that some part of the TOE setup involves manually setting up tunnels to protect other communication, and if after an outage the TOE attempts to re-establish the communication automatically with (the necessary) manual intervention, there may be a window created where an attacker might be able to gain critical information or compromise a connection.

Assurance Activity: The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST. The evaluator shall confirm that the operational guidance contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken. The evaluator shall also perform the following tests:

1. Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.
2. Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the operational guidance to ensure that in fact the communication channel can be initiated from the TOE.
3. Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data are not sent in plaintext.
4. Test 4: The evaluators shall, for each protocol associated with each authorized IT entity tested during test 1, the connection is physically interrupted. The evaluator shall ensure that when physical connectivity is restored, communications are appropriately protected.

Further assurance activities are associated with the specific protocols.

6.1.8.2 FTP_TRP.1 Trusted Path

FTP_TRP.1.1 The TSF shall use SSH and no other products 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 administration actions.

PP Application Note: *This requirement ensures that authorized remote administrators initiate all communication with the TOE via a trusted path, and that all communications with the TOE by remote administrators is performed over this path. The data passed in this trusted communication channel are encrypted as defined the protocol chosen in the first selection. The ST author chooses the mechanism or mechanisms supported by the TOE, and then*

ensures the detailed requirements in Annex C corresponding to their selection are copied to the ST if not already present.

Assurance Activity: The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The trusted path is required to be implemented with either IPsec or SSH; TLS/HTTPS is only allowed if tunneled through one of the other two protocols. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST. The evaluator shall confirm that the operational guidance contains instructions for establishing the remote administrative sessions for each supported method. The evaluator shall also perform the following tests:

1. Test 1: The evaluators shall ensure that communications using each specified (in the operational guidance) remote administration method is tested during the course of the evaluation, setting up the connections as described in the operational guidance and ensuring that communication is successful.
2. Test 2: For each method of remote administration supported, the evaluator shall follow the operational guidance to ensure that there is no available interface that can be used by a remote user to establish a remote administrative sessions without invoking the trusted path.
3. Test 3: The evaluator shall ensure, for each method of remote administration, the channel data are not sent in plaintext.

Further assurance activities are associated with the specific protocols.

6.2 Security Assurance Requirements

This Security Target is conformant with the assurance requirements specified in the NDPP. The CC Part 3 conformant security assurance requirements are listed in Table 10. The CC Part 3 extended assurance requirements are listed in Section 6.1 as “Assurance Activity” and Section 6.2.1.

Table 10: Assurance Requirements

Assurance Class	Assurance Component	Assurance Components Description
Development	ADV_FSP.1	Basic functional specification
Guidance Documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative user guidance
Life-cycle Support	ALC_CMC.1	Labeling of the TOE
	ALC_CMS.1	TOE CM coverage
Tests	ATE_IND.1	Independent testing - conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability analysis

6.2.1 Extended Security Assurance Requirements

These requirements are taken directly from the NDPP and augment or modify the existing SARs taken from CC Part 3.

6.2.1.1 ADV_FSP.1

There are no specific assurance activities associated with these SARs. The functional specification documentation is provided to support the evaluation activities described in Section 6.1, and other activities described for AGD, ATE, and AVA SARs. The requirements on the content of the functional specification information is implicitly assessed by virtue of the other assurance activities being performed; if the evaluator is unable to perform an activity because the there is insufficient interface information, then an adequate functional specification has not been provided.

6.2.1.2 AGD_OPE.1

Some of the contents of the operational guidance will be verified by the assurance activities in Section 6.1 and evaluation of the TOE according to the CEM. The following additional information is also required.

The operational guidance shall at a minimum list the processes running (or that could run) on the TOE in its evaluated configuration during its operation that are capable of processing data received on the network interfaces (there are likely more than one of these, and this is not limited to the process that "listens" on the network interface). It is acceptable to list all processes running (or that could run) on the TOE in its evaluated configuration instead of attempting to determine just those that process the network data. For each process listed, the administrative guidance will contain a short (e.g., one- or two-line) description of the process' function, and the privilege with which the service runs. "Privilege" includes the hardware privilege level (e.g., ring 0, ring 1), any software privileges specifically associated with the process, and the privileges associated with the user role the process runs as or under.

The operational guidance shall contain instructions for configuring the cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

The documentation must describe the process for verifying updates to the TOE, either by checking the hash or by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

1. For hashes, a description of where the hash for a given update can be obtained. For digital signatures, instructions for obtaining the certificate that will be used by the FCS_COP.1(2) mechanism to ensure that a signed update has been received from the certificate owner. This may be supplied with the product initially, or may be obtained by some other means.
2. Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).
3. Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes generation of the hash/digital signature.

The TOE will likely contain security functionality that does not fall in the scope of evaluation under the NDPP. The operational guidance shall make it clear to an administrator which security functionality is covered by the evaluation activities.

6.2.1.3 AGD_PRE.1

As indicated in the introduction above, there are significant expectations with respect to the documentation, especially when configuring the operational environment to support TOE functional requirements. The evaluator shall check to ensure that the guidance provided for the TOE adequately addresses all platforms claimed for the TOE in the ST.

6.2.1.4 ATE_IND.1

The evaluator shall prepare a test plan and report documenting the testing aspects of the system. The test plan covers all of the testing actions contained in the CEM and the body of this PP's Assurance Activities. While it is not necessary to have one test case per test listed in an Assurance Activity, the evaluator must document in the test plan that each applicable testing requirement in the ST is covered.

The test plan identifies the platforms to be tested, and for those platforms not included in the test plan but included in the ST, the test plan provides a justification for not testing the platforms. This justification must address the differences between the tested platforms and the untested platforms, and make an argument that the differences do not affect the testing to be performed. It is not sufficient to merely assert that the differences have no affect; rationale must be provided. If all platforms claimed in the ST are tested, then no rationale is necessary.

The test plan describes the composition of each platform to be tested, and any setup that is necessary beyond what is contained in the AGD documentation. It should be noted that the evaluator is expected to follow the AGD documentation for installation and setup of each platform either as part of a test or as a standard pre-test condition. This may include special test drivers or tools. For each driver or tool, an argument (not just an assertion) should be provided that the driver or tool will not adversely affect the performance of the functionality by the TOE and its platform. This also includes the configuration of the cryptographic engine to be used. The cryptographic algorithms implemented by this engine are those specified by the NDPP and used by the cryptographic protocols being evaluated (IPsec, TLS/HTTPS, SSH).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives. These procedures include expected results. The test report (which could just be an annotated version of the test plan) details the activities that took place when the test procedures were executed, and includes the actual results of the tests. This shall be a cumulative account, so if there was a test run that resulted in a failure, a fix installed, and then a successful re-run of the tests, the report would show a "fail" and "pass" result (and the supporting details), and not just the "pass" result.

6.2.1.5 AVA_VAN.1

As with ATE_IND, the evaluator shall generate a report to document their findings with respect to this requirement. This report could physically be part of the overall test report mentioned in ATE_IND, or a separate document. The evaluator performs a search of public information to determine the vulnerabilities that have been found in network infrastructure devices and the implemented communication protocols in general, as well as those that pertain to the particular TOE. The evaluator documents the sources consulted and the vulnerabilities found in the report. For each vulnerability found, the evaluator either provides a rationale with respect to its non-applicability, or the evaluator formulates a test (using the guidelines provided in ATE_IND) to confirm the vulnerability, if suitable. Suitability is determined by assessing the attack vector needed to take advantage of the vulnerability. For example, if the vulnerability can be detected by pressing a key combination on boot-up, a test would be suitable at the assurance level of the NDPP. If exploiting the vulnerability requires expert skills and an electron microscope, for instance, then a test would not be suitable and an appropriate justification would be formulated.

6.3 Security Requirements Rationale

6.3.1 Security Function Requirement to Security Objective Rationale

Table 11: SFR to Objective Mapping	
	Objective

#	SFR/SAR	O.PROTECTED_COMMUNICATIONS	O.VERIFIABLE_UPDATES	O.SYSTEM_MONITORING	O.DISPLAY_BANNER	O.TOE_ADMINISTRATION	O.REDIDUAL_INFORMATION_CLEARING	O.SESSION_LOCK	O.TSF_SELF_TEST
1.	FAU_GEN.1			X					
2.	FAU_GEN.2			X					
3.	FAU_STG_EXT.1			X					
4.	FCS_CKM.1	X							
5.	FCS_CKM_EXT.4	X							
6.	FCS_COP.1(1)	X							
7.	FCS_COP.1(2)	X	X						
8.	FCS_COP.1(3)	X	X						
9.	FCS_COP.1(4)	X							
10.	FCS_RGB_EXT.1	X							
11.	FDP_RIP.2						X		
12.	FIA_PMG_EXT.1					X			
13.	FIA_UIA_EXT.1					X			
14.	FIA_UAU_EXT.1								
15.	FIA_UAU.7					X			
16.	FMT_MTD.1					X			
17.	FMT_SMF.1					X			
18.	FMT_SMR.2					X			
19.	FPT_APW_EXT.1					X			
20.	FPT_SKP_EXT.1	X							
21.	FPT_STM.1			X					
22.	FPT_TUD_EXT.1		X						
23.	FPT_TST_EXT.1								X
24.	FTA_SSL_EXT.1							X	
25.	FTA_SSL.3							X	
26.	FTA_SSL.4								
27.	FTA_TAB.1				X				
28.	FTP_ITC.1	X							
29.	FTP_TRP.1	X							

The following sections present the rationales that demonstrate that the SFRs meet all security objectives for the TOE.

6.3.1.1 Protected Communications

O.PROTECTED_COMMUNICATIONS

To address the issues concerning transmitting sensitive data to and from the TOE described in Section 3.1, Table 4, row "T.UNAUTHORIZED_ACCESS", compliant TOEs will provide encryption for these communication paths between themselves and the endpoint. These channels are implemented using one (or more) of three standard protocols: IPsec, TLS/HTTPS, and SSH. These protocols are specified by RFCs that offer a variety of implementation choices. Requirements have been imposed on some of these choices (particularly those for cryptographic primitives) to provide interoperability and resistance to cryptographic attack. While compliant TOEs must support all of the choices specified in the ST, they may support additional algorithms and protocols. If such additional mechanisms are not evaluated, guidance must be given to the administrator to make clear the fact that they are not evaluated.

In addition to providing protection from disclosure (and detection of modification) for the communications, each of the protocols described in this document (IPsec, SSH, and TLS/HTTPS) offer two-way authentication of each endpoint in a cryptographically secure manner, meaning that even if there was a malicious attacker between the two endpoints, any attempt to represent themselves to either endpoint of the communications path as the other communicating party would be detected. The requirements on each protocol, in addition to the structure of the protocols themselves, provide protection against replay attacks such as those described in Section 3.1, Table 4, row "T.UNAUTHORIZED_ACCESS", usually by including a unique value in each communication so that replay of that communication can be detected.

(FCS_CKM.1, FCS_CKM_EXT.4, FCS_COP.1(1), FCS_COP.1(2), FCS_COP.1(3), FCS_COP.1(4), FCS_RBG_EXT.1, FPT_SKP_EXT.1, FTP_ITC.1, FTP_TRP.1, FCS_SSH_EXT.1)

6.3.1.2 Verifiable Updates

O.VERIFIABLE_UPDATES

As outlined in Section 3.1, Table 4, row "T.UNAUTHORIZED_UPDATE", failure by the Security Administrator to verify that updates to the system can be trusted may lead to compromise of the entire system. A first step in establishing trust in the update is to publish a hash of the update that can be verified by the System Administrator prior to installing the update. In this way, the Security Administrator can download the update, compute the hash, and compare it to the published hash. While this establishes that the update downloaded is the one associated with the published hash, it does not indicate if the source of the update/hash combination has been compromised or can't be trusted. So, there remains a threat to the system. To establish trust in the source of the updates, the system can provide cryptographic mechanisms and procedures to procure the update, check the update cryptographically through the TOE-provided digital signature mechanism, and install the update on the system. While there is no requirement that this process be completely automated, administrative guidance documentation will detail any procedures that must be performed manually, as well as the manner in which the administrator ensures that the signature on the update is valid.

(FPT_TUD_EXT.1, FCS_COP.1(2), FCS_COP.1(3))

6.3.1.3 System Monitoring

O.SYSTEM_MONITORING

In order to assure that information exists that allows Security Administrators to discover intentional and unintentional issues with the configuration and/or operation of the system as discussed in Section 3.1; Table 4; rows "T.ADMIN_ERROR", "T_UNDETECTED_ACTIONS", and "T.UNAUTHROIZED_ACCESS"; compliant TOEs have the capability of generating audit data targeted at detecting such activity. Auditing of administrative activities provides information that may hasten corrective action should the system be

configured incorrectly. Audit of select system events can provide an indication of failure of critical portions of the TOE (e.g., a cryptographic provider process not running) or anomalous activity (e.g., establishment of an administrative session at a suspicious time, repeated failures to establish sessions or authenticate to the system) of a suspicious nature.

In some instances there may be a large amount of audit information produced that could overwhelm the TOE or administrators in charge of reviewing the audit information. The TOE must be capable of sending audit information to an external trusted entity, which mitigates the possibility that the generated audit data will cause some kind of denial of service situation on the TOE. This information must carry reliable timestamps, which will help order the information when sent to the external device.

Loss of communication with the audit server is problematic. While there are several potential mitigations to this threat, the NDPP does not mandate that a specific action takes place; the degree to which this action preserves the audit information and still allows the TOE to meet its functionality responsibilities should drive decisions on the suitability of the TOE in a particular environment.

(FAU_GEN.1, FAU_GEN.2, FAU_STG_EXT.1, FPT_STM.1)

6.3.1.4 TOE Administration

O.TOE_ADMINISTRATION, O.SESSION_LOCK

In order to provide a trusted means for administrators to interact with the TOE, the TOE provides a password-based logon mechanism. The administrator must have the capability to compose a strong password, and have mechanisms in place so that the password must be changed regularly. To avoid attacks where an attacker might observe a password being typed by an administrator, passwords must be obscured during logon. Session locking or termination must also be implemented to mitigate the risk of an account being used illegitimately. Passwords must be stored in an obscured form, and there must be no interface provided for specifically reading the password or password file such that the passwords are displayed in plain text.

(FIA_UIA_EXT.1, FIA_PMG_EXT.1, FIA_UAU.7, FMT_MTD.1, FMT_SMF.1, FMT_SFR.1, FPT_APW_EXT.1, FTA_SSL_EXT.1, FTA_SSL.3)

O.DISPLAY_BANNER

(FTA_TAB.1)

6.3.1.5 Residual Information Clearing

O.RESIDUAL_INFORMATION_CLEARING

In order to counter the threat that user data is inadvertently included in network traffic not intended by the original sender, the TSF ensures that network packets sent from the TOE do not include data "left over" from the processing of previous network information.

(FDP_RIP.2)

6.3.1.6 TSF Self Test

O.TSF_SELF_TEST

In order to detect some number of failures of underlying security mechanisms used by the TSF, the TSF will perform self tests. The extent of this self testing is left to the product developer, but a more comprehensive set of self-tests should result in a more trustworthy platform on which to develop enterprise architecture.

(FPT_TST_EXT.1)

6.3.2 Security Functional Requirement Dependency Rationale

As discussed in Section 2, Conformance Claims, this PP claims strict conformance to the NDPP from which all SFRs are used. Therefore, all dependencies are filled.

6.3.3 Security Assurance Requirements Rationale

As discussed in Section 2, Conformance Claims, this PP claims strict conformance to the NDPP from which all SARs are used. Therefore, all dependencies are filled.

7 TOE Summary Specification

This section provides evaluators and potential consumers of the TOE with a high-level description of each SFR, thereby enabling them to gain a general understanding of how the TOE is implemented. These descriptions are intentionally not overly detailed, thereby disclosing no proprietary information. These sections refer to SFRs defined in Section 6, Security Requirements.

The TOE consists of the following Security Functions:

- Security Audit
- Cryptographic Operations
- User Data Protection
- Security Management
- Extended Requirements
- Protection of the TSF
- TOE Access
- Trusted Path/Channels

7.1 Security Audit

7.1.1 Audit Generation

When an individual system process creates an audit message, it will include the outcome of the event in the generated message. When the AAA module or CLI generates an audit message, it will add the subject identity, i.e. username, to the message whenever possible. Each possible audit event generated by the TOE is defined in Table 9. The syslog process formats and adds extra audit information including the time, date, system process, severity, and type of event to the audit event.

The table below contains the list of protocol failures that are auditable.

Table 12: Auditable Protocol Failures

Protocol	Auditable Failures
SSH	Communication failure • Received disconnect • Connection closed (FTP_TRP.1) Failure of the trusted path functions • Corrupted MAC on input (FCS_SSH_EXT.1) Failure to establish an SSH session • Unable to negotiate a key exchange method • Initial connection refused by the remote host • Session terminated with no password • Invalid credentials provided o Password o Public/private key pair

7.1.2 Audit Storage

Audit data is stored locally in volatile memory (DRAM). Using the Linux EOS, a separate 411MB RAM disk (tmpfs partition) is created at boot time and assigned to the /var directory as part of the global file system. Logs are stored underneath the subdirectory /var/log, where they are protected from unauthorized viewing by the filesystem role-based access control. The restrictive CLI also controls access

to the commands that allow viewing of these logs. A logrotate daemon is used to manage log size. The logrotate daemon allows system and EOS log files to grow to 10M in size. Once the logs have reached their configured capacity, they are compressed using gunzip compression provided by the Linux EOS. After four separate compressed files have been created for each log, the oldest compressed file is deleted to make room for the newest compressed file. When the device is rebooted all log files are deleted automatically because the volatile memory is cleared.

The TOE establishes a trusted channel between itself and the external audit server. A trusted channel is created when the TOE establishes an SSH session between itself and the external audit server with TCP port forwarding enabled. After the SSH session is established, the TOE is configured by the user to forward all messages received by the syslog process to the listening TCP port created by the SSH connection. This ensures that all audit traffic is encapsulated and hence protected by the SSH connection. The TOE, in the default configuration, uses the following CAVP certified algorithms for the SSH connection: RSA-2048 for public-key authentication, AES-128/256 CBC for data encryption, HMAC-SHA1 for data integrity. The CLI command available to the user to initialize the SSH session acts as a wrapper around the SSH executable and specifies the necessary options for these secure cryptographic algorithms to the underlying SSH executable. The SSH options and implementation together provide a trusted channel that protects against audit data disclosure and modification.

7.2 Cryptographic Operations

7.2.1 SSH

The TOE uses OpenSSL 1.0.0e-fips build for cryptographic operations and OpenSSH 5.5p1 which runs the SSH service. OpenSSH supports asymmetric key authentication as well as password-based authentication for administrators. In the default configuration, OpenSSH is configured to support the following algorithms:

- RSA-2048 for public-key authentication
- AES-128/256 CBC for data encryption
- HMAC-SHA1 for data integrity
- diffie-hellman-group14-sha1 for key exchange

The use of DH group 14 is enforced through the sshd_config configuration file for the SSH daemon and is not hard-coded.

In order to comply with RFC 4253 OpenSSH calculates the packet length of all received packets and simply discards packets larger than 256kb.

7.2.2 RFC Compliance

The TOE uses the SSH client and server provided by the OpenSSH implementation. Members from the OpenSSH development community participated in the definition of the RFCs and the OpenSSH organization claims to have implemented the RFCs on this website: http://www.openssh.org/specs.html. Per RFC 2119 “Key words for use in RFCs to Indicate Requirement Levels”, to claim that one implements an RFC, one must implement all MUST, REQUIRED, and SHALLs in the specification. Furthermore, one must not implement any MUST NOT and SHALL NOT. Arista's internal investigation has looked at the MUST, MUST NOT, SHALL, SHALL NOT, REQUIRED, MAY, SHOULD, SHOULD NOT, RECOMMENDED, and NOT RECOMMENDED as specified in RFCs 4251, 4252, 4253, and 4254 and determined that OpenSSH has followed the RFCs and Arista did not find any

exceptions other than those documented by the OpenSSH community. This document is included in Annex A of the ST.

7.2.3 NIST 800-56A/B Compliance

The TOE complies with both the 800-56A and 800-56B standards. Arista has done extensive code review, online research, and correspondence with the OpenSSL Development Team. The table below details the sections of 800-56A/B that the TOE complies to.

Table 13: NIST 800-56A/B Compliance

Section	Implementation Notes
800-56A • 1-5.5.1.1 • 5.5.2-5.6.1.1 • 5.6.2-5.6.2.4 • 5.6.3-5.7.1.1 • 7-9	• 5.7.2 is not supported due to MQV having patent issues and OpenSSH/OpenSSL choosing not to support it for that reason. • 5.8 and 6 are not supported due to the 800-56A standard being created after the SSH standard was finalized. However, per 800-135rev1 the SSH key-agreement protocols are permissible.
800-56B • 1-5.9 • 6-6.2.2 • 6.2.3 • 6.3.1 • 6.4.2.1 (option 1) • 6.5.1 (section 1-2) • 6.5.2 (section 6.5.2.2) • 7	• 5.9.1-5.9.2 is not used due to how the SSH protocol structures it's data • 6-6.2.2 uses PKCS#1 format which uses Chinese Remainder Theorem option of storage • 6.2.3 - Target security strength of 112 bits is used which requires keys of bit length 2048. • 6.3.1 - Generated in Chinese Remainder Theorem format • 7 - RSA-OAEP is used per RFC 4432 for the SSH Protocol

7.2.4 Secret Keys, Private Keys, and CSPs

Sections 7.2.4.1,7.2.4.2, and 7.2.4.3 contain descriptions of all secret keys, private keys, and CSPs used to generate keys.

7.2.4.1 Private SSH Key

The RSA-2048 private key used for SSH user authentication is stored in persistent flash memory until no longer required. It is generated by the device using the OpenSSL cryptographic library discussed below. When the key is no longer required, the administrator uses the CLI to replace the key. The CLI uses a secure erase tool in the background, which zeroizes the file by overwriting it with random data once before it generates a new key.

7.2.4.2 Passwords

The TOE stores the SHA-512 hash of passwords used for the following accounts: boot loader administrator, AAA users, and local administrator (root) account.

7.2.4.3 SSH and 800-90A CSPs

Each CSP used by OpenSSH is managed internally by the OpenSSL cryptographic library. They are listed in the table below. Each CSP name is generic, corresponding to API parameter data structures.

Table 14: OpenSSL Cryptographic CSPs and IVs

CSP Name	Description	Storage
RSA SGK	RSA (2048 bits) signature generation key	Volatile memory (RAM)

RSA KDK	RSA (2048 bits) key decryption (private key transport) key	Volatile memory (RAM)
AES EDK	AES (128/256) encrypt / decrypt key	Volatile memory (RAM)
AES CMAC	AES (128/256) CMAC generate / verify key	Volatile memory (RAM)
HMAC Key	Keyed hash key (160)	Volatile memory (RAM)
DH Private	DH (Diffie-Hellman) private agreement key	Volatile memory (RAM)
RNG CSPs	Seed (128 bit), AES 128/256 seed key and associated state variables for ANS X9.31 AES based RNG. Used to generate keys listed above and the SSH private key.	Volatile memory (RAM)

Each respective CSP internal to OpenSSL is zeroized when the process call exits. This is accomplished using an assembly routine that writes a single pass of zeros to the RAM memory addresses. In the case of OpenSSH, the process call exits when the SSH connection is closed and the service exits.

Each CSP is protected from unauthorized access through the operating system memory management that disallows any memory read requests from other processes. The keys are accessible only though the calling application, which can invoke API functions to export or create keys. Additionally, the API functions must, by design, be called in non-overlapping sequence to prevent API calls from executing concurrently and corrupting CSP values.

7.3 User Data Protection

When a packet of user data is received by the TOE, the hardware initially stores the data exactly as received. The TOE is able to determine the exact length (in octets) of each packet received due to the IEEE 802.3 specification that enforces preambles, interframe gaps, and Maximum Transmission Unit (MTU) values. As a packet is received by the TOE, memory segments are allocated from a global memory buffer. The packet is split and stored in these memory segments along with the count of the number of valid bytes in the segment and pointers to the head segment and next segment. The entire Ethernet packet is stored as a linked list of memory segment pointers, which allows the TOE to be aware of the exact byte length. When packets are read from memory, the number of valid bytes in the packet is used by the TOE to ensure that only valid packet data is read from memory and no data is reused.

When the switch is configured for cut-through operation, payload errors will lead to bad FCS being set and the frame discarded by endpoints. In store and forward operation, the packet will be discarded and nothing will be transmitted. Packet memory is not reused or returned to the free pool until the packet is fully transmitted.

7.4 Identification and Authentication

The login processes for both local console and remote SSH are the same and are detailed below.

Once a user initiates a connection to the administration interface they are prompted for a username and password After a user provides authentication credentials, the TOE invokes a PAM (Pluggable Authentication Modules) AAA plugin. The AAA plugin creates an internal connection to the AAA module provided by EOS and passes the authentication credentials to the module. The module checks the credentials against its database and then responds with a SUCCESS or DENIED message. Based on the response the AAA plugin then returns a SUCCESS or DENIED message. If the requesting program receives a DENIED message it prints an error message to the user and denies login. If the requesting program receives a SUCCESS message it allows the user to continue and begin issuing CLI commands. For each CLI command issued by the administrator, a request is made to the AAA module, which determines if the user has permission to execute the CLI command.

The TOE also enforces that the user provide a password for access to the boot loader shell, which allows limited configuration of the EOS image at boot time. A SHA-512 hash of the boot-loader password is stored in the flash memory, which also contains the EOS image. All authorization code is self-contained within the boot loader executable.

When RSA authentication is used with a remote SSH session, the SSH daemon performs the initial authentication verification locally, without going through AAA. Once the RSA key matches, the daemon still performs CLI authorization requests to the AAA module as detailed above.

7.5 Security Management

Locally stored password information is obscured with SHA-512 hashing. Traffic between the TOE and all authorized IT entities is encrypted over an SSHv2 connection using the AES-128/256-CBC algorithm. Authentication of the IT entity is performed using RSA-2048 asymmetric keys. The private RSA-2048 key is stored in plaintext but protected from reading by the administrative interface that restricts read access of any administrative users. The only method of bypassing the read protection of the private RSA key is by using the undocumented bash interface as the root account. Both the interface and the root account are disabled in the CC-evaluated configuration.

Software updates are verified using SHA-512 published hash values. Each of these cryptographic operations are performed by FIPS validated algorithms embedded within OpenSSL (see Cryptographic Operations).

The TOE does not allow any administrative functions before log-in.

7.6 Protection of the TSF

7.6.1 Pre-shared, Symmetric, and Private Key Storage Methods

This section details how pre-shared, symmetric, and private keys are stored to ensure that they cannot be read. Table 14 below lists the keys used by the TOE, and the TSF protection for each key.

Please see section 7.2.4 for further detail of all private keys, secret keys, and CSPs.

Table 15: Key Storage

Key Description	Key Type	Protection	Algorithm	Storage
Local Admin Password	Pre-shared	SHA-512 one-way hash	n/a	Local file system
AAA Module Users	Pre-shared	SHA-512 one-way hash	n/a	Local file system
Boot Loader Password	Pre-shared	SHA-512 one-way hash	n/a	Flash storage
Private SSH Key	Private Key	Plaintext with read control	RSA-2048	Local file system

Read access to all keys is managed through the AAA authentication mechanism and CLI which only allows authorized administrators the read the appropriate files. The CLI does not provide a documented command for any administrators to view any private keys. However, an administrator who enables a root account (against the CC-evaluated configuration) and then uses the bash prompt and the “cat” tool may circumvent the CLI to read the private RSA key.

7.6.2 Protection of Administrator Passwords

The TOE uses SHA-512 hash protection to ensure that any administrator passwords are unable to be read or stored in plaintext. When a user authenticates locally the system creates a hash of the entered password and compares it against the stored hash to ensure a plaintext password is never revealed.

7.6.3 Time

Several security functions make use of time, including the audit service rsyslog, SSH timeout during authentication, and CLI idle-timeout. Time is synchronized through an external NTP server. The CLI, only allows authorized administrators to modify the NTP settings.

7.6.4 Updates

Software updates for the TOE have a SHA-512 hash associated with them. The hash values are stored and published on the manufacturer's website that prevents modification. Candidate updates and published SHA hash values are obtained through the website by the customer. The customer must then use a trusted application, or the TOE, to calculate the SHA-512 hash value on the candidate update. The customer must verify manually that these hash values are identical. If the values are identical the customer may use the TOE's administrative interface to install the update. Otherwise the customer must retry the software update download or contact the manufacturer.

7.6.5 Self-Tests

The TOE performs a suite of self-tests during start-up.

Early in the boot process the TOE performs a memory test which writes two different patterns, 0xffffffffffffffff and 0x5555555555555555, to memory. Between each pass it also reads the memory and verifies no write errors have occurred. If an error has occurred the TOE marks the memory as unavailable and logs an error message containing the bad memory addresses.

Later in the boot process cryptographic self-tests are run which verifies the OpenSSL cryptographic algorithms are running correctly. The first self-test checks the software integrity of the library as a whole by calculating the HMAC-SHA1 value of the core binaries and comparing it against the value calculated at compile-time. All subsequent tests are designed to test each individual cryptographic algorithm supported by the OpenSSL library. Most algorithm tests provide known inputs to its respective cryptographic function, and compares the output of the function to the known correct answer, (stored at compile time). These tests, called Known Answer Tests, are run for the following cryptographic algorithms: HMAC-SHA1, HMAC-SHA512, AES-CBC, RSA, X9.31-RNG.

Combined, this suite of self-tests help ensure that the TOE is functioning as designed and without fault.

7.7 TOE Access

The table below contains each method of access available to the administrator.

Table 16: TOE Access Methods	
Applications	Interface
SSHv2* - Client (e.g. PuTTy, BSD ssh) OpenSSH 5.5p1 - Server	Ethernet administrative network interface
RS232 Console via serial cable	Serial interface

* Any compliant application capable of running this protocol or service.

7.8 Trusted Path/Channels

The two sections below contain the communication mechanisms and protocols used for each authorized IT entity supported by the TOE.

7.8.1 IT Entity - Audit Server

Table 17: Audit Protocols and Communication Mechanisms		
Applications	Allowed Communication Protocols	Communication Mechanism
EOS rsyslog - Client Syslog* - Server	Layer 7 - syslog - RFC 5424 Layer 5/6 - Plaintext socket Layer 4 - TCP Layer 3- IP	TCP port forwarding over SSHv2

* Any compliant application capable of running this protocol or service.

7.8.2 Remote TOE Administration Methods

When a user connects to the TOE via SSHv2 they are initiating an instance of the CLI which provides administration of the TOE. All CLI traffic is transmitted over the SSHv2 connection detailed below.

Table 18: Remote TOE Administration Methods	
Applications	Communication Protections
SSHv2* - Client OpenSSH 5.5p1 - Server	Authentication - RSA 2048 asymmetric key pair Encryption - AES 128/256 CBC (diffie-hellman-group14-sha1 for key exchange) Data Integrity - HMAC-SHA1

8 Terms and Definitions

Table 19: TOE Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
AAA	Authentication Authorization and Accounting
ACL	Access Control List
CLI	Command Line Interface
CPU	Central Processing Unit
CSP	Critical Security Parameter
ECN	Electronic Communication Networks
HPC	High Performance Computing
KAT	Known Answer Tests
NAT	Network Address Translation
OSI	Open Systems Interconnection
PAM	Pluggable Authentication Modules
PCT	Pairwise Consistency Test
PC	Personal Computer
QoS	Quality of Service
QSFP	Quad Small Form-factor Pluggable
SFP	Small Form-factor Pluggable
SSH	Secure Shell
STP	Spanning Tree Protocol
VLAN	Virtual Local Area Network

Table 20: CC Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
CAC	Common Access Card
CAP	Composed Assurance Package
CC	Common Criteria
CCRA	Arrangement on the Recognition of Common Criteria Certificates in the field of IT Security
DAC	Discretionary Access Control
DOD	Department of Defense
EAL	Evaluation Assurance Level
IT	Information Technology
OSP	Organizational Security Policy
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SFP	Security Function Policy
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface

9 References

Table 21: TOE Guidance Documentation		
Reference	Description	Date
[1]	Common Criteria Guidance Supplement Arista 7150, 7050X, 7250X, 7300X and 7500E Series Switches Guidance Document AGD_OPE.1, AGD_PRE.1	June 10, 2014
[2]	User Manual Arista Networks, Arista EOS Version 4.13.3.4	June 9, 2014
[3]	Arista EOS System Message Guide, Software Release 4.13.3.4	April 18, 2014
[4]	Annex D: Entropy Documentation	N/A
[5]	Arista Quick Start Guide 7000 Series 1 RU – Gen 2Data Center Switches	2014

Table 22: Common Criteria v3.1 References			
Reference	Description	Version	Date
[6]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model CCMB-2009-07-001	V3.1 R3	July 2009
[7]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components CCMB-2009-07-002	V3.1 R3	July 2009
[8]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components CCMB-2009-07-003	V3.1 R3	July 2009
[9]	Common Criteria for Information Technology Security Evaluation Evaluation Methodology CCMB-2009-07-004	V3.1 R3	July 2009

Table 23: Supporting Documentation			
Reference	Description	Version	Date
[10]	Network Device Protection Profile	1.1	June 8, 2012
[11]	Security Requirements for Network Devices Errata #1	1.0	December, 19, 2013

10 Annex A: OpenSSH Documented Deviations and Extensions

```
This documents OpenSSH's deviations and extensions to the published SSH
protocol.

Note that OpenSSH's sftp and sftp-server implement revision 3 of the SSH
filexfer protocol described in:

http://www.openssh.com/txt/draft-ietf-secsh-filexfer-02.txt

Newer versions of the draft will not be supported, though some features
are individually implemented as extensions described below.

The protocol used by OpenSSH's ssh-agent is described in the file
PROTOCOL.agent

1. Transport protocol changes

1.1. transport: Protocol 2 MAC algorithm "umac-64@openssh.com"

This is a new transport-layer MAC method using the UMAC algorithm
(rfc4418). This method is identical to the "umac-64" method documented
in:

http://www.openssh.com/txt/draft-miller-secsh-umac-01.txt

1.2. transport: Protocol 2 compression algorithm "zlib@openssh.com"

This transport-layer compression method uses the zlib compression
algorithm (identical to the "zlib" method in rfc4253), but delays the
start of compression until after authentication has completed. This
avoids exposing compression code to attacks from unauthenticated users.

The method is documented in:

http://www.openssh.com/txt/draft-miller-secsh-compression-delayed-00.txt

1.3. transport: New public key algorithms "ssh-rsa-cert-v00@openssh.com",
     "ssh-dsa-cert-v00@openssh.com",
     "ecdsa-sha2-nistp256-cert-v01@openssh.com",
     "ecdsa-sha2-nistp384-cert-v01@openssh.com" and
     "ecdsa-sha2-nistp521-cert-v01@openssh.com"

OpenSSH introduces new public key algorithms to support certificate
authentication for users and hostkeys. These methods are documented in
the file PROTOCOL.certkeys

1.4. transport: Elliptic Curve cryptography

OpenSSH supports ECC key exchange and public key authentication as
specified in RFC5656. Only the ecdsa-sha2-nistp256, ecdsa-sha2-nistp384
and ecdsa-sha2-nistp521 curves over GF(p) are supported. Elliptic
curve points encoded using point compression are NOT accepted or
generated.
```

```
1.5 transport: Protocol 2 Encrypt-then-MAC MAC algorithms

OpenSSH supports MAC algorithms, whose names contain "-etm", that
perform the calculations in a different order to that defined in RFC
4253. These variants use the so-called "encrypt then MAC" ordering,
calculating the MAC over the packet ciphertext rather than the
plaintext. This ordering closes a security flaw in the SSH transport
protocol, where decryption of unauthenticated ciphertext provided a
"decryption oracle" that could, in conjunction with cipher flaws, reveal
session plaintext.

Specifically, the "-etm" MAC algorithms modify the transport protocol
to calculate the MAC over the packet ciphertext and to send the packet
length unencrypted. This is necessary for the transport to obtain the
length of the packet and location of the MAC tag so that it may be
verified without decrypting unauthenticated data.

As such, the MAC covers:

      mac = MAC(key, sequence_number || packet_length || encrypted_packet)

where "packet_length" is encoded as a uint32 and "encrypted_packet"
contains:

      byte      padding_length
      byte[n1]  payload; n1 = packet_length - padding_length - 1
      byte[n2]  random padding; n2 = padding_length

1.6 transport: AES-GCM

OpenSSH supports the AES-GCM algorithm as specified in RFC 5647.
Because of problems with the specification of the key exchange
the behaviour of OpenSSH differs from the RFC as follows:

AES-GCM is only negotiated as the cipher algorithms
"aes128-gcm@openssh.com" or "aes256-gcm@openssh.com" and never as
an MAC algorithm. Additionally, if AES-GCM is selected as the cipher
the exchanged MAC algorithms are ignored and there doesn't have to be
a matching MAC.

2. Connection protocol changes

2.1. connection: Channel write close extension "eow@openssh.com"

The SSH connection protocol (rfc4254) provides the SSH_MSG_CHANNEL_EOF
message to allow an endpoint to signal its peer that it will send no
more data over a channel. Unfortunately, there is no symmetric way for
an endpoint to request that its peer should cease sending data to it
while still keeping the channel open for the endpoint to send data to
the peer.

This is desirable, since it saves the transmission of data that would
otherwise need to be discarded and it allows an endpoint to signal local
processes of the condition, e.g. by closing the corresponding file
descriptor.
```

```
OpenSSH implements a channel extension message to perform this
signalling: "eow@openssh.com" (End Of Write). This message is sent by
an endpoint when the local output of a session channel is closed or
experiences a write error. The message is formatted as follows:

        byte            SSH_MSG_CHANNEL_REQUEST
        uint32          recipient channel
        string          "eow@openssh.com"
        boolean         FALSE

On receiving this message, the peer SHOULD cease sending data of
the channel and MAY signal the process from which the channel data
originates (e.g. by closing its read file descriptor).

As with the symmetric SSH_MSG_CHANNEL_EOF message, the channel does
remain open after a "eow@openssh.com" has been sent and more data may
still be sent in the other direction. This message does not consume
window space and may be sent even if no window space is available.

NB. due to certain broken SSH implementations aborting upon receipt
of this message (in contravention of RFC4254 section 5.4), this
message is only sent to OpenSSH peers (identified by banner).
Other SSH implementations may be whitelisted to receive this message
upon request.

2.2. connection: disallow additional sessions extension
     "no-more-sessions@openssh.com"

Most SSH connections will only ever request a single session, but a
attacker may abuse a running ssh client to surreptitiously open
additional sessions under their control. OpenSSH provides a global
request "no-more-sessions@openssh.com" to mitigate this attack.

When an OpenSSH client expects that it will never open another session
(i.e. it has been started with connection multiplexing disabled), it
will send the following global request:

        byte            SSH_MSG_GLOBAL_REQUEST
        string          "no-more-sessions@openssh.com"
        char            want-reply

On receipt of such a message, an OpenSSH server will refuse to open
future channels of type "session" and instead immediately abort the
connection.

Note that this is not a general defence against compromised clients
(that is impossible), but it thwarts a simple attack.

NB. due to certain broken SSH implementations aborting upon receipt
of this message, the no-more-sessions request is only sent to OpenSSH
servers (identified by banner). Other SSH implementations may be
whitelisted to receive this message upon request.

2.3. connection: Tunnel forward extension "tun@openssh.com"

OpenSSH supports layer 2 and layer 3 tunnelling via the "tun@openssh.com"
channel type. This channel type supports forwarding of network packets
```

```
with datagram boundaries intact between endpoints equipped with
interfaces like the BSD tun(4) device. Tunnel forwarding channels are
requested by the client with the following packet:

        byte            SSH_MSG_CHANNEL_OPEN
        string          "tun@openssh.com"
        uint32          sender channel
        uint32          initial window size
        uint32          maximum packet size
        uint32          tunnel mode
        uint32          remote unit number

The "tunnel mode" parameter specifies whether the tunnel should forward
layer 2 frames or layer 3 packets. It may take one of the following values:

        SSH_TUNMODE_POINTOPOINT  1              /* layer 3 packets */
        SSH_TUNMODE_ETHERNET     2              /* layer 2 frames */

The "tunnel unit number" specifies the remote interface number, or may
be 0x7fffffff to allow the server to automatically chose an interface. A
server that is not willing to open a client-specified unit should refuse
the request with a SSH_MSG_CHANNEL_OPEN_FAILURE error. On successful
open, the server should reply with SSH_MSG_CHANNEL_OPEN_SUCCESS.

Once established the client and server may exchange packet or frames
over the tunnel channel by encapsulating them in SSH protocol strings
and sending them as channel data. This ensures that packet boundaries
are kept intact. Specifically, packets are transmitted using normal
SSH_MSG_CHANNEL_DATA packets:

        byte            SSH_MSG_CHANNEL_DATA
        uint32          recipient channel
        string          data

The contents of the "data" field for layer 3 packets is:

        uint32                  packet length
        uint32                  address family
        byte[packet length - 4]         packet data

The "address family" field identifies the type of packet in the message.
It may be one of:

        SSH_TUN_AF_INET         2               /* IPv4 */
        SSH_TUN_AF_INET6        24              /* IPv6 */

The "packet data" field consists of the IPv4/IPv6 datagram itself
without any link layer header.

The contents of the "data" field for layer 2 packets is:

        uint32                  packet length
        byte[packet length]     frame

The "frame" field contains an IEEE 802.3 Ethernet frame, including
header.
```

```
3. SFTP protocol changes

3.1. sftp: Reversal of arguments to SSH_FXP_SYMLINK

When OpenSSH's sftp-server was implemented, the order of the arguments
to the SSH_FXP_SYMLINK method was inadvertently reversed. Unfortunately,
the reversal was not noticed until the server was widely deployed. Since
fixing this to follow the specification would cause incompatibility, the
current order was retained. For correct operation, clients should send
SSH_FXP_SYMLINK as follows:

       uint32         id
       string         targetpath
       string         linkpath

3.2. sftp: Server extension announcement in SSH_FXP_VERSION

OpenSSH's sftp-server lists the extensions it supports using the
standard extension announcement mechanism in the SSH_FXP_VERSION server
hello packet:

       uint32         3              /* protocol version */
       string         ext1-name
       string         ext1-version
       string         ext2-name
       string         ext2-version
       ...
       string         extN-name
       string         extN-version

Each extension reports its integer version number as an ASCII encoded
string, e.g. "1". The version will be incremented if the extension is
ever changed in an incompatible way. The server MAY advertise the same
extension with multiple versions (though this is unlikely). Clients MUST
check the version number before attempting to use the extension.

3.3. sftp: Extension request "posix-rename@openssh.com"

This operation provides a rename operation with POSIX semantics, which
are different to those provided by the standard SSH_FXP_RENAME in
draft-ietf-secsh-filexfer-02.txt. This request is implemented as a
SSH_FXP_EXTENDED request with the following format:

       uint32         id
       string         "posix-rename@openssh.com"
       string         oldpath
       string         newpath

On receiving this request the server will perform the POSIX operation
rename(oldpath, newpath) and will respond with a SSH_FXP_STATUS message.
This extension is advertised in the SSH_FXP_VERSION hello with version
"1".

3.4. sftp: Extension requests "statvfs@openssh.com" and
         "fstatvfs@openssh.com"

These requests correspond to the statvfs and fstatvfs POSIX system
```

```
interfaces. The "statvfs@openssh.com" request operates on an explicit
pathname, and is formatted as follows:

        uint32          id
        string          "statvfs@openssh.com"
        string          path

The "fstatvfs@openssh.com" operates on an open file handle:

        uint32          id
        string          "fstatvfs@openssh.com"
        string          handle

These requests return a SSH_FXP_STATUS reply on failure. On success they
return the following SSH_FXP_EXTENDED_REPLY reply:

        uint32          id
        uint64          f_bsize         /* file system block size */
        uint64          f_frsize        /* fundamental fs block size */
        uint64          f_blocks        /* number of blocks (unit f_frsize) */
        uint64          f_bfree         /* free blocks in file system */
        uint64          f_bavail        /* free blocks for non-root */
        uint64          f_files         /* total file inodes */
        uint64          f_ffree         /* free file inodes */
        uint64          f_favail        /* free file inodes for to non-root */
        uint64          f_fsid          /* file system id */
        uint64          f_flag          /* bit mask of f_flag values */
        uint64          f_namemax       /* maximum filename length */

The values of the f_flag bitmask are as follows:

        #define SSH_FXE_STATVFS_ST_RDONLY     0x1     /* read-only */
        #define SSH_FXE_STATVFS_ST_NOSUID     0x2     /* no setuid */

Both the "statvfs@openssh.com" and "fstatvfs@openssh.com" extensions are
advertised in the SSH_FXP_VERSION hello with version "2".

10. sftp: Extension request "hardlink@openssh.com"

This request is for creating a hard link to a regular file. This
request is implemented as a SSH_FXP_EXTENDED request with the
following format:

        uint32          id
        string          "hardlink@openssh.com"
        string          oldpath
        string          newpath

On receiving this request the server will perform the operation
link(oldpath, newpath) and will respond with a SSH_FXP_STATUS message.
This extension is advertised in the SSH_FXP_VERSION hello with version
"1".

$OpenBSD: PROTOCOL,v 1.20 2013/01/08 18:49:04 markus Exp $
```