Motorola Network Router
Security Target

16-3324-R-0008

Version: 1.1

March 22, 2017

Prepared For:

Motorola Solutions, Inc.

1303 East Algonquin Road
Schaumburg, Illinois 60196 USA

Prepared By:

UL Verification Services Inc.
Motorola Solutions, Inc.

Notices:

Table of Contents

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 v3.1r4 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:	Motorola Network Router Security Target
ST Version:	1.1
ST Author(s):	UL Verification Services Inc., Motorola Solutions, Inc.
ST Publication Date:	March 22, 2017
Keywords	Network Device

1.2 Target of Evaluation Reference

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

TOE Developer	Motorola Solutions, Inc.
	1303 East Algonquin Road
	Schaumburg, Illinois 60196 USA
TOE Name:	Motorola Network Devices, S6000 and GGM 8000 with EOS version 16.9

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 Motorola Network Device models S6000 and GGM 8000 provide a flexible routing solution for integrated data, voice and virtual private network (VPN) applications.

These solutions feature the Motorola Enterprise OS software suite with a choice of two hardware platforms: S6000/GGM 8000 series. Each series provides different throughput and scalability capabilities. The common OS software provides Enterprise networking features including: traffic shaping and Quality of Service (QoS), WAN/LAN connectivity, Voice & Multi-Service and Network Management support. A comprehensive set of security features provide network and data protected through:

- Firewall Features: Pre-defined attack types, custom traffic filters.
- Encryption support: The TOE is FIPS 140-2 validated to Level 1 (S6000, FIPS 140-2 cert #2857) or Level 2 (GGM 8000, FIPS 140-2 cert #2858).
- Secure Tunneling/VPN support: IPsec, FRF.17, and IKE.
- Protocol Authentication: Border Gateway Protocol (BGP), Open Shortest Path First (OSPF), and Protocol Independent Multicast (PIM) protocols.

The Network Device features a comprehensive Administrative-user interface that allows for the setup, configuration, monitoring and management of the device using a Command Line Interface (CLI) over a local console interface or secured over an SSHv2 secured connection.

The GGM 8000 platform is suitable for use as edge routers for analog and digital voice systems as well as remote radio frequency (RF) site routers in digital voice systems. The GGM 8000 may include up to 2 V.24 modules that allow the processing of digital voice, Voice over IP (VoIP). When combined with the analog conventional pluggable module (E&M), the GGM 8000 is also suitable as a Conventional Channel Gateway (CCGW) in a Motorola ASTRO® 25 trunked radio communication network. In this role, the TOE exchanges call control traffic via communication with peer devices with ASTRO® 25 controllers.

The E&M pluggable module cannot be used with the S6000 platform.

The GGM 8000 and S6000 series are suitable as a Wide Area Network (WAN) interface for radio communications network transport systems or as a Core/Edge Network Device.

The GGM 8000 and S6000 series can also be used to maintain connectivity among small, midsize, and large Local Area networks via a wide variety of WAN services and accommodates extensive virtual port tunneling capabilities with data compression and high speed processing.

When used in the network core, the GGM 8000 and S6000 supply high speed, scalable performance for WAN concentration, virtual private network (VPN) tunnel termination, and efficient bandwidth utilization. However the S6000 concentrates T1/E1 or T3/E3 internet traffic at the network core, enabling multiple secure tunnels to be maintained through the public network to many remote locations simultaneously.

1.3.3 TOE Major Security Features Summary

- Audit

- Cryptography
- User Data Protection
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted Path/Channels

1.3.4 TOE IT environment hardware/software/firmware requirements

This table identifies components that must be present in the Operational Environment to support the operation of the TOE.

Table 1: Operational Environment Components	
Component	Description
RADIUS	Authentication Server (optional)[1] with IPsec peer capabilities
Syslog Host	Syslog host for offloading of audit records with IPsec peer capabilities
NTP Server	NTP Server with IPsec peer capabilities
SSHv2 client	SSHv2 client to support Administrative tunnels to the TOE
Serial Console	Console to perform local administration of the TOE.
HTTP Server for CRL	CRL Distribution Point

1.4 Target of Evaluation Description

Though section 1.3 describes the full functionality of the product, not all of those features are within the scope of a NIAP Common Criteria evaluation project. Only the functions described in this section (1.4) are within scope of the certification.

1.4.1 Target of Evaluation Physical Boundaries

The TOE consists of the following:

GGM 8000

Table 2: GGM 8000 Hardware	
Tanapa Number	Description
CLN1841F Rev AB	GGM 8000 Base Unit
CLN8787A Rev B	FIPS 140-2 Kit
CLN1850A Rev G	AC Power Option[2]
CLN1849C Rev AA	DC Power Option[3]
	Choice of Pluggable Modules for the GGM 8000 from Table 4
	Optional Analog CCGW support

With EOS Software SW/GGM8000-KS, 16.9.0.40 and Firmware BM/GGM8000, 16.9.0.40.

[1] If your organization requires authentication failure counters and account lockouts for remote accounts, ensure your RADIUS Server supports these features.

[2] Either the AC or DC Power Option must be selected.

[3] Either the AC or DC Power Option must be selected.

S6000

Table 3: S6000 Hardware	
Tanapa Number	Description
CLN1780L Rev FB	S6000 Base Unit
CLN8261D Rev NA	Encryption Module
	Choice of Pluggable Modules for the S6000 from Table 4
	Optional Analog CCGW support

With EOS Software SW/S6000-GS, 16.9.0.40 and Firmware FW/S6000, 16.9.0.40.

Table 4: Pluggable Module Combinations by Hardware Platform Shaded = N/A Numbers indicate possible configuration options (number of modules supported per chassis). A single hardware platform device of one of the two shown is required.		
Module Type	S6000	GGM 8000
T1/E1 (WAN/Telco), 2 ports per module		0, 1, 2
T1/E1 (UltraWAN), 4 ports per module	0, 1, 2	
T1/E1, 12 ports per module	0, 1, 2	
FlexWAN Serial, 1 port per module		0, 1, 2
FlexWAN Serial, 4 ports per module	0, 1, 2	
V.24, 2 ports per module		0, 1, 2
T3/E3, 2 ports (one T3/E3) per module	0, 1, 2	

Table 5: Hardware Features		
Implementation Characteristics	S6000	GGM 8000
CPU Internal Operating Frequency	1GHz	1GHz
Level-1 Instruction Cache Size / Structure	32KB, 8-Sets (Built-In)	32KB, 8-Way Set Associative
Level-1 Data Cache Size / Structure	32KB, 8-Sets (Built-In)	32KB, 8-Way Set Associative
Level-2 Cache Size	512KB (Built-In)	512KB
Cache Coherency on Shared Memory Accesses	Yes	Yes
Shared Memory Type	SDRAM	DDR2
Shared Memory Size	256 MB (DIMM)	512 MB
Shared Memory Bus Width	64 Bits	64 Bits
Shared Memory Peak Transfer Rate	1,064 MBS (133 MTS)	3,200 MBS
Embedded SW (Flash PROM Memory)	1 MB	32 MB
Flash File System (Flash PROM Memory)	16 MB	64 MB
Built-In LAN Ports	3 - 10/100	4 – 10/100/1000
Built-In WAN Ports	None	2 – T1/E1
Pluggable Module Options4	Slots for two I/O Modules	Slots for two I/O Modules
Analog CCGW option (4 Port E&M Analog module and DSP module)	No	Yes

The guidance documentation that is part of the TOE is listed in Section 9 “References” within Table 20: TOE Guidance Documentation.

[4] Table 2 specifies the maximum number of each module type that each base unit supports.

1.4.2 Target of Evaluation Description

The logical boundary of the TOE includes 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 are provided in Section 7 “TOE Summary Specification”. All TOE features not described are not within the scope of this certification.

1.4.2.1 Audit

- The TOE will audit all events and information defined in Table 11: Auditable Events.
- The TOE will also include the identity of the user that caused the event (if applicable), date and time of the event, type of event, and the outcome of the event.
- The TOE protects storage of audit information from unauthorized deletion.
- The TOE prevents unauthorized modifications to the stored audit records.
- The TOE can transmit audit data to an external IT entity using IPsec protocol.

1.4.2.2 Cryptographic Operations

The TSF performs the following cryptographic operations:

- SSH with AES-CBC-128 or AES-CBC-256 for protection of remote administrative sessions.
- IPsec with AES-CBC-128 or AES-CBC-256 for protection of communication paths with RADIUS, Syslog, and NTP hosts/servers.

The TSF zeroizes all plaintext secret and private cryptographic keys and CSPs once they are no longer required.

1.4.2.3 Identification and Authentication

- The TSF supports passwords consisting of alphanumeric and special characters. The TSF also allows administrators to set a minimum password length and support passwords with 15 characters or more.
- The TSF requires all administrative-users to authenticate before allowing the user to perform any actions other than:
    - Viewing the warning banner
    - ARP
    - ICMP
    - Routing Services
    - BFD Send
    - DHCP Services
    - SSH
    - IPDV (port UDP/49402)
    - RSVP (port UDP/1698)
    - NTP (port UDP/123)
- The TSF allows for authentication via password or public-key infrastructure (PKI).
- All authentication information is obfuscated.
- The TOE supports the use of X.509 certificates for the purposes of IPsec peer authentication, including support for creating and validating certificates.

1.4.2.4 Security Management

- The TOE manages the following TSF data:
  - User account names
  - User passwords
  - Internally generated cryptographic keys
  - Imported SSH public keys
- The only role in the TOE is that of the Administrator.
- All administration is performed over an SSH connection or via direct console session.

1.4.2.5 Protection of the TSF

- The TSF prevents the reading of secret and private keys.
- The TOE provides reliable time stamps for itself.
- The TOE runs a suite of self-tests during the initial start-up (upon power on) to demonstrate the correction operation of the TSF.
- The TOE provides a means to verify firmware/software updates to the TOE using a digital signature mechanism prior to installing those updates.

1.4.2.6 TOE Access

- The TOE, for local interactive sessions, terminates the administrative session after an Authorized Administrator-specified period of session inactivity.
- The TOE terminates a remote interactive session after an Authorized Administrator-configurable period of session inactivity.
- The TOE allows Administrator-initiated termination of the Administrator's own interactive session.
- Before establishing an administrative user session, the TOE is capable of displaying an Authorized Administrator-specified advisory notice and consent warning message regarding unauthorized use of the TOE.

1.4.2.7 Trusted Path/Channels

- The TOE uses IPsec to provide a trusted communication channel between itself and all authorized IT entities 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.
- The TOE permits the TSF, or the authorized IT entities to initiate communication via the trusted channel.
- The TOE permits remote administrators to initiate communication over SSH.
- The TOE requires the use of the trusted path for initial administrator authentication and all remote administration actions.

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 collaborative Protection Profile for Network Devices are marked “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 those 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 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 underlined text**.** Selections within selections will be identified with double underline text.

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

In the event that Technical Rapid Response Team (TRRT) or Technical Decision (TD) feedback results in modifications to any SFR, this change will be indicated by an associated footnote, and no other marking.

2. Conformance Claims

2.1 Common Criteria Conformance Claims

This Security Target is conformant to the Common Criteria Version 3.1r4, CC Part 2 extended [C2], and CC Part 3 conformant [C3].

2.2 Conformance to Protection Profiles

This Security Target claims exact compliance to the collaborative Protection Profile for Network Devices, Version 1.0, dated February 27, 2015 [NDcPP] and Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP, Version 1.0, dated February 27, 2015 [SD]. This Protection Profile will be referred to as cPP or PP for convenience throughout this Security Target.

2.3 Conformance to Security Packages

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

2.4 Conformance Claims Rationale

To demonstrate that exact conformance is met, this rationale shows all threats are addressed, all OSP are satisfied, no additional assumptions are made, all objectives have been addressed, and all SFRs and SARs have been instantiated.

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

- Threats
    - All threats defined in the cPP are carried forward to this ST;
    - No additional threats have been defined in this ST.
- Organizational Security Policies
    - All OSP defined in the cPP are carried forward to this ST;

o No additional OSPs have been defined in this ST.

- Assumptions
    - All assumptions defined in the cPP 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 cPP are carried forward to this ST.
- All SFRs and SARs defined in the cPP 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 cPP have been properly instantiated in this Security Target; therefore, this ST shows exact compliance to the cPP.

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 6: Threats

Threat	Description
T.UNAUTHORIZED_ADMINISTRATOR_ACCESS	Threat agents may attempt to gain administrator access to the network device by nefarious means such as masquerading as an administrator to the device, masquerading as the device to an administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between network devices. Successfully gaining administrator access allows malicious actions that compromise the security functionality of the device and the network on which it resides.
T.WEAK_CRYPTOGRAPHY	Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.
T.UNTRUSTED_COMMUNICATION_CHANNELS	Threat agents may attempt to target network devices that do not use standardized secure tunneling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the network device itself.
T.WEAK_AUTHENTICATION_ENDPOINTS	Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints – e.g., shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the network device itself could be compromised.
T.UPDATE_COMPROMISE	Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.
T.UNDETECTED_ACTIVITY	Threat agents may attempt to access, change, and/or modify the security functionality of the network device without administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the administrator would have no knowledge that the device has been compromised.
T.SECURITY_FUNCTIONALITY_COMPROMISE	Threat agents may compromise credentials and device data enabling continued access to the network device and its critical data. The compromise of credentials include replacing existing credentials with an attacker's credentials, modifying

Table 6: Threats	
Threat	Description
	existing credentials, or obtaining the administrator or device credentials for use by the attacker.
T.PASSWORD_CRACKING	Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the device. Having privileged access to the device provides the attacker unfettered access to the network traffic, and may allow them to take advantage of any trust relationships with other network devices.
T.SECURITY_FUNCTIONALITY_FAILURE	A component of the network device may fail during start-up or during operations causing a compromise or failure in the security functionality of the network device, leaving the device susceptible to attackers.

3.2 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 7: 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 on the operational environment in which the TOE is intended to be used. 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 8: Assumptions	
Assumption	Description
A.PHYSICAL_PROTECTION	The network device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security and/or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, the cPP will not include any requirements on physical tamper protection or other physical attack mitigations. The cPP will not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device.
A.LIMITED_FUNCTIONALITY	The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example the device should not provide computing platform for general purpose Applications (unrelated to networking functionality).
A.NO_THRU_TRAFFIC_PROTECTION	A standard/generic network device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the network device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the network device, destined for another network entity, is not

Table 8: Assumptions	
Assumption	Description
	covered by the ND cPP. It is assumed that this protection will be covered by cPPs for particular types of network devices (e.g, firewall).
A.TRUSTED_ADMINISTRATOR	The Security Administrator(s) for the network device are assumed to be trusted and to act in the best interest of security for the organization. This includes being appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The network device is not expected to be capable of defending against a malicious administrator that actively works to bypass or compromise the security of the device.
A.REGULAR_UPDATES	The network device firmware and software is assumed to be updated by an administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.ADMIN_CREDENTIALS_SECURE	The administrator's credentials (private key) used to access the network device are protected by the platform on which they reside.

4. Security Objectives

4.1 Security Objectives for the Operational Environment

Table 9: Security Objectives for the Operational Environment	
Objective	Description
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.NO_THRU_TRAFFIC_PROTECTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all guidance documentation in a trusted manner.
OE.UPDATES	The TOE firmware and software is updated by an administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_SECURE	The administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.

5. 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 cPP.

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 cPP.

6. Security Requirements

This section describes the security functional and assurance requirements for the TOE; those that are CC Part 2 conformant, CC Part 2 extended, CC Part 3 conformant, and CC Part 3 extended.

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 cPP are not signified in this section. Operations performed by the ST are denoted according to the formatting conventions in Section 1.5.

Table 10: Security Functional Requirements

#	SFR	Description
1	FAU_GEN.1	Audit Data Generation
2	FAU_GEN.2	User Audit Association
3	FAU_STG.1	Protected Audit Trail Storage
4	FAU_STG_EXT.1	Protected Audit Event Storage
5	FCS_CKM.1	Cryptographic Key Generation (Refined)
6	FCS_CKM.2	Cryptographic Key Establishment (Refined)
7	FCS_CKM.4	Cryptographic Key Destruction
8	FCS_COP.1(1)	Cryptographic Operation (AES Data Encryption/Decryption)
9	FCS_COP.1(2)	Cryptographic Operation (Signature Generation and Verification)
10	FCS_COP.1(3)	Cryptographic Operation (Hash Algorithm)
11	FCS_COP.1(4)	Cryptographic Operation (Keyed Hash Algorithm)
12	FCS_IPSEC_EXT.1	IPsec Protocol
13	FCS_RBG_EXT.1.2	Random Bit Generation
14	FCS_SSHS_EXT.1	SSH Server Protocol
15	FIA_PGM_EXT.1	Password Management
16	FIA_UIA_EXT.1	User Identification and Authentication
17	FIA_UAU_EXT.2	Password-based Authentication Mechanism
18	FIA_UAU.7	Protected Authentication Feedback
19	FIA_X509_EXT.1	X.509 Certificate Validation
20	FIA_X509_EXT.2	X.509 Certificate Authentication
21	FIA_X509_EXT.3	X.509 Certificate Requests
22	FMT_MOF.1(1)/Trusted Update	Management of Security Functions Behaviour
23	FMT_MOF.1(1)/Audit	Management of Security Functions Behaviour
24	FMT_MOF.1(2)/Audit	Management of Security Functions Behaviour
25	FMT_MOF.1(1)/AdminAct	Management of Security Functions Behaviour

Table 10: Security Functional Requirements		
#	SFR	Description
26	FMT_MOF.1(2)/AdminAct	Management of Security Functions Behaviour
27	FMT_MTD.1	Management of TSF Data
28	FMT_MTD.1/AdminAct	Management of TSF Data
29	FMT_SMF.1	Specification of Management Functions
30	FMT_SMR.2	Restrictions on Security Roles
31	FPT_APW_EXT.1	Protection of Administrator Passwords
32	FPT_SKP_EXT.1	Protection of TSF Data (for reading all symmetric keys)
33	FPT_STM.1	Reliable Time Stamps
34	FPT_TST_EXT.1(1)	TSF Testing (on power on)
35	FPT_TST_EXT.1(2)	TSF Testing (conditional)
36	FPT_TUD_EXT.1	Trusted Update
37	FTA_SSL_EXT.1	TSF-initiated Session Locking
38	FTA_SSL.3	TSF-initiated Termination
39	FTA_SSL.4	User-initiated Termination
40	FTA_TAB.1	Default TOE Access Banners
41	FTP_ITC.1	Inter-TSF Trusted Channel
42	FTP_TRP.1	Trusted Path

6.1.1 Class FAU: Security Audit

6.1.1.1 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1

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

a) Start-up and shut-down of the audit functions;
b) All auditable events for the not specified level of audit; and
c) All administrative actions comprising:
   - Administrative login and logout (name of user account shall be logged if individual user accounts are required for administrators).
   - Security related configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).
   - Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).
   - Resetting passwords (name of related user account shall be logged).
   - Starting and stopping services (if applicable)
   - no other actions
d) Specifically defined auditable events listed in Table 11.

Application Note 1

If the list of 'administrative actions' appears to be incomplete, the assignment in the selection should be used to list additional administrative actions which are audited.

The ST author replaces the cross-reference to the table of audit events with an appropriate cross-reference for the ST. This must also include the relevant parts of Table 3 and Table 4 for optional and selection-based SFRs included in the ST.

Application Note 2

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

The TSS should identify what information is logged to identify the relevant key for the administrative task of generating/import of, changing, or deleting of cryptographic keys.

With respect to FAU_GEN.1.1 the term 'services' refers to trusted path and trusted channel communications, on demand self-tests, trusted update and administrator sessions (that exist under the trusted path) (e.g. netconf).

FAU_GEN1.2

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, information specified in column three of Table 11.

Application Note 3

The ST author replaces the cross-reference to the table of audit events with an appropriate cross-reference for the ST. This must also include the relevant parts of cPP Table 3 and cPP Table 4 for optional and selection-based SFRs included in the ST.

Table 11: Auditable Events

#	SFR	Auditable Events	Additional Audit Record Contents
1	FAU_GEN.1	None.	None.
2	FAU_GEN.2	None.	None.
3	FAU_STG.1	None.	None.
4	FAU_STG_EXT.1	None.	None.
5	FCS_CKM.1	None.	None.
6	FCS_CKM.2	None.	None.
7	FCS_CKM.4	None.	None.
8	FCS_COP.1(1)	None.	None.
9	FCS_COP.1(2)	None.	None.
10	FCS_COP.1(3)	None.	None.
11	FCS_COP.1(4)	None.	None.
12	FCS_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure
13	FCS_RBG_EXT.1	None.	None.
14	FCS_SSHS_EXT.1	Failure to establish an SSH	Reason for failure.

Table 11: Auditable Events			
#	SFR	Auditable Events	Additional Audit Record Contents
		session	
		Successful SSH rekey	Non-TOE endpoint of connection (IP address)
15	FCS_PMG_EXT.1	None.	None.
16	FIA_UIA_EXT.1	All use of the identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
17	FIA_UAU_EXT.2	All use of the identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
18	FIA_UAU.7	None.	None.
19	FIA_X509_EXT.1	Unsuccessful attempt to validate a certificate.	Reason for Failure
20	FIA_X509_EXT.2	None.	None.
21	FIA_X509_EXT.3	None.	None.
22	FMT_MOF.1(1)/ TrustedUpdate	Any attempt to initiate a manual update.	None.
23	FMT_MOF.1(1)/ Audit	Modification of the behaviour of the transmission of audit data to an external identity.	None.
24	FMT_MOF.1(2)/ Audit	Modification of the behaviour of the handling of audit data.	None.
25	FMT_MOF.1(1)/ AdminAct	Modification of the behavior of the TSF.	None.
26	FMT_MOF.1(2)/ AdminAct	Starting and stopping of services.	None.
27	FMT_MTD.1	All management activities of TSF data.	None.
28	FMT_MTD.1/ AdminAct	Modification, deletion, generation/import of cryptographic keys.	None.
29	FMT_SMF.1	None.	None.
30	FMT_SMR.2	None.	None.
31	FPT_APW_EXT.1	None.	None.
32	FPT_SKP_EXT.1	None.	None.
33	FPT_STM.1	Changes to time.	The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
34	FPT_TST_EXT.1(1)	None.	None.
35	FPT_TST_EXT.1(2)	None.	None.
36	FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	No additional information.
37	FTA_SSL_EXT.1	Any attempts at unlocking of an interactive session.	None.

Table 11: Auditable Events			
#	SFR	Auditable Events	Additional Audit Record Contents
38	FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
39	FTA_SSL.4	The termination of an interactive session.	None.
40	FTA_TAB.1	None.	None.
41	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.
42	FTP_TRP.1	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	Identification of the claimed user identity.

Application Note 4

Additional audit events will apply to the TOE depending on the optional and selection-based requirements adopted from Appendix A and Appendix B. The ST author must therefore include the relevant additional events specified in the tables in [NDcPP] Table 3 and [NDcPP] Table 4. The audit event for FIA_X509_EXT.1 is based on the TOE not being able to complete the certificate validation by ensuring the following:

- *the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.*
- *Verification of the digital signature of the trusted hierarchical CA*
- *read/access the CRL or access the OCSP server.*

If any of these checks fails, then an audit event with the failure should be written to the audit log

Assurance Activity:

Guidance

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_GEN1.2, and the additional information specified in the table of auditable events.

The evaluator shall also make a determination of the administrative actions that are relevant in the context of cPP. The evaluator shall examine the guidance documentation 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 cPP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are security relevant with respect to the cPP. The evaluator may perform this activity as part of the activities associated with ensuring that the corresponding guidance documentation satisfies the requirements related to it.

Test

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 the table of audit events and administrative actions listed above. 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. Logging of all activities related to trusted update should be tested in detail and with utmost diligence. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the guidance documentation, 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.

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.1 Protected Audit Trail Storage

FAU_STG.1.1

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

FAU_STG.1.2

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

Assurance Activity

TSS

The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally and how these records are protected against unauthorized modification or deletion. The evaluator shall ensure that the TSS describes the conditions that must be met for authorized deletion of audit records.

Guidance

The evaluator shall examine the guidance documentation to determine that it describes any configuration required for protection of the locally stored audit data against unauthorized modification or deletion.

Test

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall access the audit trail as an unauthorized administrator and attempt to modify and delete the audit records. The evaluator shall verify that these attempts fail.
b) Test 2: The evaluator shall access the audit trail as an authorized administrator and attempt to delete the audit records. The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records authorized for deletion are deleted.

6.1.1.4 FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1

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

Application Note 5

For selecting the option of transmission of generated audit data to an external IT entity the TOE relies on a non-TOE audit server for storage and review of 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 in that case.

FAU_STG_EXT.1.2

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

FAU_STG_EXT.1.3

The TSF shall overwrite previous audit records according to the following rule: **overwrite the oldest stored audit records**, no other action when the local storage space for audit data is full.

Application Note 6

The external log server might be used as alternative storage space in case the local storage space is full. The 'other action' could in this case be defined as 'send the new audit data to an external IT entity'.

Assurance Activity

TSS

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.

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.

If the TOE complies with FAU_STG_EXT.2 the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2 are correct when performing the tests for FAU_STG_EXT.1.3.

The evaluator shall examine the TSS to ensure that it details the behaviour of the TOE when the storage space for audit data is full. When the option 'overwrite previous audit record' is selected this description should include an outline of the rule for overwriting audit data. If 'other actions' are chosen such as sending the new audit data to an external IT entity, then the related behaviour of the TOE shall also be detailed in the TSS.

Guidance

The evaluator shall also examine the guidance documentation 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 also examine the guidance documentation 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 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.

The evaluator shall also ensure that the guidance documentation describes all possible configuration options for FAU_STG_EXT.1.3 and the resulting behaviour of the TOE for each possible configuration. The description of possible configuration options and resulting behaviour shall correspond to those described in the TSS.

Test

Testing of the trusted channel mechanism for audit will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall perform the following additional test for this requirement:

a) 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.

The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behaviour defined in FAU_STG_EXT.1.3. Depending on the configuration this means that the evaluator has to check the content of the audit data when the audit data is just filled to the maximum and then verifies that:

a) The audit data remains unchanged with every new auditable event that should be tracked but that the audit data is recorded again after the local storage for audit data is cleared (for the option 'drop new audit data' in FAU_STG_EXT.1.3).
b) The existing audit data is overwritten with every new auditable event that should be tracked according to the specified rule (for the option 'overwrite previous audit records' in FAU_STG_EXT.1.3)
c) The TOE behaves as specified (for the option 'other action' in FAU_STG_EXT.1.3).

6.1.2 Class FCS: Cryptographic Support

6.1.2.1 FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1

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

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

Application Note 7

The ST author selects all key generation schemes used for key establishment and device authentication. When key generation is used for key establishment, the schemes in FCS_CKM.2.1 and selected

cryptographic protocols must match the selection. When key generation is used for device authentication, the public key is expected to be associated with an X.509v3 certificate.
If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.

Assurance Activity

TSS

The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Guidance

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all uses defined in this PP.

Test

See Annex A for Algorithm Validation Assurance Activities.

6.1.2.2 FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1

The TSF shall perform cryptographic key establishment in accordance with a specified cryptographic key establishment method:

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

Application Note 8

This is a refinement of the SFR FCS_CKM.2 to deal with key establishment rather than key distribution.

The ST author selects all key establishment schemes used for the selected cryptographic protocols.

The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B; however, Section 9 relies on implementation of other sections in SP 800-56B. If the TOE acts as a receiver in the RSA key establishment scheme, the TOE does not need to implement RSA key generation.

The elliptic curves used for the key establishment scheme correlate with the curves specified in FCS_CKM.1.1.

The domain parameters used for the finite field-based key establishment scheme are specified by the key generation according to FCS_CKM.1.1.

Assurance Activity

TSS

The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Guidance

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Test

See Annex A for Algorithm Validation Assurance Activities.

6.1.2.3 FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1[5]

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

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

that meets the following: No Standard.

Assurance Activity

TSS

The evaluator shall check to ensure the TSS lists each type of plaintext key material and its origin and storage location.

The evaluator shall verify that the TSS describes when each type of key material is cleared (for example, on system power off, on wipe function, on disconnection of trusted channels, when no longer needed by the trusted channel per the protocol, etc.).

The evaluator shall also verify that, for each type of key, the type of clearing procedure that is performed (cryptographic erase, overwrite with zeros, overwrite with random pattern, or block erase) is listed. 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 clearing procedure in terms of the memory in which the data are stored (for example, "secret keys stored on flash are cleared by overwriting once with zeros, while secret keys stored on the internal persistent storage device are cleared by overwriting three times with a random pattern that is changed before each write").

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

FCS_COP.1.1(1)

The TSF shall perform encryption/decryption in accordance with a specified cryptographic algorithm AES used in CBC mode and cryptographic key sizes 128 bits, 256 bits that meet the following: AES as specified in ISO 18033-3, CBC as specified in ISO 10116.

Application Note 9

For the first selection of FCS_COP.1.1(1), the ST author should choose the mode or modes in which AES operates. For the second selection, the ST author should choose the key sizes that are supported by this

[5] SFR modification is based on TD0130

functionality. The modes and key sizes selected here correspond to the cipher suite selections made in the trusted channel requirements.

Assurance Activity

Test

See Annex A for Algorithm Validation Assurance Activities.

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

FCS_COP.1.1(2)

The TSF shall perform cryptographic signature services (generation and verification) in accordance with a specified cryptographic algorithm:

- RSA Digital Signature Algorithm and cryptographic key sizes (modulus) **2048 bits**,
- Elliptic Curve Digital Signature Algorithm and cryptographic key sizes: **256 bits or 384 bits**

that meet the following:

- For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,
- For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" P-256, P-384, and no other curves; ISO/IEC 14888-3, Section 6.4.

Application Note 10

The ST Author should choose the algorithm implemented to perform digital signatures. For the algorithm(s) chosen, the ST author should make the appropriate assignments/selections to specify the parameters that are implemented for that algorithm.

ST Author Note

*The [NDcPP] incorrectly references "RSASSA-PKCS**2**v1_5." This ST has been updated to reflect the intended standard "RSASSA-PKCS**1**v1_5."*

Assurance Activity

Test

See Annex A for Algorithm Validation Assurance Activities.

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

FCS_COP.1.1(3)

The TSF shall perform cryptographic hashing services in accordance with a specified cryptographic algorithm SHA-1, SHA-256, SHA-384 that meet the following: ISO/IEC 10118-3:2004.

Application Note 11

Vendors are strongly encouraged to implement updated protocols that support the SHA-2 family; until updated protocols are supported, this PP allows support for SHA-1 implementations in compliance with SP 800-131A. I

Assurance Activity

TSS

The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Guidance

The evaluator checks the AGD documents to determine that any configuration that is required to configure the required hash sizes is present.

Test

See Annex A for Algorithm Validation Assurance Activities.

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

FCS_COP.1.1(4)

The TSF shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and cryptographic key sizes **112-1024 bits** and message digest sizes 160, 256, 384 bits that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

Application Note 12

The key size [k] in the assignment falls into a range between L1 and L2 (defined in ISO/IEC 10118 for the appropriate hash function). For example, for SHA-256, L1=512, L2=256, where L2<=k<=L1.

Assurance Activity

TSS

The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

Test

For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and IV using a known good implementation.

6.1.2.8 FCS_IPSEC_EXT.1 IPsec Protocol

FCS_IPSEC_EXT.1.1

The TSF shall implement the IPsec architecture as specified in RFC 4301.

Application Note 52

RFC 4301 calls for an IPsec implementation to protect IP traffic through the use of a Security Policy Database (SPD). The SPD is used to define how IP packets are to be handled: PROTECT the packet (e.g., encrypt the packet), BYPASS the IPsec services (e.g., no encryption), or DISCARD the packet (e.g., drop the packet). The SPD can be implemented in various ways, including router access control lists, firewall rulesets, a "traditional" SPD, etc. Regardless of the implementation details, there is a notion of a "rule" that a packet is "matched" against and a resulting action that takes place.

While there must be a means to order the rules, a general approach to ordering is not mandated, as long as the SPD can distinguish the IP packets and apply the rules accordingly. There may be multiple SPDs (one for each network interface), but this is not required.

Assurance Activity

TSS

The evaluator shall examine the TSS and determine that it describes what takes place when a packet is processed by the TOE, e.g., the algorithm used to process the packet. The TSS describes how the SPD is implemented and the rules for processing both inbound and outbound packets in terms of the IPsec policy. The TSS describes the rules that are available and the resulting actions available after matching a rule. The TSS describes how those rules and actions form the SPD in terms of the BYPASS (e.g., no encryption), DISCARD (e.g., drop the packet), and PROTECT (e.g., encrypt the packet) actions defined in RFC 4301.

As noted in section 4.4.1 of RFC 4301, the processing of entries in the SPD is non-trivial and the evaluator shall determine that the description in the TSS is sufficient to determine which rules will be applied given the rule structure implemented by the TOE. For example, if the TOE allows specification of ranges, conditional rules, etc., the evaluator shall determine that the description of rule processing (for both inbound and outbound packets) is sufficient to determine the action that will be applied, especially in the case where two different rules may apply. This description shall cover both the initial packets (that is, no SA is established on the interface or for that particular packet) as well as packets that are part of an established SA.

Guidance

The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to construct entries into the SPD that specify a rule for processing a packet. The description includes all three cases – a rule that ensures packets are encrypted/decrypted, dropped, and flow through the TOE without being encrypted. The evaluator shall determine that the description in the guidance documentation is consistent with the description in the TSS, and that the level of detail in the guidance documentation is sufficient to allow the administrator to set up the SPD in an unambiguous fashion. This includes a discussion of how ordering of rules impacts the processing of an IP packet.

Test

The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

a) Test 1: The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The selectors used in the construction of the rule shall be different such that the evaluator can generate a packet and send packets to the gateway with the appropriate fields (fields that are used by the rule - e.g., the IP addresses, TCP/UDP ports) in the packet header. The evaluator performs both positive and negative test cases for each type of rule (e.g. a packet that matches the rule and another that does not match the rule). The evaluator observes via the audit trail, and packet captures that the TOE exhibited the expected behavior: appropriate packets were dropped, allowed to flow without modification, encrypted by the IPsec implementation.
b) Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. As with Test 1, the evaluator ensures both positive and negative test cases are constructed. These scenarios must exercise the range of possibilities for SPD entries and processing modes as outlined in the TSS and guidance documentation. Potential areas to cover include rules with overlapping ranges and conflicting entries, inbound and outbound packets, and

packets that establish SAs as well as packets that belong to established SAs. The evaluator shall verify, via the audit trail and packet captures, for each scenario that the expected behavior is exhibited, and is consistent with both the TSS and the guidance documentation.

FCS_IPSEC_EXT.1.2

The TSF shall have a nominal, final entry in the SPD that matches anything that is otherwise unmatched, and discards it.

Assurance Activity

Test

The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The evaluator may use the SPD that was created for verification of FCS_IPSEC_EXT.1.1. The evaluator shall construct a network packet that matches the rule to allow the packet to flow in plaintext and send that packet. The evaluator should observe that the network packet is passed to the proper destination interface with no modification. The evaluator shall then modify a field in the packet header; such that it no longer matches the evaluator-created entries (there may be a "TOE created" final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was dropped.

FCS_IPSEC_EXT.1.3

The TSF shall implement transport mode and tunnel mode.

Assurance Activity

TSS

The evaluator checks the TSS to ensure it states that the VPN can be established to operate in transport mode and/or tunnel mode (as identified in FCS_IPSEC_EXT.1.3).

Guidance

The evaluator shall confirm that the guidance documentation contains instructions on how to configure the connection in each mode selected.

Test

The evaluator shall perform the following test(s) based on the selections chosen:

a) Test 1 (conditional): If tunnel mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in tunnel mode and also configures a VPN peer to operate in tunnel mode. The evaluator configures the TOE and the VPN peer to use any of the allowable cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator shall then initiate a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

b) Test 2: The evaluator uses the guidance documentation to configure the TOE to operate in transport mode and also configures a VPN peer to operate in transport mode. The evaluator configures the TOE and the VPN peer to use any of the allowed cryptographic algorithms,

authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

FCS_IPSEC_EXT.1.4

The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms AES-CBC-128, AES-CBC-256 (both specified by RFC 3602) and no other algorithms together with a Secure Hash Algorithm (SHA)-based HMAC.

Assurance Activity

TSS

The evaluator shall examine the TSS to verify that the algorithms AES-CBC-128 and AES-CBC-256 are implemented. If the ST author has selected either AES-GCM-128 or AES-GCM-256 in the requirement, then the evaluator verifies the TSS describes these as well. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1(4) Cryptographic Operations (for keyed-hash message authentication).

Guidance

The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms, and if either AES-GCM-128 or AES-GCM-256 have been selected the guidance instructs how to use these as well.

Test

The evaluator shall configure the TOE as indicated in the guidance documentation configuring the TOE to use each of the supported algorithms, attempt to establish a connection using ESP, and verify that the attempt succeeds.

FCS_IPSEC_EXT.1.5

The TSF shall implement the protocol:

- IKEv1, using Main Mode for Phase 1 exchanges, as defined in RFCs 2407, 2408, 2409, RFC 4109, no other RFCs for extended sequence numbers and RFC 4868 for hash functions.

Application Note 53

If the TOE implements SHA-2 hash algorithms for IKEv1 or IKEv2, the ST author selects RFC 4868. If the ST author selects IKEv1, FCS_IPSEC_EXT.1.15 must also be included in the ST. IKEv2 will be required for those TOEs entering evaluation after Quarter 3, 2016.

ST Author Note

Though IKEv1 has been selected, [NDcPP] does not define FCS_IPSEC_EXT.1.15.

Assurance Activity

TSS

The evaluator shall examine the TSS to verify that IKEv1 and/or IKEv2 are implemented.

For IKEv1 implementations, the evaluator shall examine the TSS to ensure that, in the description of the IPsec protocol, it states that aggressive mode is not used for IKEv1 Phase 1 exchanges, and that only main mode is used. It may be that this is a configurable option.

Guidance

The evaluator shall check the guidance documentation to ensure it instructs the administrator how to configure the TOE to use IKEv1 and/or IKEv2 (as selected), and uses the guidance to configure the TOE to perform NAT traversal for the following test (if selected).

If the IKEv1 Phase 1 mode requires configuration of the TOE prior to its operation, the evaluator shall check the guidance documentation to ensure that instructions for this configuration are contained within that guidance.

Test

Tests are performed in conjunction with the other IPsec evaluation activities.

(conditional): The evaluator shall configure the TOE as indicated in the guidance documentation, and attempt to establish a connection using an IKEv1 Phase 1 connection in aggressive mode. This attempt should fail. The evaluator should then show that main mode exchanges are supported.

(conditional): The evaluator shall configure the TOE so that it will perform NAT traversal processing as described in the TSS and RFC 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

FCS_IPSEC_EXT.1.6

The TSF shall ensure the encrypted payload in the IKEv1 protocol uses the cryptographic algorithms AES-CBC-128, AES-CBC-256 as specified in RFC 3602 and no other algorithm.

Application Note 54

AES-GCM-128 and AES-GCM-256 may only be selected if IKEv2 is also selected, as there is no RFC defining AES-GCM for IKEv1.

Assurance Activity

TSS

The evaluator shall ensure the TSS identifies the algorithms used for encrypting the IKEv1 and/or IKEv2 payload, and that the algorithms AES-CBC-128, AES-CBC-256 are specified, and if others are chosen in the selection of the requirement, those are included in the TSS discussion.

Guidance

The evaluator ensures that the guidance documentation describes the configuration of the mandated algorithms, as well as any additional algorithms selected in the requirement. The guidance is then used to configure the TOE to perform the following test for each ciphersuite selected.

Test

The evaluator shall configure the TOE to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

FCS_IPSEC_EXT.1.7

The TSF shall ensure that:

- IKEv1 Phase 1 SA lifetimes can be configured by an Security Administrator based on length of time, where the time values can configured within **1- 504** hours.

Application Note 55

The ST author chooses either the IKEv1 requirements or IKEv2 requirements (or both, depending on the selection in FCS_IPSEC_EXT.1.5). The ST author chooses either volume-based lifetimes or time-based lifetimes (or a combination). This requirement must be accomplished by providing Security Administrator-configurable lifetimes (with appropriate instructions in documents mandated by AGD_OPE). Hardcoded limits do not meet this requirement. In general, instructions for setting the parameters of the implementation, including lifetime of the SAs, should be included in the guidance documentation generated for AGD_OPE.

Assurance Activity

TSS

The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 1 SA lifetime and/or the IKEv2 SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Guidance

The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, the evaluator ensures that the Administrator is able to configure Phase 1 SA values for 24 hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Test

When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1 (Conditional): The evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.
b) Test 2 (Conditional): The evaluator shall configure a maximum lifetime of 24 hours for the Phase 1 SA following the guidance documentation. The evaluator shall configure a test peer with a lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, maintain the Phase 1 SA for 24 hours, and determine that once 24 hours has

elapsed, a new Phase 1 SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.

FCS_IPSEC_EXT.1.8

The TSF shall ensure that:

- IKEv1 Phase 2 SA lifetimes can be configured by a Security Administrator based on:
  - number of bytes;
  - length of time, where the time values can be configured within **1-504** hours.

Application Note 56

The ST author chooses either the IKEv1 requirements or IKEv2 requirements (or both, depending on the selection in FCS_IPSEC_EXT.1.5). The ST author chooses either volume-based lifetimes or time-based lifetimes (or a combination). This requirement must be accomplished by providing Security Administrator-configurable lifetimes (with appropriate instructions in documents mandated by AGD_OPE). Hardcoded limits do not meet this requirement. In general, instructions for setting the parameters of the implementation, including lifetime of the SAs, should be included in the guidance documentation generated for AGD_OPE.

Assurance Activity

TSS

The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 2 SA lifetime and/or the IKEv2 Child SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.3.

Guidance

The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, the evaluator ensures that the Administrator is able to configure Phase 2 SA values for 8hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Test

When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1 (Conditional): The evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA

between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

b) Test 2 (Conditional): The evaluator shall configure a maximum lifetime of 8 hours for the Phase 2 SA following the guidance documentation. The evaluator shall configure a test peer with a lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, maintain the Phase 1 SA for 8 hours, and determine that once 8 hours has elapsed, a new Phase 2SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

FCS_IPSEC_EXT.1.9

The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange ("x" in g^x mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least **224, 256 or 384** bits.

Application Note 57

For DH groups 19 and 20, the "x" value is the point multiplier for the generator point G.

Since the implementation may allow different Diffie-Hellman groups to be negotiated for use in forming the SAs, the assignment in FCS_IPSEC_EXT.1.9 may contain multiple values. For each DH group supported, the ST author consults Table 2 in NIST SP 800-57 "Recommendation for Key Management – Part 1: General" to determine the security strength ("bits of security") associated with the DH group. Each unique value is then used to fill in the assignment for this element. For example, suppose the implementation supports DH group 14 (2048-bit MODP) and group 20 (ECDH using NIST curve P-384). From Table 2, the bits of security value for group 14 is 112, and for group 20 it is 192.

Assurance Activity

TSS

The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating "x". The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of "x" meets the stipulations in the requirement.

FCS_IPSEC_EXT.1.10

The TSF shall generate nonces used in IKEv1 exchanges of length **at least 112, 128 or 192 bits**.

Application Note 58

The ST author must select the second option for nonce lengths if IKEv2 is also selected (as this is mandated in RFC 5996). The ST author may select either option for IKEv1. For the first option for nonce lengths, since the implementation may allow different Diffie-Hellman groups to be negotiated for use in forming the SAs, the assignment in FCS_IPSEC_EXT.1.10 may contain multiple values. For each DH group supported, the ST author consults Table 2 in NIST SP 800-57 "Recommendation for Key Management–Part 1: General" to determine the security strength ("bits of security") associated with the DH group. Each unique value is then used to fill in the assignment for this element. For example, suppose the implementation supports DH group 14 (2048-bit MODP) and group 20 (ECDH using NIST curve P-384). From Table 2, the bits of security value for group 14 is 112, and for group 20 it is 192.

Because nonces may be exchanged before the DH group is negotiated, the nonce used should be large enough to support all TOE-chosen proposals in the exchange.

Assurance Activity

Test

(conditional) If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

(conditional) If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

FCS_IPSEC_EXT.1.11

The TSF shall ensure that all IKE protocols implement DH Groups 14 (2048-bit MODP), and 19 (256-bit Random ECP), 20 (384-bit Random ECP), no other DH groups.

Application Note 59

The selection is used to specify additional DH groups supported. This applies to IKEv1 and IKEv2 exchanges. For products entering into evaluation after Quarter 3, 2015, DH Group 19 (256-bit Random ECP) and DH Group 20 (384-bit Random ECP) will be required. It should be noted that if any additional DH groups are specified, they must comply with the requirements (in terms of the ephemeral keys that are established) listed in FCS_CKM.1.

Assurance Activity

TSS

The evaluator shall check to ensure that the DH groups specified in the requirement are listed as being supported in the TSS. If there is more than one DH group supported, the evaluator checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Guidance

The evaluator ensures that the guidance documentation describes the configuration of the mandated algorithms, as well as any additional algorithms selected in the requirement. The guidance is then used to configure the TOE to perform the following test for each ciphersuite selected.

Test

For each supported DH group, the evaluator shall test to ensure that all supported IKE protocols can be successfully completed using that particular DH group.

FCS_IPSEC_EXT.1.12

The TSF shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the IKEv1 Phase 1 connection is greater than or equal to the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the IKEv1 Phase 2 connection.

Application Note 60

The ST author chooses either or both of the IKE selections based on what is implemented by the TOE. Obviously, the IKE version(s) chosen should be consistent not only in this element, but with other choices for other elements in this component. While it is acceptable for this capability to be configurable, the

default configuration in the evaluated configuration (either "out of the box" or by configuration guidance in the AGD documentation) must enable this functionality.

Assurance Activity

TSS

The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges. The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

Test

The evaluator simply follows the guidance to configure the TOE to perform the following tests.

a) Test 1: This test shall be performed for each version of IKE supported. The evaluator shall successfully negotiate an IPsec connection using each of the supported algorithms and hash functions identified in the requirements.
b) Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP that selects an encryption algorithm with more strength than that being used for the IKE SA (i.e., symmetric algorithm with a key size larger than that being used for the IKE SA). Such attempts should fail.
c) Test 3: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an IKE SA using an algorithm that is not one of the supported algorithms and hash functions identified in the requirements. Such an attempt should fail.
d) Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP (assumes the proper parameters where used to establish the IKE SA) that selects an encryption algorithm that is not identified in FCS_IPSEC_EXT.1.4. Such an attempt should fail.

FCS_IPSEC_EXT.1.13

The TSF shall ensure that all IKE protocols perform peer authentication using RSA, ECDSA that use X.509v3 certificates that conform to RFC 4945 and Pre-shared Keys.

Application Note 61

At least one public-key-based Peer Authentication method is required in order to conform to this PP; one or more of the public key schemes is chosen by the ST author to reflect what is implemented. The ST author also ensures that appropriate FCS requirements reflecting the algorithms used (and key generation capabilities, if provided) are listed to support those methods. Note that the TSS will elaborate on the way in which these algorithms are to be used (for example, RFC 2409 specifies three authentication methods using public keys; each one supported will be described in the TSS). Peer authentication using ECDSA X.509v3 certificates will be required for TOEs entering evaluation after Quarter 3, 2015.

Assurance Activity

TSS

The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1(2) Cryptographic Operations (for cryptographic signature).

If pre-shared keys are chosen in the selection, the evaluator shall check to ensure that the TSS describes how pre-shared keys are established and used in authentication of IPsec connections. The description in the TSS shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Guidance

The evaluator ensures the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

The evaluator shall check that the guidance documentation describes how pre-shared keys are to be generated and established. The description in the guidance documentation shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

In order to construct the environment and configure the TOE for the following tests, the evaluator will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA, and ensure a valid certificate for that CA is loaded into the TOE and marked “trusted”.

Test

For efficiency sake, the testing that is performed may be combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1. The following tests shall be repeated for each peer authentication selected in the FCS_IPSEC_EXT.1.1 selection above:

a) Test 1: The evaluator shall configure the TOE to use a private key and associated certificate signed by a trusted CA and shall establish an IPsec connection with the peer.
b) Test 2 [conditional]: The evaluator shall generate a pre-shared key off-TOE and use it, as indicated in the guidance documentation, to establish an IPsec connection with the peer.

FCS_IPSEC_EXT.1.14

The TSF shall only establish a trusted channel to peers with valid certificates.

Application Note 62

Supported peer certificate algorithms are the same as FCS_IPSEC_EXT.1.1.

Assurance Activity

TSS

The evaluator shall verify that the TSS describes how the DN in the certificate is compared to the expected DN.

Guidance

The evaluator shall ensure that the guidance documentation includes configuration of the expected DN for the connection.

Test

The evaluator shall, if necessary, configure the expected DN according to the guidance documentation. The evaluator shall send a peer certificate signed by a trusted CA with a DN that does not match an expected DN and verify that the TOE denies the connection.

6.1.2.9 FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1

The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using Hash_DRBG (any).

FCS_RBG_EXT.1.2

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

Application Note 13

For the first selection in FCS_RBG_EXT.1.2, the ST selects at least one of the types of noise sources. If the TOE contains multiple noise sources of the same type, the ST author fills the assignment with the appropriate number for each type of source (e.g., 2 software-based noise sources, 1 hardware-based noise source). The documentation and tests required in the Evaluation Activity for this element necessarily cover each source indicated in the ST.

ISO/IEC 18031:2011 contains three 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, and include the specific underlying cryptographic primitives used in the requirement. 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 CTR_DRBG are allowed.

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.

Assurance Activity

Documentation shall be produced—and the evaluator shall perform the activities—in accordance with Appendix D of [NDcPP].

Test

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

If the RNG 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 SP800-90A).

If the RNG 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.

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

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

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.

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

6.1.2.10 FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1

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

Application Note 69

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). RFC 4253 indicates that certain cryptographic algorithms are "REQUIRED". This means that the implementation must include support, not that the algorithms must be enabled for use. Ensuring that algorithms indicated as "REQUIRED" but not listed in the later elements of this component are implemented is out of scope of the assurance activity for this requirement.

FCS_SSHS_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

TSS

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_SSHS_EXT.1.5, and ensure that password-based authentication methods are also allowed.

Test

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 guidance documentation.

Test 2: The evaluator shall choose one public key algorithm supported by the TOE. The evaluator shall generate a new key pair for that algorithm without configuring the TOE to recognize the public key for authentication. The evaluator shall use an SSH client to attempt to connect to the TOE with the new key pair and demonstrate that authentication fails.

Test 3: Using the guidance documentation, 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.

Test 4: The evaluator shall use an SSH client, enter an incorrect password to attempt to authenticate to the TOE, and demonstrate that the authentication fails.

FCS_SSHS_EXT.1.3

The TSF shall ensure that, as described in RFC 4253, packets greater than **4096** bytes in an SSH transport connection are dropped.

Application Note 70

RFC 4253 provides for the acceptance of "large packets" with the caveat that the 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

TSS

The evaluator shall check that the TSS describes how "large packets" in terms of RFC 4253 are detected and handled.

Test

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

FCS_SSHS_EXT.1.4

The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: aes128-cbc, aes256-cbc, no other algorithms.

Application Note 71

RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as encryption algorithms when the same algorithm is being used as the MAC algorithm. 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.

Assurance Activity

TSS

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.

Guidance

The evaluator shall also check the guidance documentation 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).

Test

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.

Test 2: The evaluator shall configure an SSH client to only allow the 3des-cbc encryption algorithm and no other encryption algorithms. The evaluator shall attempt to establish an SSH connection from the SSH client to the TOE and observe that the connection is rejected.

FCS_SSHS_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) and rejects all other public key algorithms.

Application Note 72

Implementations that select only ssh-rsa will not achieve the 112-bit security strength in the digital signature generation for SSH authentication as is recommended in NIST SP 800-131A. Future versions of this profile may remove ssh-rsa as a selection.

Assurance Activity

TSS

The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics are specified, and the public key algorithms supported are specified as well. The evaluator shall check the TSS to ensure that the public key algorithms specified are identical to those listed for this component.

Guidance

The evaluator shall also check the guidance documentation 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).

Test

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

Test 2: The evaluator shall configure an SSH client to only allow the ssh-dsa public key algorithm and no other public key algorithms. The evaluator shall attempt to establish an SSH connection from the SSH client to the TOE and observe that the connection is rejected.

FCS_SSHS_EXT.1.6

The TSF shall ensure that the SSH transport implementation uses hmac-sha1 and no other MAC algorithms as its MAC algorithm(s) and rejects all other MAC algorithm(s).

Application Note 73

RFC 5647 specifies the use of the AEAD_AES_128_GCM and AEAD_AES_256_GCM algorithms in SSH. As described in RFC 5647, AEAD_AES_128_GCM and AEAD_AES_256_GCM can only be chosen as MAC algorithms when the same algorithm is being used as the encryption algorithm. RFC 6668 specifies the use of the sha2 algorithms in SSH.

Assurance Activity

TSS

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.

Guidance

The evaluator shall also check the guidance documentation 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).

Test

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.

Test 2: The evaluator shall configure an SSH client to only allow the "none" MAC algorithm. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

Test 3: The evaluator shall configure an SSH client to only allow the hmac-md5 MAC algorithm. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

FCS_SSHS_EXT.1.7

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

Assurance Activity

Guidance

The evaluator shall also check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections with the TOE.

Test

Test 1: The evaluator shall configure an SSH client to only allow the diffie-hellman-group1-sha1 key exchange. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

Test 2: For each allowed key exchange method, the evaluator shall configure an SSH client to only allow that method for key exchange, attempt to connect from the client to the TOE, and observe that the attempt succeeds.

FCS_SSHS_EXT.1.8

The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

Assurance Activity

Test

The evaluator shall configure the TOE to create a log entry when a rekey occurs. The evaluator shall connect to the TOE with an SSH client and cause 2^28 packets to be transmitted from the client to the TOE, and subsequently review the audit log to ensure that a rekey occurred.

6.1.3 Class FIA: Identification and Authentication

6.1.3.1 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1

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

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: “!”, “%”, “&”, “*”, “(“, “)”, "+", ":", ";", "<", ">", "?";

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

Application Note 14

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, over protocols that support passwords, such as SSH and HTTPS, or to grant configuration data that supports other SFRs in the Security Target.

Assurance Activity

Guidance

The evaluator shall examine the guidance documentation 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.

Test

The evaluator shall perform the following tests.

a) 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.

6.1.3.2 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1.1

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

- Display the warning banner in accordance with FTA_TAB.1;
- **ARP, ICMP, routing services, BFD send, DHCP service, SSH, IPDV (port UDP/49402), RSVP (port UDP/1698), NTP (port UDP/123).**

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.

Application Note 15

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 (such as 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

TSS

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".

Guidance

The evaluator shall examine the guidance documentation to determine that any necessary reparatory 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 guidance documentation 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 guidance documentation provides sufficient instruction on limiting the allowed services.

Test

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:

a) Test 1: The evaluator shall use the guidance documentation 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.
b) Test 2: The evaluator shall configure the services allowed (if any) according to the guidance documentation, 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.
c) 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.

6.1.3.3 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1

The TSF shall provide a local password-based authentication mechanism, **RADIUS** to perform administrative user authentication.

Application Note 16

The assignment should be used to identify any additional local authentication mechanisms supported. Local authentication mechanisms are defined as those that occur through the local console; remote administrative sessions (and their associated authentication mechanisms) are specified in FTP_TRP.1.

Assurance Activity

Evaluation 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.

6.1.3.4 FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1

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

Application Note 17

"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

Test

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

a) 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.

6.1.3.5 FIA_X509_EXT.1 X.509 Certificate Validation

FIA_X509_EXT.1.1

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

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates.
- The TSF shall validate the revocation status of the certificate using a Certificate Revocation List (CRL) as specified in RFC 5759.
- The TSF shall validate the extendedKeyUsage field according to the following rules:

- Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.
- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
- OCSP Certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

Application Note 18

FIA_X509_EXT.1.1 lists the rules for validating certificates. The ST author selects whether revocation status is verified using OCSP or CRLs. The trusted channel/path protocols require that certificates are used; this use requires that the extendedKeyUsage rules are verified.

The validation is expected to end in a trusted root CA certificate in a root store managed by the platform.

FIA_X509_EXT.1.2

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

Application Note 19

This requirement applies to certificates that are used and processed by the TSF and restricts the certificates that may be added as trusted CA certificates.

Assurance Activity

TSS

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place. The evaluator ensures the TSS also provides a description of the certificate path validation algorithm.

Test

The evaluator shall perform the following tests for FIA_X509_EXT.1.1:

a) Test 1: The evaluator shall demonstrate that validating a certificate without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate to be used in the function, and demonstrate that the function succeeds. The evaluator shall then delete one of the certificates, and show that the function fails.
b) Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.
c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-– conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the TOE certificate and revocation of the TOE intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails.

d) Test 4: If OCSP is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set, and verify that validation of the CRL fails.
e) Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)
f) Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)
g) Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The hash of the certificate will not validate.)

The evaluator shall perform the following tests for FIA_X509_EXT.1.2. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules.

The evaluator shall create a chain of at least four certificates: the node certificate to be tested, two intermediate CAs, and the self-signed Root CA.

a) Test 1: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate does not contain the basicConstraints extension. The validation of the certificate path fails.
b) Test 2: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension set to FALSE. The validation of the certificate path fails.
c) Test 3: The evaluator shall construct a certificate path, such that the certificate of the CA issuing the TOE's certificate has the cA flag in the basicConstraints extension set to TRUE. The validation of the certificate path succeeds.

6.1.3.6 FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1

The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for IPsec and no additional uses.

Application Note 20

The ST author's selection matches the selection of FTP_ITC.1.1. Certificates may optionally be used for trusted updates of system software (FPT_TUD_EXT.1) and for integrity verification (FPT_TST_EXT.2).

FIA_X509_EXT.2.2

When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall accept the certificate.

Application Note 21

Often a connection must be established to check the revocation status of a certificate -either to download a CRL or to perform a lookup using OCSP. The selection is used to describe the behavior in the event that such a connection cannot be established (for example, due to a network error). If the TOE has determined the certificate valid according to all other rules in FIA_X509_EXT.1, the behavior indicated in the selection determines the validity. The TOE must not accept the certificate if it fails any of the other validation rules

in FIA_X509_EXT.1. If the administrator-configured option is selected by the ST Author, the ST Author also selects the corresponding function in FMT_SMF.1.

Assurance Activity

TSS

The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

The evaluator shall examine the TSS to confirm that it describes the behaviour of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the guidance documentation contains instructions on how this configuration action is performed.

Test

The evaluator shall perform the following test for each trusted channel:

The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2is performed. If the selected action is administrator-configurable, then the evaluator shall follow the guidance documentation to determine that all supported administrator-configurable options behave in their documented manner.

6.1.3.7 FIA_X509_EXT.3 X.509 Certificate Requests

FIA_X509_EXT.3.1

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

Application Note 22

The public key is the public key portion of the public-private key pair generated by the TOE as specified in FCS_CKM.1(1).

FIA_X509_EXT.3.2

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

Assurance Activity

TSS

If the ST author selects "device-specific information", the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

Guidance

The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate Request Message. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this

guidance includes instructions for establishing these fields before creating the certificate request message.

Test

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall use the guidance documentation to cause the TOE to generate a certificate request message. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the certificate request provides the public key and other required information, including any necessary user-input information.
b) Test 2: The evaluator shall demonstrate that validating a certificate response message without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message, and demonstrate that the function succeeds. The evaluator shall then delete one of the certificates, and show that the function fails

6.1.4 Class FMT: Security Management

6.1.4.1 FMT_MOF.1(1)/TrustedUpdate Management of Security Functions Behaviour

FMT_MOF.1.1(1)/TrustedUpdate

The TSF shall restrict the ability to enable the functions to perform manual update to Security Administrators.

Application Note 23

FMT_MOF.1(1)/TrustedUpdate restricts the initiation of manual updates to Security Administrators.

Assurance Activity

Test

The evaluator shall try to perform the update using a legitimate update image without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all –depending on the configuration of the TOE). This test should fail.

The evaluator shall try to perform the update with prior authentication as security administrator using a legitimate update image. This test should pass. This test case should be covered by the tests for FPT_TUD_EXT.1 already.

6.1.4.2 FMT_MOF.1(1)/Audit Management of Security Functions Behaviour

FMT_MOF.1.1(1)/Audit

The TSF shall restrict the ability to determine the behaviour of, modify the behaviour of the functions transmission of audit data to an external IT entity to Security Administrators.

Application Note 43

FMT_MOF.1(1)/Audit should always be chosen if the transmission protocol for transmission of audit data to an external IT entity as defined in FAU_STG_EXT.1.1 is configurable.

Assurance Activity

Test

The evaluator shall try to modify all parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). This test should fail.

The evaluator shall try to modify all parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity with prior authentication as security administrator. The effects of the modifications should be confirmed.

The evaluator does not necessarily have to test all possible values of all parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity but at least one allowed value per configurable parameter.

6.1.4.3 FMT_MOF.1(2)/Audit Management of Security Functions Behaviour

FMT_MOF.1.1(2)/Audit

The TSF shall restrict the ability to determine the behaviour of, modify the behaviour of the functions handling of audit data to Security Administrators.

Application Note 44

FMT_MOF.1(2)/Audit should only be chosen if the handling of audit data is configurable. The term 'handling of audit data' refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2.

Assurance Activity

Test

The evaluator shall try to modify all parameters for configuration of the handling of audit data without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all –depending on the configuration of the TOE). This test should fail. The term 'handling of audit data' refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2.

The evaluator shall try to modify all parameters for configuration of the handling of audit data with prior authentication as security administrator. The effects of the modifications should be confirmed. The term 'handling of audit data' refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2.

The evaluator does not necessarily have to test all possible values of all parameters for configuration of the handling of audit data but at least one allowed value per configurable parameter.

6.1.4.4 FMT_MOF.1(1)/AdminAct Management of Security Behaviour

FMT_MOF.1.1(1)/AdminAct

The TSF shall restrict the ability to modify the behaviour of the functions TOE Security Functions to Security Administrators.

Application Note 45

FMT_MOF.1(1)/AdminAct should only be chosen if the behaviour of the TOE Security Functions is configurable.

Assurance Activity

Test

The evaluator shall try to perform at least one of the related actions without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). These attempts should fail.

The evaluator shall try to perform at least one of the related actions with prior authentication as security administrator. These attempts should succeed.

6.1.4.5 FMT_MOF.1(2)/AdminAct Management of Security Behaviour

FMT_MOF.1.1(2)/AdminAct

The TSF shall restrict the ability to enable, disable the functions services to Security Administrators.

Application Note 46

FMT_MOF.1(2)/AdminAct should only be chosen if the Security Administrator has the ability to start and stop services.

Assurance Activity

Test

The evaluator shall try to perform at least one of the related actions without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). These attempts should fail.

The evaluator shall try to perform at least one of the related actions with prior authentication as security administrator. These attempts should succeed.

6.1.4.6 FMT_MTD.1 Management of TSF Data

FMT_MTD.1.1

The TSF shall restrict the ability to manage the TSF Data to Security Administrators.

Application Note 24

The word "manage" includes but is not limited to create, initialize, view, change default, modify, delete, clear, and append. This SFR includes also the resetting of user passwords by the Security Administrator.

Assurance Activity

TSS

The evaluator shall examine the TSS to determine that, for each administrative function identified in the guidance documentation; 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.

Guidance

The evaluator shall review the guidance documentation to determine that each of the TSF-data-manipulating functions implemented in response to the requirements of the cPP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

6.1.4.7 FMT_MTD.1/AdminAct Management of TSF Data

FMT_MTD.1.1/AdminAct

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

Application Note 48

FMT_MTD.1.1/AdminAct should only be chosen if cryptographic keys can be modified, deleted or generated/imported by the Security Administrator.

Assurance Activity

Test

The evaluator shall try to perform at least one of the related actions without prior authentication as security administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). This test should fail.

The evaluator shall try to perform at least one of the related actions with prior authentication as security administrator. This test should pass.

6.1.4.8 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

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

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- Ability to configure audit behavior;
- Ability to configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1;
- Ability to configure the cryptographic functionality;

Application Note 25

The TOE must provide functionality for both local and remote administration, including the ability to configure the access banner for FTA_TAB.1 and the session inactivity time(s) for FTA_SSL.3 & FTA_SSL.4. The item "Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates" includes the relevant management functions from FMT_MOF.1(1)/TrustedUpdate, FMT_MOF.1(2)/TrustedUpdate (if included in the ST), FIA_X509_EXT.2.2 and FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action). Similarly, the selection "Ability to configure audit behavior" includes the relevant management functions from FMT_MOF.1(1)/Audit, FMT_MOF.1(2)/Audit, FMT_MOF.1.1(1)/AdminAct, FMT_MOF.1.1(2)/AdminAct and FMT_MOF.1/LocSpace (for all of these SFRs that are included in the ST). If the TOE offers the ability for the administrator to configure the audit behaviour, 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 cPP and are included as part of the requirements in FTA_TAB.1, FTA_SSL.3, FTA_SSL.4, FMT_MOF.1(1)/TrustedUpdate, FMT_MOF.1(2)/TrustedUpdate (if included in the ST), IA_X509_EXT.2.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1(1)/Audit, FMT_MOF.1(2)/Audit, FMT_MOF.1.1(1)/AdminAct, FMT_MOF.1.1(2)/AdminAct and FMT_MOF.1/LocSpace (for all of these SFRs that are included in the ST), 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

6.1.4.9 FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1

The TSF shall maintain the roles:

- Security Administrator.

FMT_SMR.2.2

The TSF shall be able to associate users with roles.

FMT_SMR.2.3

The TSF shall ensure that the conditions:

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

are satisfied.

Application Note 26

FMT_SMR.2.3 requires that a Security Administrator be able to administer the TOE through the local console and through a remote mechanism (IPsec, SSH, TLS, HTTPS).

Assurance Activity

Guidance

The evaluator shall review the guidance documentation 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.

Test

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 this cPP 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.

6.1.5 Class FPT: Protection of the TSF

6.1.5.1 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.

Application Note 28

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.

Assurance Activity

TSS

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.

6.1.5.2 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.

Application Note 27

The intent of this requirement is for the device to protect keys, key material, and authentication credentials from unauthorized disclosure. This data should only be accessed for the purposes of their assigned security functionality, and there is no need for them to be displayed/accessed at any other time. This requirement does not prevent the device from providing indication that these exist, are in use, or are still valid. It does, however, restrict the reading of the values outright.

Assurance Activity

TSS

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.

6.1.5.3 FPT_TST_EXT.1(1) TSF Testing (on power on)

FPT_TST_EXT.1.1(1)

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

- **Firmware Integrity 16 bit CRC performed over all code in flash**

- **AES - Hardware implementation KATs: Encryption, Decryption, Modes: CBC, Key sizes:128 bits**
- **AES - Firmware implementation KATs: Encryption, Decryption, Modes: CBC Key sizes: 128, 256 bits**
- **DRBG (KATs: Hash DRBG)**
- **HMAC - Hardware implementation (KATs: Generation, verification, SHA-1) HMAC - Firmware implementation (KATs: Generation, verification, SHA-1, SHA-256, SHA-384)**
- **RSA KATs: Signature Generation, Signature Verification, Key:2048 bits**
- **ECDSA KATs: Signature Generation, Signature Verification, NIST curves: P-256, P-384**
- **SHA KAT: SHA-1, SHA-256, SHA-384**

Application Note 29

It is expected that self-tests are carried out during initial start-up (on power on). Other options should only be used if the developer can justify why they are not carried out during initial start-up. It is expected that at least self-tests for verification of the integrity of the firmware and software as well as for the correct operation of cryptographic functions necessary to fulfil the SFRs will be performed. If not all self-test are performed during start-up multiple iterations of this SFR are used with the appropriate options selected. In future versions of this cPP the suite of self-tests will be required to contain at least mechanisms for measured boot including self-tests of the components which perform the measurement.

Application Note 30

If certificates are used by the self-test mechanism (e.g. for verification of signatures for integrity verification), certificates are validated in accordance with FIA_X509_EXT.1 and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TST_EXT.2 must be included in the ST.

Assurance Activity

TSS

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF; 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.

Guidance

The evaluator shall also ensure that the guidance documentation 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.

Test

Future versions of this cPP will mandate a clearly defined minimum set of self tests. But also for this version of the cPP it is expected that at least the following tests are performed:

a) Verification of the integrity of the firmware and executable software of the TOE
b) Verification of the correct operation of the cryptographic functions necessary to fulfill any of the SFRs.

Although formal compliance is not mandated, the self tests performed should aim for a level of confidence comparable to:

a) FIPS 140-2, chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software.
b) FIPS 140-2, chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

The evaluator shall verify that the self tests described above are either carried out during initial start-up and that the developer has justified any deviation from this (if applicable).

6.1.5.4 FPT_TST_EXT.1(2) TSF Testing (conditional)

FPT_TST_EXT.1.1(2)

The TSF shall run a suite of the following self-tests at the conditions **listed in Table 12: Conditional Self-Tests** to demonstrate the correct operation of the TSF: **self-tests listed in Table 12: Conditional Self-Tests.**

Table 12: Conditional Self-Tests	
Condition	Test
A random value is requested from the NDRNG	NDRNG Continuous Test
A random value is requested from the DRBG	DRBG Continuous Test
Firmware is loaded	RSA 2048 signature verification
Key Establishment (RSA and ECDSA)	Pair-wise consistency test
All conditions defined in FIPS SP 800-90 section 11.3	All DRBG Health Checks defined in FIPS SP 800-90, section 11.3
Alternating Bypass service is called	Bypass Test

Application Note 29

It is expected that self-tests are carried out during initial start-up (on power on). Other options should only be used if the developer can justify why they are not carried out during initial start-up. It is expected that at least self-tests for verification of the integrity of the firmware and software as well as for the correct operation of cryptographic functions necessary to fulfil the SFRs will be performed. If not all self-test are performed during start-up multiple iterations of this SFR are used with the appropriate options selected. In future versions of this cPP the suite of self-tests will be required to contain at least mechanisms for measured boot including self-tests of the components which perform the measurement.

Application Note 30

If certificates are used by the self-test mechanism (e.g. for verification of signatures for integrity verification), certificates are validated in accordance with FIA_X509_EXT.1 and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TST_EXT.2 must be included in the ST.

Assurance Activity

TSS

The evaluator shall examine the TSS to ensure that it details the self tests that are run by the TSF; 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.

Guidance

The evaluator shall also ensure that the guidance documentation 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.

Test

Future versions of this cPP will mandate a clearly defined minimum set of self tests. But also for this version of the cPP it is expected that at least the following tests are performed:

c) Verification of the integrity of the firmware and executable software of the TOE
d) Verification of the correct operation of the cryptographic functions necessary to fulfill any of the SFRs.

Although formal compliance is not mandated, the self tests performed should aim for a level of confidence comparable to:

c) FIPS 140-2, chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software.
d) FIPS 140-2, chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

The evaluator shall verify that the self tests described above are either carried out during initial start-up and that the developer has justified any deviation from this (if applicable).

6.1.5.5 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1

The TSF shall provide Security Administrators the ability to query the currently executing version of the TOE firmware/software as well as the most recently installed version of the TOE firmware/software.

Application Note 31

The version currently running (being executed) may not be the version most recently installed. For instance, maybe the update was installed but the system requires a reboot before this update will run. Therefore, it needs to be clear that the query should indicate both the most recently executed version as well as the most recently installed update.

FPT_TUD_EXT.1.2

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

Application Note 32

The selection in FPT_TUD_EXT.1.2 distinguishes the support of automatic checking for updates and support of automatic updates. The first option refers to a TOE that checks whether a new update is available, communicates this to the administrator (e.g. through a message during an administrator session, through log files) but requires some action by the administrator to actually perform the update. The second option refers to a TOE that checks for updates and automatically installs them upon availability.

FPT_TUD_EXT.1.3

The TSF shall provide means to authenticate firmware/software updates to the TOE using a digital signature mechanism prior to installing those updates.

Application Note 33

The digital signature mechanism referenced in the selection of FPT_TUD_EXT.1.3 is one of the algorithms specified in FCS_COP.1(2). The published hash referenced in FPT_TUD_EXT.1.3 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.

Application Note 34

Future versions of this cPP will mandate the use of a digital signature mechanism for trusted updates.

Application Note 35

If certificates are used by the update verification mechanism, certificates are validated in accordance with FIA_X509_EXT.1 and should be selected in FIA_X509_EXT.2.1. Additionally, FPT_TUD_EXT.2 must be included in the ST.

Application Note 36

"Update" in the context of this SFR refers to the process of replacing a non-volatile, system resident software component with another. The former is referred to as the NV image, and the latter is the update image. While the update image is typically newer than the NV image, this is not a requirement. There are legitimate cases where the system owner may want to rollback a component to an older version (e.g. when the component manufacturer releases a faulty update, or when the system relies on an undocumented feature no longer present in the update). Likewise, the owner may want to update with the same version as the NV image to recover from faulty storage.

All discrete software components (e.g. applications, drivers, kernel, firmware) of the TSF, should be digitally signed by the corresponding manufacturer and subsequently verified by the mechanism performing the update. Since it is recognized that components may be signed by different manufacturers, it is essential that the update process verify that both the update and NV images were produced by the same manufacturer (e.g. by comparing public keys) or signed by legitimate signing keys (e.g. successful verification of certificates when using X.509 certificates).

Assurance Activity

TSS

The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system software. The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. Alternatively an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the TSS contains a description of how the certificates are contained on the device. The evaluator also ensures that the TSS (or guidance documentation) describes how the certificates are installed/updated/selected, if necessary.

Guidance

The evaluator shall verify that the guidance documentation describes how the verification of the authenticity of the update is performed (digital signature verification or verification of published hash). The description shall include the procedures for successful and unsuccessful verification. The description shall correspond to the description in the TSS.

Test

The evaluator shall perform the following tests:

a) Test 1: The evaluator performs the version verification activity to determine the current version of the product as well as the most recently installed version (should be the same version before updating). The evaluator obtains a legitimate update using procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ('activation' could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

b) Test 2: The evaluator performs the version verification activity to determine the current version of the product as well as the most recently installed version (should be the same version before updating). The evaluator obtains or produces illegitimate updates as defined below, and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:
   1) A modified version (e.g. using a hex editor) of a legitimately signed update (if digital signatures are used) or a version that does not match the published hash (if published hashes are used)
   2) An image that has not been signed (if digital signatures are used) or an image without published hash (if published hashes are used)
   3) An image signed with an invalid signature (e.g. by using a different key as expected for creating the signature or by manual modification of a legitimate signature) (only if digital signatures are used)
   4) The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

The evaluator shall perform the Tests 1 and 2 for all methods supported (manual updates, automatic checking for updates, automatic updates).

6.1.5.6 FPT_STM.1 Reliable Time Stamps

FPT_STM.1.1

The TSF shall be able to provide reliable time stamps.

Application Note 37

The TSF does not provide reliable information about the current time at the TOE's location by itself, but depends on external time and date information, either provided manually by the administrator or through the use of an NTP server. The term 'reliable time stamps' refers to the strict use of the time and date information, that is provided externally, and the logging of all changes to the time settings including information about the old and new time. With this information the real time for all audit data can be calculated.

Assurance Activity

TSS

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 guidance documentation to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the guidance documentation 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.

Test

The evaluator shall perform the following tests:

a) Test 1: The evaluator uses the guidance documentation to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

b) Test 2: If the TOE supports the use of an NTP server; the evaluator shall use the guidance documentation 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 guidance documentation.

If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

6.1.6 Class FTA: TOE Access

6.1.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1.1

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

Assurance Activity

Test

The evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation 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.

6.1.6.2 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

Test

The evaluator shall perform the following test:

a) Test 1: The evaluator follows the guidance documentation 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.

6.1.6.3 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

Test

The evaluator shall perform the following tests:

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

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

6.1.6.4 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.

Application Note 38

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

TSS

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).

Test

The evaluator shall also perform the following test:

a) Test 1: The evaluator follows the guidance documentation 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.

6.1.7 Class FTP: Trusted Path/Channels

6.1.7.1 FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1

The TSF shall be capable of using IPsec to provide a trusted communication channel between itself and authorized IT entities supporting the following capabilities: audit server, authentication server, **NTP 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, RADIUS, NTP**.

Application Note 39

The intent of the above requirement is to provide a means by which a cryptographic protocol may be used to protect external communications with authorized IT entities that the TOE interacts with to perform its functions. The TOE uses at least one of the listed protocols 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 capable of being protected by one of the listed protocols. 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). The ST author selects the mechanism or mechanisms supported by the TOE, and then ensures that the detailed protocol requirements in Appendix B corresponding to their selection are included in the ST. If TLS is selected, the ST author will claim FCS_TLSC_EXT.2 instead of FCS_TLSC_EXT.1.

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

TSS

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.

Guidance

The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

Test

The evaluator shall perform the following tests:

a) 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 guidance documentation and ensuring that communication is successful.

b) Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be initiated from the TOE.

c) Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

d) 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.

6.1.7.2 FTP_TRP.1 Trusted Path

FTP_TRP.1.1

The TSF shall be capable of using SSH to provide a communication path between itself and authorized remote administrators that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from disclosure and provides detection of modification of the channel data.

FTP_TRP.1.2

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

FTP_TRP.1.3

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

Application Note 40

This requirement ensures that authorized remote administrators initiate all communication with the TOE via a trusted path, and that all communication with the TOE by remote administrators is performed over this path. The data passed in this trusted communication channel are encrypted as defined by the protocol

chosen in the first selection. The ST author selects the mechanism or mechanisms supported by the TOE, and then ensures that the detailed protocol requirements in Appendix B corresponding to their selection are included in the ST.

Assurance Activity

TSS

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 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.

Guidance

The evaluator shall confirm that the guidance documentation contains instructions for establishing the remote administrative sessions for each supported method.

Test

The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each specified (in the guidance documentation) remote administration method is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

b) Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be initiated from the TOE.

c) Test 3: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

d) Test 4: The evaluators shall ensure that, 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.

6.2 Security Assurance Requirements

This Security Target is conformant with the assurance requirements specified in the cPP.

Table 13: Assurance Requirements

Assurance Class	Assurance Component
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Stated security requirements (ASE_REQ.1)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1)

	Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Labeling of the TOE (ALC_CMC.1)
	TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing –sample (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

6.2.1 Extended Security Assurance Requirements

These requirements are taken directly from the cPP.

6.2.1.1 ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM. In addition, there may be Evaluation Activities specified within the SD that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

Appendix D provides a description of the information expected to be provided regarding the quality of entropy in the random bit generator.

The requirements for exact conformance of the Security Target are described in [NDcPP, 2] and in [SD, 3.1].

6.2.1.1.1 Conformance Claims (ASE_CCL.1)

The table below indicates the actions to be taken for particular ASE_CCL.1 elements in order to determine exact conformance with a cPP.

Table 14: Conformance Claims

ASE_CCL.1 Element	Evaluator Action
ASE_CCL.1.8C	The evaluator shall check that the statements of security problem definition in the PP and ST are identical.
ASE_CCL.1.9C	The evaluator shall check that the statements of security objectives in the PP and ST are identical.
ASE_CCL.1.10C	The evaluator shall check that the statements of security requirements in the ST include all the mandatory SFRs in the cPP, and all of the selection-based SFRs that are entailed by selections made in other SFRs (including any SFR iterations added in the ST). The evaluator shall check that if any other SFRs are present in the ST (apart from iterations of SFRs in the cPP) then these are taken only from the list of optional SFRs specified in the cPP (the cPP will not necessarily include optional SFRs, but may do so). If optional SFRs from the cPP are included in the ST then the evaluator shall check that any selection-based SFRs entailed by the optional SFRs adopted are also included in the ST.

6.2.1.1.2 TOE Summary Specification (ASE_TSS.1)

Evaluation Activities

The TOE summary specification shall describe how the TOE meets each SFR. In the case of entropy analysis the TSS is used in conjunction with required supplementary information on Entropy.

6.2.1.2 ADV: Development

The design information about the TOE is contained in the guidance documentation available to the end user as well as the TSS portion of the ST, and any required supplementary information required by this cPP that is not to be made public.

6.2.1.2.1 Basic Functional Specification (ADV_FSP.1)

The functional specification describes the TOE Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this cPP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE users, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional "functional specification" documentation is necessary to satisfy the Evaluation Activities specified in the SD.

The Evaluation Activities in the SD are associated with the applicable SFRs; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

Evaluation Activities

The evaluator shall check the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g., audit review or performing updates). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. The intent, is that these interfaces will be adequately tested, and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied.

The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.

The documents to be examined for this assurance component in an evaluation are therefore the Security Target, AGD documentation, and any supplementary information required by the cPP for aspects such as entropy analysis or cryptographic key management architecture[6]: no additional "functional specification" documentation is necessary to satisfy the Evaluation Activities. The interfaces that need to be evaluated are also identified by reference to the assurance activities listed for each SFR, and are expected to be identified in the context of the Security Target, AGD documentation, and any supplementary information required by the cPP rather than as a separate list specifically for the purposes of CC evaluation. The direct identification of documentation requirements and their assessment as part of the Evaluation Activities for each SFR also means that the tracing required in ADV_FSP.1.2D is treated as implicit, and no separate mapping information is required for this element.

However, if the evaluator is unable to perform some other required Evaluation Activity because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate

[6] The Security Target and AGD documentation are public documents. Supplementary information may be public or proprietary: the cPP and/or Evaluation Activity descriptions will identify where such supplementary documentation is permitted to be proprietary and non-public.

functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

6.2.1.3 AGD: Guidance Documentation

The guidance documents will be provided with the ST. Guidance must include a description of how the IT personnel verifies that the Operational Environment can fulfill its role for the security functionality. The documentation should be in an informal style and readable by the IT personnel.

Guidance must be provided for every operational environment that the product supports as claimed in the ST. This guidance includes:

- instructions to successfully install the TSF in that environment; and
- instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and
- instructions to provide a protected administrative capability.

Guidance pertaining to particular security functionality must also be provided; requirements on such guidance are contained in the Evaluation Activities specified in the SD.

6.2.1.3.1 Operational User Guidance (AGD_OPE.1)

The operational user guidance does not have to be contained in a single document. Guidance to users, administrators and application developers can be spread among documents or web pages.

The developer should review the Evaluation Activities contained in the SD to ascertain the specifics of the guidance that the evaluator will be checking for. This will provide the necessary information for the preparation of acceptable guidance.

Evaluation Activities

The evaluator shall check the requirements below are met by the guidance documentation.

Guidance documentation shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

Guidance documentation must be provided for every Operational Environment that the product supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target.

The contents of the guidance documentation will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in section 2 above.

In addition to SFR-related Evaluation Activities, the following information is also required.

a) The guidance documentation shall contain instructions for configuring any 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.

b) The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

1) 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).

2) 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.

c) The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The guidance documentation shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.

6.2.1.3.2 Preparative Procedures (AGD_PRE.1)

As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

Evaluation Activities

The evaluator shall check the requirements below are met by the preparative procedures.

The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in section 2 above.

Preparative procedures shall be distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

The contents of the preparative procedures will be verified by the Evaluation Activities defined below and as appropriate for each individual SFR in section 2 above.

In addition to SFR-related Evaluation Activities, the following information is also required.

Preparative procedures must include a description of how the administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target). The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

Preparative procedures must be provided for every Operational Environment that the product supports as claimed in the Security Target and must adequately address all platforms claimed for the TOE in the Security Target.

The preparative procedures must include

a) instructions to successfully install the TSF in each Operational Environment; and

b) instructions to manage the security of the TSF as a product and as a component of the larger operational environment; and

c) instructions to provide a protected administrative capability.

6.2.1.4 Class ALC: Life-cycle Support

At the assurance level provided for TOEs conformant to this cPP, life-cycle support is limited to end-user-visible aspects of the life-cycle, rather than an examination of the TOE vendor's development and configuration management process. This is not meant to diminish the critical role that a developer's practices play in contributing to the overall trustworthiness of a product; rather, it is a reflection on the information to be made available for evaluation at this assurance level.

6.2.1.4.1 Labelling of the TOE (ALC_CMC.1)

This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a "hard label" (e.g., stamped into the metal, paper label) or a "soft label" (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

6.2.1.4.2 TOE CM Coverage (ALC_CMS.1)

Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

6.2.1.5 Class ATE: Tests

Testing is specified for functional aspects of the system as well as aspects that take advantage of design or implementation weaknesses. The former is done through the ATE_IND family, while the latter is through the AVA_VAN family. For this cPP, testing is based on advertised functionality and interfaces with dependency on the availability of design information. One of the primary outputs of the evaluation process is the test report as specified in the following requirements.

6.2.1.5.1 Independent Testing – Conformance (ATE_IND.1)

Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes "evaluated configuration" instructions). The focus of the testing is to confirm that the requirements specified in Section 5 are being met. The Evaluation Activities in the SD identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

Evaluation Activities

The evaluator shall examine the TOE to determine that the test configuration is consistent with the configuration under evaluation as specified in the ST.

The evaluator shall examine the TOE to determine that it has been installed properly and is in a known state.

The evaluator shall prepare a test plan that covers all of the testing actions for ATE_IND.1 in the CEM and in the SFR-related Evaluation Activities. While it is not necessary to have one test case per test listed in an Evaluation Activity, the evaluator must show in the test plan that each applicable testing requirement in the SFR-related Evaluation Activities is covered.

The test plan identifies the platforms to be tested, and for any 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 and configuration of each platform to be tested, and any setup actions that are 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 any cryptographic engine to be used (e.g. for cryptographic protocols being evaluated).

The test plan identifies high-level test objectives as well as the test procedures to be followed to achieve those objectives, and the expected results.

The test report (which could just be an updated 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, so that a fix was then installed and then a successful re-run of the test was carried out, then the report would show a "fail" result followed by a "pass" result (and the supporting details), and not just the "pass" result[7].

6.2.1.6 Class AVA: Vulnerability Assessment

For the first generation of this cPP, the iTC is expected to survey open sources to discover what vulnerabilities have been discovered in these types of products and provide that content into the AVA_VAN discussion. In most cases, these vulnerabilities will require sophistication beyond that of a basic attacker. This information will be used in the development of future protection profiles.

6.2.1.6.1 Vulnerability Survey (AVA_VAN.1)

Appendix A in [SD] provides a guide to the evaluator in performing a vulnerability analysis.

Evaluation Activities

The evaluator shall document their analysis and testing of potential vulnerabilities with respect to this requirement. This report could be included as part of the test report for ATE_IND, or could be a separate document.

The evaluator formulates hypotheses in accordance with process defined in Appendix A. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.5. The evaluator shall then perform vulnerability analysis in accordance with Appendix A.4. The results of the analysis shall be documented in the report according to Appendix A.5.

[7] It is not necessary to capture failures that were due to errors on the part of the tester or test environment. The intention here is to make absolutely clear when a planned test resulted in a change being required to the originally specified test configuration in the test plan, to the evaluated configuration identified in the ST and guidance documentation, or to the TOE itself.

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

7.1 Security Audit

7.1.1 Audit Generation

The TOE generates Audit records for system configuration, administrative-user management, cryptographic operations and traffic events. Audit logs can be sent from the local buffer to a syslog server in the Operational Environment. The TOE can be configured to send audit logs to up to five syslog servers. The TOE maintains an internal time source and can synchronize time using an NTP server in the Operational Environment.

The log messages for administrative-user and security management include information such as:

- User logins and listens (logouts)
- Failed login or set privilege attempts
- Successfully executed configuration commands
- File operations (including generation, import, changing, or deleting of keys)
- System messages (including password –related commands)
- Reboot information

In addition, the AuditLog service can be configured to provide System Message logging to the syslog server that contains information regarding particular traffic flows. System log data includes:

- Date/Time of the event
- Interface
- Packet Header summary
- Reason/Event summary
- Status: Success/Failure

Cryptographic functions are logged in Local Audit logs and configured syslog servers that include Key Encryption Key (KEK) generation/zeroization and session data supporting IPsec and Internet Key Exchange (IKE). IKE failure logs include cookie pair identification and the name of the payload responsible for the rejection.

Logs are coded using a severity level number based on the perceived importance of the event. The coding system follows the convention:

0=Emergency, 1=Alert, 2=Critical, 3=Error, 4=Warning, 5=Notice, 6=Info, 7=Debug

Logs may also be viewed directly from the local audit buffer by executing a SHow command. The TOE allows filtering of audit logs based on service, severity, facility and/or message identifier.

Logs are formatted as follows:

<Priority>	The priority of the message.
<SeqNumber>	A number from 0 to 255. This is an identifier for the Syslog event.
<Hostname/MAC Address>	The resolved host name or MACaddress. When displaying the logfiles on the syslog server, this field is prepended to each log entry.
<Severity>	The severity level in numeric form. Severity levels are as follows: 0=Emergency 1=Alert 2=Critical 3=Error 4=Warning 5=Notice 6=Info 7=Debug
<Status>	Success or Failure of the action.
<Entity|Username>	The entity or username that initiated the log message. Username specifies who initiated the command.
<Service>	Service of the EOS that initiated the log message.
<Source>	Source of the log message. Possible sources include: CONSOLE — The console port. EOS — The Enterprise OS system. LoadConfig — The UI LoadConfig command. This source is visible only on locally logged messages. xxx.xxx.xxx.xxx — The IP address of the SNMP management station that initiated an SNMP SET request. This source is visible only on locally logged messages.
<Text>	A description of the event.

FAU_GEN.1, FAU_GEN.2

7.1.2 Audit Storage

Only the Administrator may delete audit records from the buffer on the device. When a command is issued that would interact with the audit log, the resulting action is taken if permission to the audit log directory is allowed.

When a new audit log is created, it is entered into the local audit log. At that time, the syslog message is transmitted to the configured syslog server on UDP port 514, through the IPsec connection. The IPsec connection is established upon device startup, based on IPsec configuration. The audit log is never re-transmitted. The buffer size is 64KB. The TOE overwrites the oldest records when the buffer is filled.

FAU_STG.1, FAU_STG_EXT.1

7.2 Cryptographic Support

7.2.1 Cryptographic Key Generation and Establishment

The TOE can generate 2048-bit RSA for authentication purposes in SSH and IKEv1. Additionally 256-bit ECDSA or 384-bit ECDSA keys can be generated for authentication purposes in IKEv1. The TOE can generate also 256-bit ECDSA or 384-bit ECDSA ephemeral keys for use for IKEv1 key establishment."

FCS_CKM.1, FCS_CKM.2

7.2.2 Cryptographic Key Destruction

Table 15: Plaintext Key Information

Key Identifier	Origin	Storage Location	Clearing description
KEK	Generated within TOE via UI command.	Plaintext in FLASH, plaintext in RAM (copied to RAM upon bootup).	Zeroized by UI commands: *kekzeroize* *zeroize*
IKEv1 Preshared Keys	Imported into the TOE via UI command.	Encrypted with the KEK in Flash, plaintext in RAM during use.	Zeroized by UI commands: *del –crypto fpsk* *kekzeroize* *zerorize*
SKEYID	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE session is terminated or *zeroize* command is executed.
SKEYID_d	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE session is terminated or *zeroize* command is executed.
SKEYID_a	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE session is terminated or *zeroize* command is executed.
SKEYID_e	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE session is terminated or *zeroize* command is executed.
Ephemeral DH Phase-1 private key (a)	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE Phase-1 SA is expired or *zeroize* command is executed.
Ephemeral DH Phase-2 private key (a)	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IKE Phase-2 SA is expired or *zeroize* command is executed.
IPSec Session Keys	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when IPsec session is terminated.
IKEv1 RSA Private Key	Generated within the TOE via UI command.	Encrypted with the KEK in Flash, plaintext in RAM during use.	Zeroized by UI commands: *del –pki keypair* *kekzeroize* *zeroize*
IKEv1 ECDSA Private Key	Generated within the TOE via UI command.	Encrypted with the KEK in Flash, plaintext in RAM during use.	Zeroized by UI commands: *del –pki keypair* *kekzeroize* *zeroize*

Table 15: Plaintext Key Information			
Key Identifier	Origin	Storage Location	Clearing description
SSH RSA Private Key	Generated within the TOE via UI command.	Encrypted with the KEK in Flash, plaintext in RAM during use.	Zeroized by UI commands: *kekzeroize* *zeroize*
SSH Session Keys	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized when SSH session is terminated or *zeroize* command is executed.
SSH DH Private Key	Generated within the TOE	Not stored in flash. Plaintext in RAM during use.	Zeroized after establishing SSH Session Keys or when *zeroize* command is executed.

Table 15 Plaintext Key Information describes the key identifier, usage, origin, storage location, and clearing description. At each use of the term "zeroize," the TOE performs a direct overwrite of the key with zeros, followed by a read-verify. The read-verify applies to all instances of the key in flash memory only.

FCS_CKM.4

7.2.3 Cryptographic Operations

Table 16: Cryptographic Algorithm Certificates				
SFR	Description	S6000 Algorithm Certificate Number	GGM8000 Algorithm Certificate Number	Description of use
FCS_CKM.1	RSA Key Generation	2049	2049	Asymmetric ley used to authenticate oneself to peer.
FCS_CKM.1	ECDSA Key Generation	887	887	Asymmetric key used to authenticate oneself to peer.
FCS_CKM.2	ECDSA Key Establishment	816	816	Algorithm to establish session keys.
FCS_COP.1(1)	AES-CBC 128, 256	3993	3993	Algorithm to protect SSH sessions
FCS_COP.1(1)	AES-CBC 128, 256 (Hardware)	173	962	Algorithm to protect IPsec sessions
FCS_COP.1(2)	RSA Signature Generation	2049	2049	IKEv1 and SSH Authentication
FCS_COP.1(2)	ECDSA Signature Generation	887	887	IKEv1 Authentication
FCS_COP.1(3)	SHA-1, SHA-256, SHA-384 Hashing	3295	3295	Digital Signature generation, verification for SSH, Trusted Update, and as PRF function in IKEv1.
FCS_COP.1(3)	SHA-1 Hashing (Hardware)	258	933	IPsec ESP for data authentication and integrity.

Table 16: Cryptographic Algorithm Certificates				
SFR	Description	S6000 Algorithm Certificate Number	GGM8000 Algorithm Certificate Number	Description of use
FCS_COP.1(4)	HMAC-SHA-1-96 Keyed Hashing (Hardware)	39	1487	Used for integrity of IPsec connections. Note that this 96-bit digest is generated by truncating a SHA-1 160-bit digest.
FCS_COP.1(4)	HMAC-SHA-1 Keyed Hashing (OpenSSH)	2607	2607	Used for integrity of SSH connections.
FCS_COP.1(4)	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 Keyed Hashing (OpenSSL)	2606	2606	Used as PRF function in IKEv1.
FCS_RBG_EXT.1	Hash-Based DRBG (OpenSSL)	1184	1184	Used in generation of random numbers.

Table 17: HMAC Characteristics				
HMAC Algorithm	Hash Function Used	Block Size	Output MAC Length	Key Size
HMAC-SHA-1-96 (Hardware)	SHA-1	512 bits	96 bits	160 bits, 256 bits, 384 bits
HMAC-SHA-1 (OpenSSH)	SHA-1	512 bits	160 bits	160 bits
HMAC-SHA-1 (OpenSSL)	SHA-1	512 bits	160 bits	112 – 1024 bits
HMAC-SHA-256 (OpenSSL)	SHA-256	512 bits	256 bits	112 – 1024 bits
HMAC-SHA-384 (OpenSSL)	SHA-384	1024 bits	384 bits	112 – 1024 bits

The TOE supports following DH groups in IKEv1:

- Group14 - 2048-bit MODP Group
- Group19 - ECDH using NIST curve P-256
- Group20 - ECDH using NIST curve P-384

SSH:

- Group14 - 2048-bit MODP Group

FCS_COP.1.1(1), FCS_COP.1.1(2), FCS_COP.1.1(3), FCS_COP.1.1(4)

7.2.4 IPsec Protocol

The TOE can operate in transport mode or tunnel mode, supporting both AES-CBC-128 and AES-CBC-256 for ESP encryption.

The TOE supports IKEv1, exchanges in Phase 1 are done in main mode and there is not possibility to enable aggressive mode.

In IKEv1 the TOE supports:

- pre-shared keys, RSA and ECDSA as peer authentication methods,
- AES-CBC-128 and AES-CBC-256 as encryption algorithms (for IKEv1 payloads),
- SHA1, SHA-256 and SHA-384 as hashing algorithms,
- DH groups: Group14, Group19 and Group20.

The Administrator configures IKEv1 by defining IKE Profiles. IKE Profile is a set of following parameters: priority, authentication method, encryption algorithm, hash algorithm, DH Group, lifetime.

Priority determines order in which IKE Profiles are used by the TOE for IKEv1 Phase 1 SA creation. The Administrator can assign highest priority for IKE profile with preferred DH Group causing that this group will be sent in first SA proposal to the peer.

DRBG described in section 7.2.5 is used by the TOE to generate ephemeral DH private keys. These keys have fixed length: 2048 bits for Group14, 256 bits for Group19 and 384 bits for Group20.

Lifetime from IKE Profile determines how long created IKEv1 Phase 1 SA can be used by the TOE.

The IKEv1 Phase 2 SA lifetime can be limited by the Administrator globally or for particular IPsec connection. Limitation can be expressed in time period or amount of transferred data with particular SA.

The TOE cannot generate itself pre-shared keys used for authentication of IPsec connections. Pre-shared keys have to be created / generated outside of the TOE and then manually configured by the Administrator on the TOE. The TOE doesn't display pre-shared key in plaintext form, only its hash can be displayed.

AES-CBC-128 and AES-CBC-256 are allowed encryption algorithms for the IKEv1 and ESP exchanges. The strength of AES-CBC-128 is 128bits while for AES-CBC-256 it is 256bits. The TOE by default ensures that the strength of encryption algorithm negotiated in IKEv1 Phase1 is greater or equal to the strength of encryption algorithm used to protect IKEv1 Phase2 connection.

The Administrator can specify list of peers permitted to establish connection with the TOE. Permitted peer can be identified by IP address, DNS name or email address. In case of RSA or ECDSA authentication method, peer’s identifier must be present in certificate presented by the peer. The TOE performs certificate path validation for peer’s certificate and establishes IPsec connection only when validation succeeds.

The TOE supports multiple SPDs, where single SPD is defined by selector and access control lists. Selector list consists of SPD rules and its priorities. Each SPD rule contains “Include” or “Exclude” action. Packets matching SPD rule with “Exclude” action are BYPASSED, packets matching SPD rule with “Include” action are PROTECTED while packets not matching any SPD rule are DISCARDED.
SPD rules on access control list have “Permit” or “Deny” action associated. Packets matching SPD rule “Deny” are DISCARDED, while packets matching SPD rule “Permit” are further process by the TOE. Detailed decryption of packets processing can be found in next sections of this document.

7.2.4.1 Processing Incoming Packets

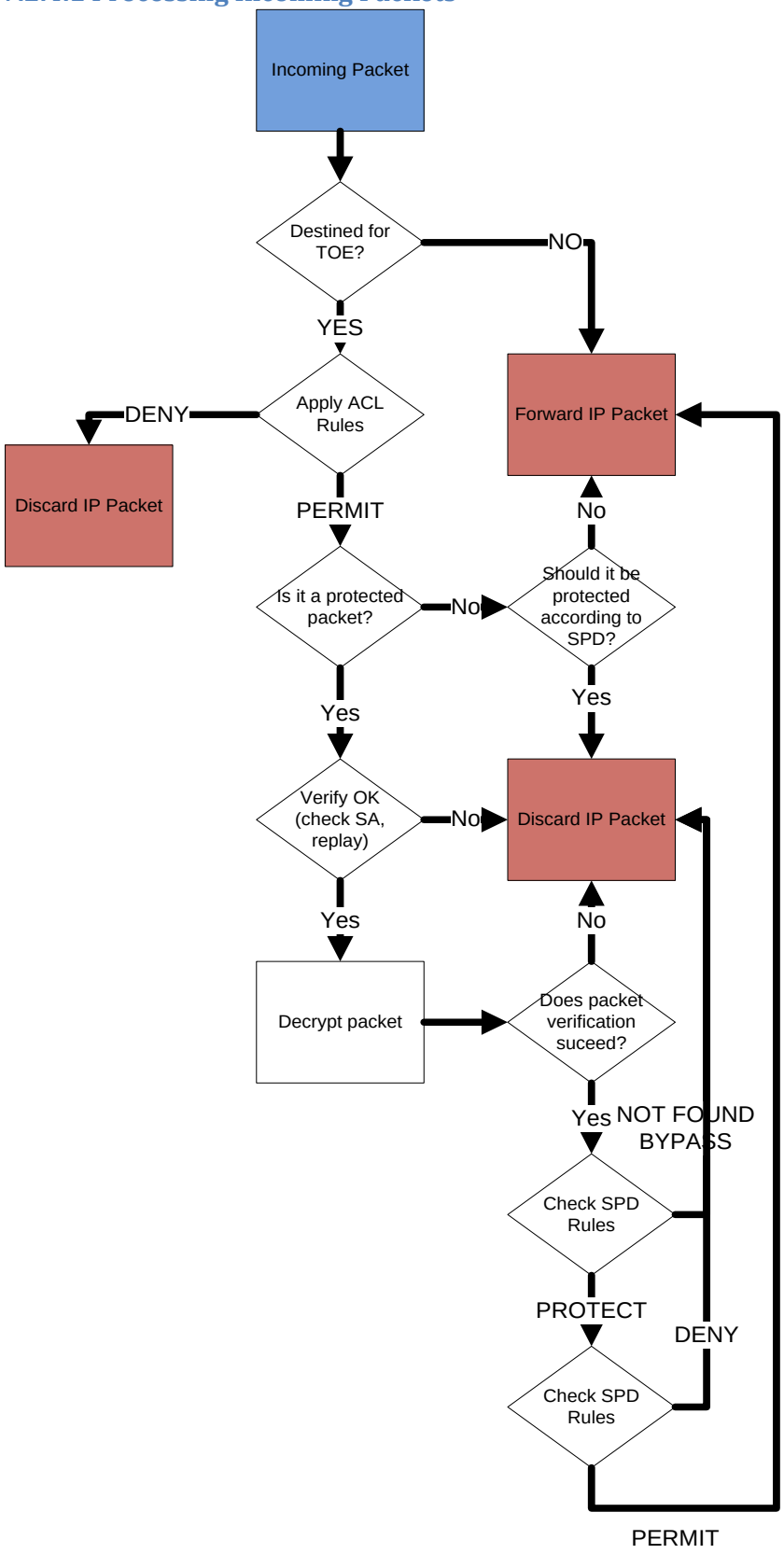

Figure 1 – Incoming Packets Processing Logic

Figure 1 – Incoming Packets Processing Logic illustrates the chain of logic used to determine the correct actions for incoming packets.

Upon receipt of a packet, the TOE first identifies whether the packet is destined for the TOE or not. The IPv4 header information is compared to the TOE IPv4 address to make this determination. It means that all packets from IPsec tunnels terminated on the TOE will be addressed to the TOE. If the packet is not destined for the TOE, the packet is sent through to the appropriate interface for transmission (packet forwarding).

If destined for the TOE, the packet is processed by the ACL filter and is deleted (discarded) by the TOE based on ACL rules for the packet IP/port information or is inspected to determine if the packet is protected. The ACL rules determine if the packet is to be deleted (discarded). If the packet is not discarded, the header information is inspected to determine if the packet is protected. If packet is not protected but is ISAKMP packet it is processed by the TOE. If the packet is not protected but should be per the SPD (based on header information comparison to selectors in the SPD), the packet is deleted (discarded). The TOE supports following SPD rules for incoming packets: Protect, Bypass, Discard. If the packet is not protected and does not meet SPD rules, the packet is forwarded. If the packet is protected, the packet is compared against current Security Associations (SAs) using the SPI field. If the packet does not match an SA, the packet is deleted. Protected packets matching an SA are decrypted and verified. Verification begins with ensuring that packet is the length specified in the header. The SA lifetime is checked to ensure that the SA is still valid, and the SA size (amount of data that is allowed per SA) is checked to ensure that the packet can be transmitted under the valid SA. Successfully verified packets are compared against the set of SPD rules. If the packet does not meet SPD rules (bypass/discard) the packet is discarded. If the packet meets SPD rules then is processed by the ACL filter and is forwarded to the appropriate destination while all other packets are deleted.

7.2.4.2 Processing Outgoing Packets

Figure 2 - Outgoing Packets Processing Logic illustrates the chain of logic used to determine the correct actions for outgoing packets.

Packets are filtered through ACL rules based on IPv4 address and port. The ACL rules determine if the packet should be deleted (discarded) or processed further.

Outgoing packets are assessed based on their designated outgoing interface. If the interface is configured as an unprotected interface (logical or physical), the packet is forwarded through to the appropriate interface for transmission.

Packets being transmitted through protected interface are compared against the set of SPD rules for that interface. The TOE supports following SPD rules for outgoing packets: Protect, Bypass, Discard. Port and IPv4 address information is used for the comparison. All packets that do not match an SPD rule are forwarded (bypassed).

Packets matching an SPD are encrypted and forwarded to the interface for transmission.

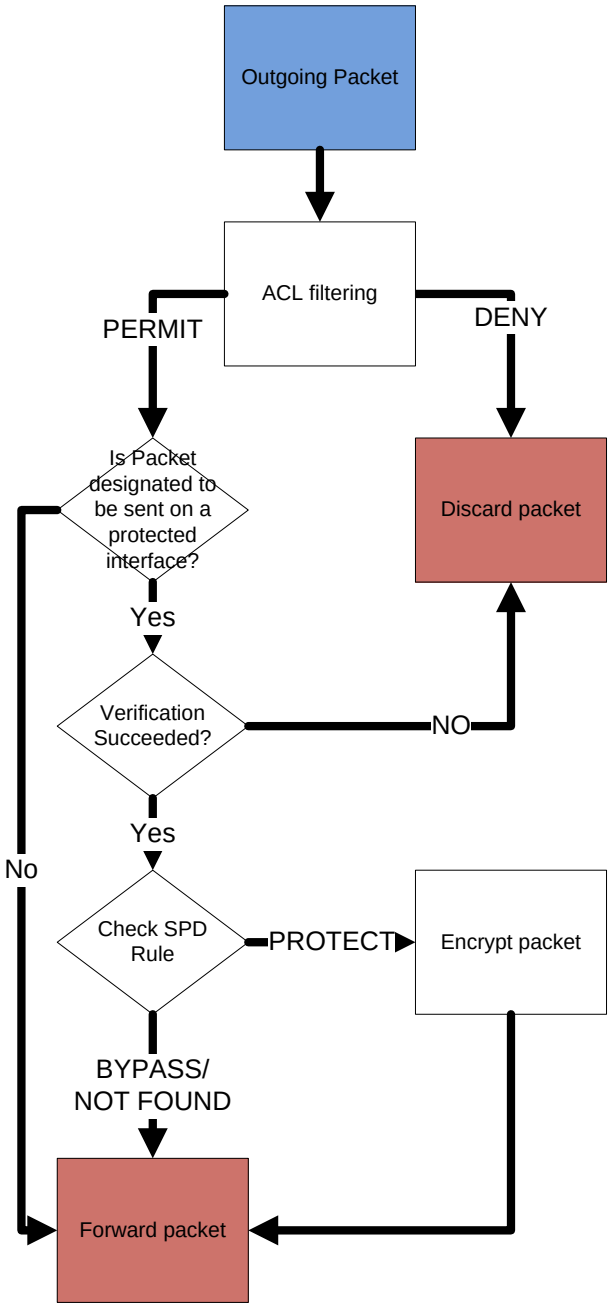

Figure 2 - Outgoing Packets Processing Logic

FCS_IPSEC_EXT.1.1

7.2.5 Random Bit Generation

The Random Bit Generator (RBG) used by the TOE varies between the hardware platforms. The Entropy Assessment [ENT] describes all states of Entropy, including the amount of data sampled. The TOE ensures that the DRBG is seeded with at least 256 bits of entropy at all times. See the [EAR] for additional detail. Note that [EAR] is proprietary, and not available for public consumption.

FCS_RBG_EXT.1

7.2.6 SSH Server Protocol

The TOE is conformant with RFC 4251, 4252, 4253, 4254.

The TOE supports only the following options in the referenced RFCs:

- ssh-rsa and password-based as authentication methods,
- aes128-cbc and aes256-cbc as encryption algorithms,
- hmac-sha1 as MAC algorithm,
- diffie-hellman-group14-sha1 as key exchange method.

SSH Packets larger than 4096 bytes are dropped. The TOE compares the total length of the received SSH packet (before decryption) with the maximum allowed value (4096 bytes) and drops the packet if the length is greater.

The TOE maintains a counter of the number of packets that have been transmitted using the current key. The TOE forces the connection to rekey after 2^{28} packets have been transmitted using that key.

The TOE does not support any compression features.

See Section 7.3.2, User Identification and Authentication for a full description of the logon process.

FCS_SSHS_EXT.1

7.3 Identification and Authentication

7.3.1 Password Management

The TOE requires positive identification and authentication of administrative-users prior to granting access the security management interface. Authentication may be performed either by the TOE itself using an internal database or using a RADIUS server in the Operational Environment. Upon the entry of password authentication credentials, the TOE processes the password. The processed values of the entered credentials are checked against the internal database or passed to a RADIUS for validation. The TOE does not store the plaintext password.

An operator must enter a username and its password to log in. Passwords are alphanumeric strings consisting of 7 to 128 characters chosen from all upper case, lower case, numbers, and special characters “!”, “%”, “&”, “*”, “(“, “)”, "+", ":", ";", "<", ">", "?".

FIA_PMG_EXT.1.1

7.3.2 User Identification and Authentication and Password-based Authentication Mechanism

Local administrative sessions require a direct serial connection. Once the low-level connectivity is established, the user will see the warning banner populate the screen, as well as the login prompt, asking for user name. The user enters the name (which is echoed back during entry). The user is then prompted for the password. The user enters the password (which is not echoed back to the user during entry). The password is hashed and compared with the internally stored password. In the case of remote authentication (RADIUS), the password is transmitted to the RADIUS server. Upon successful comparison, authentication is successful. This grants the user access into the TOE, and begins the Administrative session.

Remote administrative sessions require an SSH client with (optional) RSA-based authentication support. The SSH client will negotiate a session key using Diffie-Hellman group 14 with SHA-1. Once connected, the user will have the same experience as described for local sessions.

In the case of RSA-based authentication, no password is used. Instead, the client sends a KEYID (key identifier) to the TOE. The TOE uses the internally stored public key associated with that KEYID and encrypts a random number. That encrypted number is sent to the client. The client decrypts that number, and creates an MD5 hash. The MD5 hash is sent back to the TOE. The TOE creates an MD5 hash of the originally transmitted random number and compares that to the received value from the client. If they are equal, the authentication is successful.

The only services available to non-authenticated entities are network-based services (ARP, ICMP, routing services, BFD send, DHCP services, SSH, IPDV (port UDP/49402), RSVP (port UDP/1698), NTP (port UDP/123)) and the warning banner available over Administrative channels.

FIA_UIA_EXT.1, FIA_UAU_EXT.2.1

7.3.3 Protected Authentication Feedback

During authentication of local and remote SSH sessions, all password information provided to the TOE is not displayed on the screen. The only fields presented to the user are the access banner and the username.

FIA_UAU.7.1

7.3.4 X.509 Certificate Validation, Authentication, and Request

The TOE can be used to generate a CSR (Certificate Signing Request) to be transmitted by the Administrator to a CA (Certificate Authority). The Administrator can provide CN, O, OU, C attributes for Subject field of generated CSR. Device specific information like IP address, DNS name and e-mail address, if configured by the Administrator, will be added by the TOE as Subject Alternative Name for generated CSR.

The newly generated CSR is stored internally in PKCS#10 PEM format. The CSR must be copied from the TOE by the Administrator in order to transmit to the CA. This transmission must be performed manually by the Administrator (using non-TOE resources). The CSR can be viewed by the Administrator, but not modified.

Once the CA has received the CSR, it must return the signed certificate back to the Administrator (using non-TOE resources). The Administrator can then import the public certificate to be used for all future PKI-based communications. The TOE can have only one self certificate. The TOE allows the Administrator to import trusted CA, intermediate CA, and end-entity public certificates. All certificates must be imported manually by the Administrator.

All certificates are validated during the import process. Validation process can be divided on three steps.

- The first step is inspection of certificate's mandatory fields in order to determine its presence, correctness and validity.
- The second step is inspection of certificate extensions according to configured trust validation criteria. Checking whether the CA flag is true and Path length is part of the path validation process.
- The third step is path validation which is performed for all untrusted certificates. Checking CA Flag and Path length is part of path validation process.

Only certificates which have successfully passed the validation can be used by the TOE for PKI-based communications. Validation status for imported certificates can be checked in any time by the Administrator.

CRLs are used to check revocation status for certificates during path validation process. The TOE supports HTTP servers as distribution points for CRLs. For certificates without the CRL distribution point extension the TOE allows static configuration of CRL distribution point.

If a CRL is unavailable during the authentication process, the TOE will assume that the certificate is not revoked and operate as if the CRL was verified and the certificate is in good standing (from a revocation perspective only).

Authenticating a peer certificate (used during establishment of an IPsec tunnel) requires the configuration of an IKE profile. This IKE profile will include specific details about the allowed peers. Based on the IKE profile and the presented X.509 certificate (from the peer), the TOE will perform certificate validation and establish a connection following the steps defined by RFC 5280 and 4945, including verification that the certificate is trusted by the appropriate root CA (as installed by the Administrator).

FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3

7.4 Security Management

7.4.1 Management of Security Functions Behaviour

In order to manage security functions of the TOE, the Administrator must authenticate to the TOE per Section 7.3.2 User Identification and Authentication and Password-based Authentication Mechanism.

The Administrator can modify the following TSF functionality:

- Global settings
  - Console timeout threshold
  - Password complexity
  - Account expiration
  - Account lockout duration
  - Software Update
- IPsec
  - Port (physical, virtual port number)
  - SPD rules
  - IKE profiles
  - Dynamic and manual security policies
  - Global manual security policy
- SSH
  - Session timeout threshold
  - Encryption algorithm
- Audit
  - Number of log files for reboot logs
  - Syslog IP address
  - Syslog facility number
  - Syslog time format (classic, rfc)
- NTP
  - Server address
  - Polling interval
- RADIUS
  - Server address
  - Port number
  - Resolution order (local/remote)

- Radius secret
- Retransmission timer
- Expiration timer

The Administrator can enable or disable IPsec, routing, SSH, Auditing, NTP, and/or RADIUS authentication.

FMT_MOF.1.1(1)/TrustedUpdate, FMT_MOF.1.1(1)/Audit, FMT_MOF.1.1(2)/Audit, FMT_MOF.1.1(1)/AdminAct, FMT_MOF.1.1(2)/AdminAct

7.4.2 Management of TSF Data

No administrative interfaces are available prior to successful authentication. The following data can be modified or overwritten by the Administrator:

- User account names
- User passwords
- Internally generated cryptographic keys
- Imported SSH public keys

FMT_MTD.1, FMT_MTD.1.1/AdminAct

7.4.3 Specification of Management Functions and Restrictions on Security Roles

In addition to the TSF data and services that an Administrator can modify in Sections 7.4.1 and 7.4.2, The Administrator is also able to configure the text of the access banner (including removing the access banner), and initiate TOE updates.

All authenticated users are considered Administrative, as there are no unprivileged accounts.

FMT_SMF.1, FMT_SMR.2

7.5 Protection of the TSF

7.5.1 Protection of Administrator Passwords

Administrator passwords (those passwords used for authenticating users) are stored in two separate ways. For those accounts that are built into the TOE (hardcoded usernames), the password is stored in an MD5 hashed form only. For all other users (with usernames created by other users), the password is stored in a non-plaintext form. In both cases, the plaintext password does not exist within the TOE. Additionally, no interfaces exist to view the password associated with any account.

FPT_APW_EXT.1

7.5.2 Protection of TSF Data (for reading of all symmetric keys)

All pre-shared, symmetric, and private keys that are stored in the TOE are stored in an encrypted format. All of the above keys are encrypted using AES-128 and a Key-Encrypting Key (KEK). See Table 15: Plaintext Key Information for more information. No interfaces exist to view any plaintext keys.

FPT_SKP_EXT.1

7.5.3 TSF Testing

The TOE runs a suite of self-tests during initial start-up to verify its correct operation. If any of the tests fail, the TOE will reboot and restart all tests. During the system boot process (power on or reboot), Power on Startup Test (POST) tests are performed for all the cryptographic module functions. The Software Integrity Test is run automatically whenever the system image is loaded and confirms through use of a checksum verification (in the case of the S6000) and a signature verification (in the case of the GGM 8000). This ensures that the software image to be loaded has not been corrupted and has maintained its integrity.

NDRNG , DRBG, Firmware load, Key Establishment (RSA and ECDSA), and Alternating Bypass Self-Tests are all performed according to FIPS 140-2 section 4.9.2.

The KATs ensure that the cryptographic functionality is operating correctly and has not been compromised and the integrity test ensures that the software has not been modified/corrupted.

FPT_TST_EXT.1(1), FPT_TST_EXT.1(2)

7.5.4 Trusted Update

In order to initiate the update process, the Administrator must copy the candidate update image to a specific file location within the TOE. Once the copy is completed, the TOE immediately checks file contents (as it does for all new files loaded into the TOE). Once the TOE identifies the file as a software image (update candidate), the image is cryptographically verified using the public key stored within the current running image using SHA-256 and RSA-2048. If the signature verification is successful, the new software image (update candidate) is available for installation and use during the next boot cycle. If the verification fails, the update candidate is deleted from the file system and no update is performed.

FPT_TUD_EXT.1

7.5.5 Reliable Time Stamps

The TOE includes a real-time clock (RTC) within the TOE hardware. The TOE is reliant on this device to provide accurate time. Based on the RTC vendor statements, the real-time clock should not draft more than 1.53 minutes in any month. Given the use of NTP, this does not represent a security concern.

The TOE supports the use of an NTP server in the Operational Environment. This requires the configuration and use of an IPsec channel between the TOE and the NTP server. Once the IPsec channel is established, NTP can be configured to automatically update the system time. Each time that the system time is changed, the TOE audits the original time and modified time, as well as the user (or process) initiating the update.

Though the time may be used for many functions (a pseudorandom number for a non-security relevant function, for example), an accurate time value is only needed for the auditing process. The time interval (counter) is used for IPsec, session timers, audit log, PKI, firewall, and IKEv1 services.

FPT_STM.1

7.6 TOE Access

7.6.1 TSF and User-initiated Session Locking and Termination

All administrative sessions use the same mechanisms for TOE access.

As a session begins, a timer is started. Each time the user issues a command, the timer is reset to zero. When the timer increases to an Administrator-defined threshold, the connection is terminated, requiring Administrator authentication prior to accessing the TOE. The termination threshold for local sessions can be set independently from remote administrative sessions. At any time while logged in to the TOE, the user can end the session by issuing the corresponding command.

FTA_SSL_EXT.1, FTA_SSL.3, FTA_SSL.4

7.6.2 Default TOE Access Banners

The TOE supports local (serial) and remote (SSH) connections for all administrative sessions. Prior to entering any username into the TOE, the access banner is output to the user. The text of the access banner is configurable by the Administrator.

FTA_TAB.1

7.7 Trusted Path/Channels

7.7.1 Inter-TSF Trusted Channel

The TOE implements the IPsec protocol for all trusted channels as described in Section 7.2.4 IPsec Protocol. The trusted channels support all communication to and from the RADIUS server, NTP server, and syslog server.

FTP_ITC.1

7.7.2 Trusted Path

The TOE implements the SSH protocol for all trusted paths as described in Section 7.2.6 SSH Server Protocol, using AES-128-CBC or AES-256-CBC. The trusted path supports all communications for the purpose of remote administration.

FTP_TRP.1

8. Terms and Definitions

Table 18: TOE Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
AES	Advanced Encryption Standard
CA	Certificate Authority
CN	CommonName
cPP	Collaborative Protection Profile (a reference to [NDcPP])
CSR	Certificate Signing Request
DRBG	Deterministic Random Bit Generator
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standards
HMAC	(Keyed-) Hash Message Authenticating Code
IKE	Internet Key Exchange
KAT	Known Answer Test
KEK	Key-Encrypting Key
NTP	Network Time Protocol
POST	Power On Self-Test
RADIUS	Remote Authentication Dial-In User Service
RBG	Random Bit Generator
RSA	Rivest, Shamir, Adleman
RTC	Real-Time Clock
SA	Security Association
SHA	Secure Hash Algorithm
SPD	Security Policy Database
SSH	Secure Shell

Table 19: 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
SPD	Security Policy Database
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface

9. References

Table 20: TOE Guidance Documentation			
Reference	Description	Version	Date
[AGD]	Network Device S6000 and GGM8000 with EOS Version 16.9 Common Criteria User Guide	1.2	July 28, 2016
[EOS1]	Enterprise OS Software Version 16.9 Reference Guide		June 28, 2016
[EOS2]	Enterprise OS Software Version 16.9 User Guide		June 26, 2016
[GGM]	GGM 8000 Hardware User Guide		May 30, 2016
[S6000]	S6000 Hardware User Guide		May 30, 2016

Table 21: Common Criteria v3.1 References			
Reference	Description	Version	Date
[C1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model CCMB-2012-09-001	V3.1 R4	September 2012
[C2]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components CCMB-2012-09-002	V3.1 R4	September 2012
[C3]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components CCMB-2012-09-003	V3.1 R4	September 2012
[C4]	Common Criteria for Information Technology Security Evaluation Evaluation Methodology CCMB-2012-09-004	V3.1 R4	September 2012

Table 22: Supporting Documentation			
Reference	Description	Version	Date
[NDcPP]	Collaborative Protection Profile for Network Devices	1.0	February 27, 2015
[SD]	Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP	1.0	February 27, 2015
[EAR]	Motorola Network Routers Entropy Assessment Report	0.1	April 13, 2016
[FIPS]	FIPS 140-2	N/A	May 25, 2001

Annex A Algorithm Validation Requirements

FCS_CKM.1.1

Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

Key Generation for FIPS PUB 186-4 RSA Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d.

Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include:

a) Random Primes:
   - Provable primes
   - Probable primes

b) Primes with Conditions:
   - Primes p1, p2, q1, q2, p and q shall all be provable primes
   - Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes
   - Primes p1, p2, q1, q2, p and q shall all be probable primes

To test the key generation method for the Random Provable primes method and for all the Primes with Conditions methods, the evaluator must seed the TSF key generation routine with sufficient data to deterministically generate the RSA key pair. This includes the random seed(s), the public exponent of the RSA key, and the desired key length. For each key length supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-384and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-384and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct).The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

- Primes q and p shall both be provable primes
- Primes q and field prime p shall both be probable primes

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process
- Generator g constructed through an unverifiable process.

The Key generation specifies 2 ways to generate the private key x:

- len(q) bit output of RBG where 1 <=x <= q-1
- len(q) + 64 bit output of RBG, followed by a mod q-1 operation where 1<= x<=q-1.

The security strength of the RBG must be at least that of the security offered by the FFC parameter set.

To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm:

- g != 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

for each FFC parameter set and key pair.

FCS_CKM.2.1

Key Establishment Schemes

The evaluator shall verify the implementation of the key establishment schemes of the supported by the TOE using the applicable tests below.

SP800-56A Key Establishment Schemes

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role-key confirmation type combination, the

tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MACtags generated from these values.

If key confirmation is supported, the TSF shall perform the above for each implemented approved MAC algorithm.

Validity Test

The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

The TOE shall use these modified test vectors to emulate the key agreement scheme using the corresponding parameters. The evaluator shall compare the TOE's results with the results using a known good implementation verifying that the TOE detects these errors.

SP800-56B Key Establishment Scheme Testing

The evaluator shall verify that the TSS describes whether the TOE acts as a sender, a recipient, or both for RSA-based key establishment schemes.

If the TOE acts as a sender, the following assurance activity shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

a) To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with or without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA public key, the plaintext keying material, any

additional input parameters if applicable, the MacKey and MacTag if key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform a key establishment encryption operation on the TOE with the same inputs (in cases where key confirmation is incorporated, the test shall use the MacKey from the test vector instead of the randomly generated MacKey used in normal operation) and ensure that the outputted ciphertext is equivalent to the ciphertext in the test vector.

If the TOE acts as a receiver, the following assurance activities shall be performed to ensure the proper operation of every TOE supported combination of RSA-based key establishment scheme:

a) To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each combination of supported key establishment scheme and its options (with our without key confirmation if supported, for each supported key confirmation MAC function if key confirmation is supported, and for each supported mask generation function if KTS-OAEP is supported), the tester shall generate 10 sets of test vectors. Each test vector shall include the RSA private key, the plaintext keying material (KeyData), any additional input parameters if applicable, the MacTag in cases where key confirmation is incorporated, and the outputted ciphertext. For each test vector, the evaluator shall perform the key establishment decryption operation on the TOE and ensure that the outputted plaintext keying material (KeyData) is equivalent to the plaintext keying material in the test vector. In cases where key confirmation is incorporated, the evaluator shall perform the key confirmation steps and ensure that the outputted MacTag is equivalent to the MacTag in the test vector.

b) The evaluator shall ensure that the TSS describes how the TOE handles decryption errors. In accordance with NIST Special Publication 800-56B, the TOE must not reveal the particular error that occurred, either through the contents of any outputted or logged error message or through timing variations. If KTS-OAEP is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.2.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each. If KTS-KEM-KWS is supported, the evaluator shall create separate contrived ciphertext values that trigger each of the three decryption error checks described in NIST Special Publication 800-56B section 7.2.3.3, ensure that each decryption attempt results in an error, and ensure that any outputted or logged error message is identical for each.

FCS_COP.1.1(1)

AES-CBC Known Answer Tests

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

KAT-1. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

KAT-2. To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

KAT-3. To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key I in each set shall have the leftmost i bits be ones and the rightmost N-I bits be zeros, for I in [1,N].

To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

KAT-4. To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an I-block message where 1 < i <=10. The evaluator shall choose a key, an IV and plaintext message of length I blocks and encrypt the message, using the mode to be tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

The evaluator shall also test the decrypt functionality for each mode by decrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and a ciphertext message of length I blocks and decrypt the message, using the mode to be tested, with the chosen key and IV. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key and IV using a known good implementation.

AES-CBC Monte Carlo Tests

The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3-tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
        if i == 1:
                CT[1] = AES-CBC-Encrypt(Key, IV, PT)
                PT = IV
else:
                CT[i] = AES-CBC-Encrypt(Key, PT)
                PT = CT[i-1]
```

The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-GCM Test

The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

a) Two plaintext lengths. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.
b) Three AAD lengths. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.
c) Two IV lengths. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known

The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the valuator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

FCS_COP.1.1(2)

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature Generation by the TOE using the Signature Generation Test. To conduct this test the evaluator must generate or obtain 10 messages from a trusted reference implementation for each modulus size/SHA combination supported by the TSF. The evaluator shall have the TOE use their private key and modulus value to sign these messages.

The evaluator shall verify the correctness of the TSF's signature using a known good implementation and the associated public keys to verify the signatures.

Signature Verification Test

The evaluator shall perform the Signature Verification test to verify the ability of the TOE to recognize another party's valid and invalid signatures. The evaluator shall inject errors into the test vectors produced during the Signature Verification Test by introducing errors in some of the public keys e, messages, IR format, and/or signatures. The TOE attempts to verify the signatures and returns success or failure.

The evaluator shall use these test vectors to emulate the signature verification test using the corresponding parameters and verify that the TOE detects these errors.

FCS_COP.1.1(3)

The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode

The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode

The evaluators devise an input set consisting of m messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m+ 99*i, where 1 ≤ i ≤ m. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m+ 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.