Klas Telecom
VoyagerTDC 10G Switch
Security Target

18-4188-R-0008

Version: 1.1.2

November 16, 2019

Prepared For:

Klas Telecom, Inc.

1101 30th NW Street

Washington D.C., 20007

Prepared By:

Brad Mitchell

UL Verification Services Inc.

Notices:

Table of Contents

1. Security Target (ST) Introduction

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:	Klas Telecom VoyagerTDC 10G Switch Security Target
ST Version Number:	Version 1.1.2
ST Author(s):	Brad Mitchell
ST Publication Date:	November 16, 2018
Keywords	Network Device

1.2 Target of Evaluation Reference

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

TOE Developer	Klas Telecom, Inc. 1101 30th NW Street Washington DC, 20007
TOE Name:	VoyagerTDC 10G Switch
TOE Version	2.0

1.3 Target of Evaluation Overview

1.3.1 TOE Product Type

The TOE is classified as a Network Device.

1.3.2 TOE Usage

The Klas Voyager TDC Switch running KlasOS firmware provides connectivity to multiple devices into the same network segment. A real-time clock is present on all KlasOS devices. Authentication can be provided locally or over a trusted channel using SSH and all logs can be securely sent to a syslog server. KlasOS provides a Command Line Interface (CLI) for device configuration.

The Klas Voyager range of products provide expandable, enterprise-grade rugged mobility solutions. The Klas Voyager TDC switch is used in a variety of these Klas Voyager products that provide the ability to establish highly secure IPSec tunnels using FIPS approved algorithms. (Note: the Voyager TDC Switch does not provide IPSec functionality itself)

The TOE provides TenGigabit Ethernet, and layer 2 high-speed switching and removable storage using the VIK.

1.3.3 TOE Major Security Features Summary

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

1.3.4 TOE IT environment hardware/software/firmware requirements

An administrative remote console may be used to administer the TOE over SSH. The remote console must implement SSH conformant to RFCs 4251, 4252, 4253, 4254, 5656, and 6668, and providing the following algorithms and parameters:

- Password authentication.
    - Optionally, ECDSA public key authentication using NIST curves P-256 or P-384.
    - Optionally, RSA public key authentication using 2048, 3072, or 4096-bit keys
- AES-CBC-128 or AES-CBC-256 encryption.
- HMAC-SHA1, HMAC-SHA2-256, or HMAC-SHA2-512 for message authentication.
- Key exchange using Diffie-Hellman Group 14, ECDH over NIST P-256, or ECDH over NIST P-384.

The environment must provide an RFC 5424 compliant syslog server. This server must be or be protected by an SSH server as defined above.

An administrative local console must be present to configure the TOE. The local console must provide a DB-9 serial port and be capable of supporting a VT-100 compatible terminal or emulator.

1.4 Target of Evaluation Description

1.4.1 Target of Evaluation Physical Boundaries

The TOE consist of the following hardware:

- KLAS-VOY-TDC-R2.0

Running:

- KlasOS fastnet v5.2.0rc7

KlasOS is based on the Linux 3.10.70 kernel, and includes custom and proprietary modifications to provide the functionality described in this Security Target and address security vulnerabilities.

On:

Marvell Prestera 98DX8212

The Voyager TDC Switch contains 10 10-Gigabit switch ports (4 available as copper or SFP to support fiber connectivity), 1 Gigbit management port, 1 VIK slot (for removable storage), and a console port.

The use of the VIK port was not evaluated as part of the Common Criteria validation.

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

1.4.2 Target of Evaluation Description

The logical boundary of the TOE include those security functions implemented exclusively by the TOE. These security functions are listed 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”.

1.4.2.1 Audit

- The TOE will audit all events and information defined in Table 3: 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 can transmit audit data to an external IT entity using SSH protocol.

1.4.2.2 Cryptographic Operations

The TSF performs the following cryptographic operations:

- SSHv2 using
  - AES-CBC-128 or AES-CBC-256 for encryption;
  - DH Group 14, or NIST P-256 / P-384 for key exchange;
  - HMAC SHA1, HMAC-SHA2-256, or HMAC-SHA2-512 for message authentication;
  - RSA public key authentication using 2048-bit keys & EC public key authentication using NIST P-256 or P-384.

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 supports public key-based authentication methods.
- The TSF requires all administrative-users to authenticate before allowing the user to perform any actions other than:
  - Viewing the warning banner.

1.4.2.4 Security Management

The TOE provides secure administrative services for management of general TOE configuration and the security functionality provided by the TOE. All TOE administration occurs via a local serial console connection or remote SSH session. The TOE provides the ability to securely manage:

- All TOE administrative users, including identification and authentication parameters and credentials
- Timestamps maintained by the TOE
- Update to the TOE

Only one administrative user can be created on the TOE, and the administrative user can perform all of the above security relevant management functions. Administrators can create configurable login banners to be displayed at time of login and can also define an inactivity timeout to terminate sessions after a set period of inactivity.

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, shall terminate the 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 SSH 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 via the trusted path.
- 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.

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 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; Section 8.1, “Operations” 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 performed by the ST author are indicated by a number in parenthesis following the requirement number, e.g., FIA_UAU.1.1(1); the iterated requirement titles are similarly indicated, e.g., FIA_UAU.1(1). Iterations performed by the PP author are indicated by a slash followed by a short description, e.g. FCS_COP.1/Hash.

Assignments are identified with **bold text.**

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

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

2. Conformance Claims

2.1 Common Criteria Conformance Claims

This Security Target is conformant to the Common Criteria Version 3.1r4, 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 2.0 + Errata 20180314, dated March 14, 2018 [cPP]. This Protection Profile will be referred to as cPP or PP for convenience throughout this Security Target.

The TOE conforms with the following Technical Decisions:

Table 1: Applied Technical Decisions

TD	TD Title
TD0339	NIT Technical Decision for Making Password-based authentication optional in FCS_SSHS_EXT.1.2
TD0338	NIT Technical Decision for Access Banner Verification
TD0337	NIT Technical Decision for Selections in FCS_SSH*_EXT.1.6
TD0336	NIT Technical Decision for Audit Requirements for FCS_SSH*_EXT.1.8
TD0334	NIT Technical Decision for Testing SSH when password-based authentication is not supported.
TD0324	NIT Technical Decision for Correction of section numbers in SD Table 1
TD0321	Protection of NTP communications
TD0291	NIT technical decision for DH14 and FCS_CKM.1
TD0290	NIT technical decision for physical interruption of trusted path/channel
TD0260	NIT Technical Decision for Typo in FCS_SSHS_EXT.1.4

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;
    - No additional threats have been defined in this ST.
- Organizational Security Policies

  - All OSP defined in the cPP are carried forward to this ST;
  - No additional OSPs have been defined in this ST.
- Assumptions
  - All assumptions defined in the cPP for a standalone TOE 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 for a standalone TOE are carried forward to this ST. Optional and selection based SFRs defined in the cPP are carried forward to this Security Target as required by the cPP.
- All mandatory 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 section 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.

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 tunnelling 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. a 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 includes replacing existing credentials with an attacker's credentials, modifying 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
An external, unauthorized entity could make use of failed or compromised security functionality and might therefore subsequently use or abuse security functions without prior authentication to access, change or modify device data, critical network traffic or security functionality of the device.

3.2 Organizational Security Policies

The following section 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.

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.

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 a 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 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.

A.RESIDUAL_INFORMATION

The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

4. Security Objectives

4.1 Security Objectives for the Operational Environment

OE.PHYSICAL

Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

OE.NO_GENERAL_PURPOSE

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

OE.NO_THRU_TRAFFIC_PROTECTION

The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.

OE.TRUSTED_ADMIN

Security 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.

OE.RESIDUAL_INFORMATION

The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

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 2: Security Functional Requirements	
SFR	Description
FAU_GEN.1	Audit Data Generation
FAU_GEN.2	User Identity Association
FAU_STG_EXT.1	Protected Audit Event Storage
FCS_CKM.1	Cryptographic Key Generation (Refinement)
FCS_CKM.2	Cryptographic Key Establishment (Refinement)
FCS_CKM.4	Cryptographic Key Destruction
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)
FCS_COP.1/SigGen	Cryptographic Operation (Signature Verification)
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1	Random Bit Generation
FCS_SSHC_EXT.1	SSH Client
FCS_SSHS_EXT.1	SSH Server
FIA_AFL.1	Authentication Failure Management (Refinement)
FIA_PMG_EXT.1	Password Management
FIA_UIA_EXT.1	User Identification and Authentication
FIA_UAU_EXT.2	Password-based Authentication Mechanism
FIA_UAU.7	Protected Authentication Feedback
FMT_MOF.1/Functions	Management of security functions behavior
FMT_MOF.1/ManualUpdate	Management of security functions behavior
FMT_MTD.1/CoreData	Management of TSF Data
FMT_MTD.1/CryptoKeys	Management of TSF data
FMT_SMF.1	Specification of Management Functions
FMT_SMR.2	Restrictions on security roles
FPT_APW_EXT.1	Protection of Administrator Passwords

Table 2: Security Functional Requirements	
SFR	Description
FPT_TST_EXT.1	TSF Testing (Extended)
FPT_TUD_EXT.1	Trusted Update
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
FPT_STM_EXT.1	Reliable Time Stamps
FTA_SSL_EXT.1	TSF-initiated Session Locking
FTA_SSL.3	TSF-initiated Termination (Refinement)
FTA_SSL.4	User-initiated Termination (Refinement)
FTA_TAB.1	Default TOE Access Banners (Refinement)
FTP_ITC.1	Inter-TSF trusted channel (Refinement)
FTP_TRP.1/Admin	Trusted Path (Refinement)

6.1.1 Security Audit (FAU)

6.1.1.1 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1

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

a) Start-up and shut-down of the audit functions;
b) All auditable events for the not specified level of audit; and
c) All administrative actions comprising:
   - Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).
   - Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).
   - Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).
   - Resetting passwords (name of related user account shall be logged).
   - No other actions;
d) Specifically defined auditable events listed in Table 3.

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 4 and Table 5 for optional and selection-based SFRs included in the ST.

For distributed TOEs each component must generate an audit record for each of the SFRs that it implements. If more than one TOE component is involved when an audit event is triggered, the event has to be audited on each component (e.g. rejection of a connection by one component while attempting to establish a secure communication channel between two components should result in an audit event being generated by both components). This is not limited to error cases but also includes events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

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). If the optional SFR FMT_MOF.1/Services is included in the ST, the option "starting and stopping services" needs to be chosen from the selection in FAU_GEN.1.1.

FAU_GEN.1.2

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the cPP/ST, information specified in column three of Table 3.

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 Table 4 and Table 5 for optional and selection-based SFRs included in the ST.

Table 3: Auditable Events

SFR	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG_EXT.1	None.	None.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.
FCS_CKM.4	None.	None.
FCS_COP.1 /DataEncryption	None.	None.
FCS_COP.1 /SigGen	None.	None.
FCS_COP.1 /Hash	None.	None.
FCS_COP.1 /KeyedHash	None.	None.
FCS_RBG_EXT.1	None.	None.
FCS_SSHC_EXT.1	Failure to establish an SSH session	Reason for failure
FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).

Table 3: Auditable Events		
SFR	Auditable Events	Additional Audit Record Contents
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.
FIA_X509_EXT.1/Rev (selection-based)	Unsuccessful attempt to validate a certificate	Reason for failure
FMT_MOF.1/Functions	Modification of the behaviour of the transmission of audit data to an external IT entity, the handling of audit data, the audit functionality when Local Audit Storage Space is full.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	All management activities of TSF data.	None.
FMT_MTD.1/CryptoKeys	Management of cryptographic keys.	None.
FMT_MOF.1/Services	Starting and stopping of services.	None.
FMT_SMF.1	None.	None.
FMT_SMR.2	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_SKP_EXT.1	None.	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.

Table 3: Auditable Events		
SFR	Auditable Events	Additional Audit Record Contents
FTA_TAB.1	None.	None.
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1 /Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.

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 Table 4 and Table 5.

Application Note 39

The audit event for FIA_X509_EXT.1/ITT 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 (according to selection in the ST).*

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

Application Note 49

The audit event for FIA_X509_EXT.1/Rev 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 (according to selections in the ST).*

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

Assurance Activity

TSS

For the administrative task of generating/import of, changing, or deleting of cryptographic keys as defined in FAU_GEN.1.1c, the TSS should identify what information is logged to identify the relevant key.

For distributed TOEs the evaluator shall examine the TSS to ensure it describes which auditable events are generated and recorded by which TOE components. The evaluator shall confirm that all components defined as generating audit information for a particular SFR should also contribute

to that SFR as defined in the mapping of SFRs to TOE components, and that the audit records generated by each component cover all the SFRs that it implements.

Guidance Documentation

The evaluator shall check the guidance documentation 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 cPP 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 audit events.

The evaluator shall also make a determination of the administrative actions related to TSF data related to configuration changes. 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 related to TSF data related to configuration changes. 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.

Tests

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. 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.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of auditable events to TOE components in the Security Target. For all events involving more than one TOE component when an audit event is triggered, the evaluator has to check that the event has been audited on both sides (e.g. failure of building up a secure communication channel between the two components). This is not limited to error cases but includes also events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

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.

Application Note 5

Where an auditable event is triggered by another component, the component that records the event must associate the event with the identity of the initiating component that caused the event (applies to distributed TOEs only).

Assurance Activity

Tests

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

For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one test on one component where another component instigates an auditable event. The evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

6.1.1.3 FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1

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

Application Note 6

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. Since the external audit server is not part of the TOE, there are no requirements on it except the capabilities for FTP_ITC.1 transport for audit data. No requirements are placed upon the format or underlying protocol of the audit data being transferred. The TOE must be capable of being configured to transfer audit data to an external IT entity without Administrator intervention. Manual transfer would not meet the requirements. Transmission could be done in real-time or periodically. If the transmission is not done in real-time then the TSS describes what event stimulates the transmission to be made and what range of frequencies the TOE supports for making transfers of audit data to the audit server; the TSS also suggests typical acceptable frequencies for the transfer.

For distributed TOEs each component must be able to export audit data across a protected channel external (FTP_ITC.1) or intercomponent (FPT_ITT.1 or FTP_ITC.1) as appropriate. At least one component of the TOE must be able to export audit records via FTP_ITC.1 such that all TOE audit records can be exported to an external IT entity.

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: **Delete the current logfile and start a new one** when the local storage space for audit data is full.

Application Note 7

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".

For distributed TOEs each component is not required to store generated audit data locally but the overall TOE needs to be able to store audit data locally. Each component must at least provide the ability to temporarily buffer audit information locally to ensure that audit records are preserved in case of network connectivity issues (for details see also chap. 6.3.3). Buffering audit information locally, does not necessarily involve non-volatile memory: audit information could be buffered in volatile memory. However, the local storage of audit information in the sense of FAU_STG_EXT.1.3 needs to be done in non-volatile memory. For every component which performs local storage of audit information, the behaviour when local storage is exhausted needs to be described. For every component which is buffering audit information instead of storing audit information locally itself, it needs to be described what happens in case the buffer space is exhausted.

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.

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.

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit information to an external IT entity can be done in real- time or periodically. In case the TOE does not perform transmission in real- time the evaluator needs to verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible as well as acceptable frequency for the transfer of audit data.

For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how audit data transfer to the external audit server is implemented among the different TOE components (e.g. every TOE components does its own transfer or the data is sent to another TOE component for central transfer of all audit events to the external audit server).

For distributed TOEs the evaluator shall examine the TSS to ensure it describes which TOE components are storing audit information locally and which components are buffering audit information and forwarding the information to another TOE component for local storage. For every component the TSS shall describe the behaviour when local storage space or buffer space is exhausted.

Guidance Documentation

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.

Tests

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 tests 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 verify that the TOE is capable of transferring audit data to an external audit server automatically without administrator intervention.
b) Test 2: 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
    1) 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).
    2) 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)
    3) The TOE behaves as specified (for the option ‘other action’ in FAU_STG_EXT.1.3).
c) Test 3: If the TOE complies with FAU_STG_EXT.2/LocSpace the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2/LocSpace are correct when performing the tests for FAU_STG_EXT.1.3
d) Test 4: For distributed TOEs, Test 1 defined above should be applicable to all TOE components that forward audit data to an external audit server. For the local storage according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3 the Test 2 specified above shall be applied to all TOE components that store audit data locally. For all TOE components that store audit data locally and comply with FAU_STG_EXT.2/LocSpace Test 3 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

6.1.2 Cryptographic Support (FCS)

6.1.2.1 FCS_CKM.1 Cryptographic Key Generation (Refinement) [TD0291[1]]

FCS_CKM.1.1

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

- RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;
- ECC schemes using "NIST curves" P-256, P-384 that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;
- FFC Schemes using Diffie-Hellman group 14 that meet the following: RFC 3526, Section 3

Application Note 8

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, other than ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384 and ecdsa-sha2-nistp521, the public key is expected to be associated with an X.509v3 certificate.

If the TOE acts as a receiver in the key establishment schemes and is not configured to support mutual authentication, the TOE does not need to implement key generation.

In a distributed TOE, if the TOE component acts as a receiver in the key establishment scheme, the TOE does not need to implement 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 Documentation

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 cryptographic protocols defined in the Security Target.

Tests

See Annex A.

6.1.2.2 FCS_CKM.2 Cryptographic Key Establishment (Refinement)

FCS_CKM.2.1

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

[1] FCS_CKM.1.1 SFR text was modified by TD0291.

- Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";
- Key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526, Section 3;

~~that meets the following:~~ **~~ECC as specified in NIST SP 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography", DH as specified in RFC 3526 Section 3.~~**

Application Note 9

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. For Diffie-Hellman group 14, ST authors should make the corresponding selection from the SFR instead of using the Finite field-based key establishment selection.

The RSA-based key establishment schemes are described in Section 9 of NIST SP 800-56B Revision 1; however, Section 9 relies on implementation of other sections in SP 800-56B Revision 1.

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 (including whether the TOE acts as a sender, a recipient, or both). If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall describe how the implementation meets RFC 3526 Section 3.

Guidance Documentation

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

Tests

See Annex A.

6.1.2.3 FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1

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

- For plaintext keys in volatile storage, the destruction shall be executed by a single overwrite consisting of zeroes;
- For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that

- logically addresses the storage location of the key and performs a single-pass overwrite consisting of a new value of the key;
- instructs a part of the TSF to destroy the abstraction that represents the key.

that meets the following: No Standard.

Application Note 10

In parts of the selections where keys are identified as being destroyed by "a part of the TSF", the TSS identifies the relevant part and the interface involved. The interface referenced in the requirement could take different forms for different TOEs, the most likely of which is an application programming interface to an OS kernel. There may be various levels of abstraction visible. For instance, in a given implementation the application may have access to the file system details and may be able to logically address specific memory locations. In another implementation the application may simply have a handle to a resource and can only ask another part of the TSF such as the interpreter or OS to delete the resource.

Where different key destruction methods are used for different keys and/or different destruction situations then the different methods and the keys/situations they apply to are described in the TSS (and the ST may use separate iterations of the SFR to aid clarity). The TSS describes all relevant keys used in the implementation of SFRs, including cases where the keys are stored in a non-plaintext form. In the case of non-plaintext storage, the encryption method and relevant key-encrypting-key are identified in the TSS.

Some selections allow assignment of "a value that does not contain any CSP". This means that the TOE uses some specified data not drawn from an RBG meeting FCS_RBG_EXT requirements, and not being any of the particular values listed as other selection options. The point of the phrase "does not contain any CSP" is to ensure that the overwritten data is carefully selected, and not taken from a general pool that might contain current or residual data that itself requires confidentiality protection.

For the avoidance of doubt: the "cryptographic keys" in this SFR include session keys. Key destruction does not apply to the public component of asymmetric key pairs.

Assurance Activity

TSS

The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support

FPT_APW_EXT.1 and FPT_SKP_EXT.1, are accounted for[2]). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

[2] Where keys are stored encrypted or wrapped under another key then this may need to be explained in order to allow the evaluator to confirm the consistency of the description of keys with the TOE functions.

The evaluator shall check to ensure the TSS identifies how the TOE destroys keys stored as plaintext in non-volatile memory, and that the description includes identification and description of the interfaces that the TOE uses to destroy keys (e.g., file system APIs, key store APIs).

Note that where selections involve 'destruction of reference' (for volatile memory) or 'invocation of an interface' (for non-volatile memory) then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory the evaluator includes in their examination the relevant interface description for each media type on which plaintext keys are stored. The presence of OS-level and storage device-level swap and cache files is not examined in the current version of the Evaluation Activity.

Where the TSS identifies keys that are stored in a non-plaintext form, the evaluator shall check that the TSS identifies the encryption method and the key-encrypting-key used, and that the key-encrypting-key is either itself stored in an encrypted form or that it is destroyed by a method included under FCS_CKM.4.

The evaluator shall check that the TSS identifies any configurations or circumstances that may not conform to the key destruction requirement (see further discussion in the Guidance Documentation section below). Note that reference may be made to the Guidance Documentation for description of the detail of such cases where destruction may be prevented or delayed.

Where the ST specifies the use of "a value that does not contain any CSP" to overwrite keys, the evaluator examines the TSS to ensure that it describes how that pattern is obtained and used, and that this justifies the claim that the pattern does not contain any CSPs.

Guidance Documentation

A TOE may be subject to situations that could prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS (and any other supporting information used). The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command[3] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

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

FCS_COP.1.1/DataEncryption

The TSF shall perform encryption/decryption in accordance with a specified cryptographic algorithm AES used in CBCmode 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 11

[3] Where TRIM is used then the TSS and/or guidance documentation is also expected to describe how the keys are stored such that they are not inaccessible to TRIM, (e.g. they would need not to be contained in a file less than 982 bytes which would be completely contained in the master file table).

For the first selection of FCS_COP.1.1/DataEncryption, the ST author chooses the mode or modes in which AES operates. For the second selection, the ST author chooses the key sizes that are supported by this functionality. The modes and key sizes selected here correspond to the cipher suite selections made in the trusted channel requirements.

Assurance Activity

Tests

See Appendix A

6.1.2.5 FCS_COP.1/SigGen Cryptographic Operation (Signature Verification)

FCS_COP.1.1/SigGen

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

- RSA Digital Signature Algorithm and cryptographic key sizes (modulus) **2048, 3072, 4096 bits**,
- Elliptic Curve Digital Signature Algorithm and cryptographic key sizes **256, 384**

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; ISO/IEC 14888-3, Section 6.4.

Application Note 12

The ST Author chooses the algorithm(s) implemented to perform digital signatures. For the algorithm(s) chosen, the ST author makes the appropriate assignments/selections to specify the parameters that are implemented for that algorithm. The ST author ensures that the assignments and selections for this SFR include all the parameter values necessary for the cipher suites selected for the protocol SFRs (see Appendix B.2.1) that are included in the ST. The ST Author checks for consistency of selections with other FCS requirements, especially when supporting elliptic curves.

Assurance Activity

Tests

See Annex A.

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

FCS_COP.1.1/Hash

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

Application Note 13

Vendors are strongly encouraged to implement updated protocols that support the SHA-2 family; until updated protocols are supported, this cPP allows support for SHA-1 implementations in

compliance with SP 800-131A. In a future version of this cPP, SHA-256 will be the minimum requirement for all TOEs.

The hash selection should be consistent with the overall strength of the algorithm used for FCS_COP.1/DataEncryption and FCS_COP.1/SigGen (for example, SHA 256 for 128-bit keys).

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 Documentation

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

Tests

See Annex A.

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

FCS_COP.1.1/KeyedHash

The TSF shall perform keyed-hash message authentication in accordance with a specified cryptographic algorithm HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512 and cryptographic key sizes **160, 256, and512 bits** and message digest sizes 160, 256, 512 bits that meet the following: ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”.

Application Note 14

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.

Tests

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 message data using a known good implementation.

6.1.2.8 FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1

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

FCS_RBG_EXT.1.2

The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from **5** software-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 15

For the first selection in FCS_RBG_EXT.1.2, the ST author 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 should be repeated to 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- 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, which must be equal or greater than the security strength of any key generated by the TOE.

Assurance Activity

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

TSS

The evaluator shall examine the TSS to determine that it specifies the DRBG type, identifies the entropy source(s) seeding the DRBG, and state the assumed or calculated min-entropy supplied either separately by each source or the min- entropy contained in the combined seed value.

Guidance Documentation

The evaluator shall confirm that the guidance documentation contains appropriate instructions for configuring the RNG functionality.

Tests

The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration.

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.

6.1.2.9 FCS_SSHC_EXT.1 SSH Client [TD0259[4], TD0336[5], TD0337[6]]

FCS_SSHC_EXT.1.1

The TSF shall implement the SSH protocol that complies with RFC(s) 4251, 4252, 4253, 4254, 5656, 6668.

Application Note 90

The ST author selects which of the 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 evaluation activity for this requirement.

RFC 5647 only applies to the RFC compliant implementation of GCM; a TOE that only implements the "@openssh.com" variant of GCM should not select 5647. aes-gcm@openssh.com is specified in Section 1.6 of the OpenSSH Protocol Specification (https://cvsweb.openbsd.org/cgi-bin/cvsweb/src/usr.bin/ssh/PROTOCOL?rev=1.31)*

FCS_SSHC_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, no other method.

Assurance Activity

[4] The SFR text and application note for FCS_SSHC_EXT.1.5 was modified by TD0259. The Application Note for FCS_SSHC_EXT.1.9 was modified by TD0259.

[5] The Assurance activities for FCS_SSHC_EXT.1.8 were modified by TD0336

[6] The SFR text, application notes, and assurance activities for FCS_SSHC_EXTS.1.1, FC_SSHC_EXT.1.4, and FCS_SSHC_EXT.1.6 were modified by TD0337.

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 and that this list conforms to FCS_SSHC_EXT.1.5 and ensure that if password-based authentication methods have been selected in the ST then these are also described.

Tests

Test 1: If password-based authentication methods have been selected in the ST then using the guidance documentation, the evaluator shall configure the TOE to perform password-based authentication to an SSH server, and demonstrate that a user can be successfully authenticated by the TOE to an SSH server using a password as an authenticator.

Note: Public key authentication is tested as part of testing for FCS_SSHC_EXT.1.5

FCS_SSHC_EXT.1.3

The TSF shall ensure that, as described in RFC 4253, packets greater than **33,292** bytes in an SSH transport connection are dropped.

Application Note 91

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.

Tests

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

FCS_SSHC_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.

Application Note 92

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. Corresponding FCS_COP entries are included in the ST for the algorithms selected here.

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 Documentation

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).

Tests

The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish a SSH connection. To verify this, the evaluator shall start session establishment for a SSH connection with a remote server (referred to as 'remote endpoint' below). The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the captured traffic that the TOE offers all the ciphers defined in the TSS for the TOE for SSH sessions, but no additional ones compared to the definition in the TSS. The evaluator shall perform one successful negotiation of an SSH session to verify that the TOE behaves as expected. It is sufficient to observe the successful negotiation of the session to satisfy the intent of the test. If the evaluator detects that not all ciphers defined in the TSS for SSH are supported by the TOE and/or the TOE supports one or more additional ciphers not defined in the TSS for SSH, the test shall be regarded as failed.

FCS_SSHC_EXT.1.5

The TSF shall ensure that the SSH public-key based authentication implementation uses ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, as its public key algorithm(s) and rejects all other public key algorithms.

Application Note 93

If x509v3-ssh-rsa, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521 or x509v3-rsa2048-sha256 are selected, then the list of trusted certification authorities must be selected in FCS_SSHC_EXT.1.9 and the FIA_X509_EXT SFRs in Appendix B are applicable.

It is recommended to configure the TOE to reject presented RSA keys with a key length below 2048 bit.

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 Documentation

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).

Tests

Test 1: The evaluator shall establish a SSH connection using each of the public key algorithms specified by the requirement to authenticate an SSH server to the TOE. 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 server to only allow a public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.

FCS_SSHC_EXT.1.6

The TSF shall ensure that the SSH transport implementation uses hmac-sha1, hmac-sha2-256, hmac-sha2-512 as its data integrity MAC algorithm(s) and rejects all other MAC algorithm(s).

Application Note 94

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.

The ST author selects "implicit" when, and only when, aes-gcm@openssh.com is selected as an encryption algorithm. When aes*-gcm@openssh.com is negotiated as the encryption algorithm, the MAC algorithm field is ignored and GCM is implicitly used as the MAC. "implicit" is not an SSH algorithm identifier and will not be seen on the wire; however, the negotiated MAC might be decoded as "implicit".*

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 Documentation

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).

Tests

Test 1: (conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST) The evaluator shall establish an SSH connection using each of the algorithms, except "implicit", 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.

Note: To ensure the observed algorithm is used, the evaluator shall ensure a non-aes*-gcm@openssh.com encryption algorithm is negotiated while performing this test.

Test 2: (conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST) The evaluator shall configure an SSH server to only allow a MAC algorithm that is not included in the ST selection. The evaluator shall attempt to connect from the TOE to the SSH server and observe that the attempt fails.

Note: To ensure the proposed MAC algorithm is used, the evaluator shall ensure a non-aes*-gcm@openssh.com encryption algorithm is negotiated while performing this test.

FCS_SSHC_EXT.1.7

The TSF shall ensure that diffie-hellman-group14-sha1, ecdh-sha2-nistp256 and ecdh-sha2-nistp384 are the only allowed key exchange methods used for the SSH protocol.

Assurance Activity

TSS

The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms, and that that list corresponds to the list in this component.

Guidance Documentation

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.

Tests

Test 1: The evaluator shall configure an SSH server to permit all allowed key exchange methods. The evaluator shall attempt to connect from the TOE to the SSH server using each allowed key exchange method, and observe that each attempt succeeds.

FCS_SSHC_EXT.1.8[7]

The TSF shall ensure that within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

Application Note 95

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.

Assurance Activity

TSS

The evaluator shall check that the TSS specifies the following:

1. Both thresholds are checked by the TOE.
2. Rekeying is performed upon reaching the threshold that is hit first.

Guidance Documentation

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, then the evaluator shall check that the guidance documentation describes how to configure those thresholds. Either the allowed values are specified in the guidance documentation and must not exceed the limits specified in the SFR (one hour of session time, one gigabyte of transmitted traffic) or the TOE must not accept values beyond the limits specified in the SFR. The evaluator shall check that the guidance documentation describes that the TOE reacts to the first threshold reached.

[7] The assurance activities for this SFR were modified by TD0336

Tests

The evaluator needs to perform testing that rekeying is performed according to the description in the TSS. The evaluator shall test both, the time-based threshold and the traffic-based threshold.

For testing of the time-based threshold the evaluator shall use the TOE to connect to an SSH server and keep the session open until the threshold is reached. The evaluator shall verify that the SSH session has been active longer than the threshold value and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one hour of sesion time but the value used for testing shall not exceed one hour. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH server the TOE is connected to.

For testing of the traffic-based threshold the evaluator shall use the TOE to connect to an SSH server, and shall transmit data from and to the TOE within the active SSH session until the threshold for transmitted traffic is reached. The transmitted traffic is the total traffic comprising incoming and outgoing traffic.

The evaluator shall verify that more data has been transmitted within the SSH session than the threshold allows and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one gigabyte o transferred traffic but the value used for testing shall not exceed one gigabyte. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH server the TOE is connected to.

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, the evaluator needs to verify that the threshold(s) can be configured as described in the guidance documentation and the evaluator needs to test that modification of the thresholds is restricted to Security Administrators (as required by FMT_MOF.1/Functions).

FCS_SSHC_EXT.1.9

The TSF shall ensure that the SSH client authenticates the identity of the SSH server using a local database associating each host name with its corresponding public key or no other methods as described in RFC 4251 section 4.1.

Application Note 96

The list of trusted certification authorities can only be selected if x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384 or x509v3-ecdsa-sha2-nistp521 are selected in FCS_SSHC_EXT.1.5.

Assurance Activity

Tests[8]

Test 1: The evaluator shall delete all entries in the TOE's list of recognized SSH server host keys and, if selected, all entries in the TOE's list of trusted certification authorities. The evaluator shall initiate a connection from the TOE to an SSH server. The evaluator shall ensure that the TOE either rejects the connection or displays the SSH server's public key (either the key bytes

[8] The Assurance Activities for FCS_SSHC_EXT.1.9 were modified by TD0334

themselves or a hash of the key using any allowed hash algorithm) and prompts the user to accept or deny the key before continuing the connection.

Test 2: "The evaluator shall add an entry associating a host name with a public key into the TOE's local database. The evaluator shall replace, on the corresponding SSH server, the server's host key with a different host key. If 'password-based' is selected for the TOE in FCS_SSHC_EXT.1.2, the evaluator shall initiate a connection from the TOE to the SSH server using password-based authentication, shall ensure that the TOE rejects the connection, and shall ensure that the password was not transmitted to the SSH server (for example, by instrumenting the SSH server with a debugging capability to output received passwords). If 'password-based' is not selected for the TOE in FCS_SSHC_EXT.1.2, the evaluator shall initiate a connection from the TOE to the SSH server using public key-based authentication, and shall ensure that the TOE rejects the connection."

6.1.2.10 FCS_SSHS_EXT.1 SSH Server [TD0259[9], TD0260[10], TD0336[11], TD0337[12], TD0339[13]]]

FCS_SSHS_EXT.1.1

The TSF shall implement the SSH protocol that complies with RFC(s) 4251, 4252, 4253, 4254, 5656, 6668.

Application Note 97

The ST author selects which of the 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 evaluation activity for this requirement.

RFC 5647 only applies to the RFC compliant implementation of GCM; a TOE that only implements the "@openssh.com" variant of GCM should not select 5647. aes-gcm@openssh.com is specified in Section 1.6 of the OpenSSH Protocol Specification (https://cvsweb.openbsd.org/cgi-bin/cvsweb/src/usr.bin/ssh/PROTOCOL?rev=1.31).*

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.

Application Note

[9] The SFR text and application note for FCS_SSHS_EXT.1.5 was modified by TD0259.

[10] The typo in FCS_SSHS_EXT.1.4 was fixed by TD0260.

[11] The assurance activities for FCS_SSHS_EXT.1.8 were modified by TD0336

[12] The SFR text, application notes, and assurance activities for FCS_SSHS_EXT.1.1, FCS_SSHS_EXT.1.4, and FCS_SSHS_EXT.1.6 were modified by TD0337

[13] The SFR text, application notes, and assurance activities for FCS_SSHS_EXT.1.2 were modified by TD0339

If the TOE supports password-based authentication, the option 'password-based' shall be selected. If the TOE supports only public key-based authentication, the option 'no other method' shall be chosen.

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 and that this list conforms to FCS_SSHS_EXT.1.5. and ensure that if password-based authentication methods have been selected in the ST then these are also described.

Tests

Test 1: If password-based authentication methods have been selected in the ST then using the guidance documentation, the evaluator shall configure the TOE to accept password-based authentication, and demonstrate that user authentication succeeds when the correct password is provided by the user.

Test 2: If password-based authentication methods have been selected in the ST then the evaluator shall use an SSH client, enter an incorrect password to attempt to authenticate to the TOE, and demonstrate that the authentication fails.

Note: Public key authentication is tested as part of testing for FCS_SSHS_EXT.1.5

t**FCS_SSHS_EXT.1.3**

The TSF shall ensure that, as described in RFC 4253, packets greater than **33,292** bytes in an SSH transport connection are dropped.

Application Note 98

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.

Tests

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.

Application Note 99

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. Corresponding FCS_COP entries are included in the ST for the algorithms selected here.

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 Documentation

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).

Tests

The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish a SSH connection. To verify this, the evaluator shall start session establishment for a SSH connection from a remote client (referred to as 'remote endpoint' below). The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the captured traffic that the TOE offers all the ciphers defined in the TSS for the TOE for SSH sessions, but no additional ones compared to the definition in the TSS. The evaluator shall perform one successful negotiation of an SSH session to verify that the TOE behaves as expected. It is sufficient to observe the successful negotiation of the session to satisfy the intent of the test. If the evaluator detects that not all ciphers defined in the TSS for SSH are supported by the TOE and/or the TOE supports one or more additional ciphers not defined in the TSS for SSH, the test shall be regarded as failed.

FCS_SSHS_EXT.1.5

The TSF shall ensure that the SSH public-key based authentication implementation uses ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384 as its public key algorithm(s) and rejects all other public key algorithms.

Application Note 100

If x509v3-ssh-rsa, x509v3-ecdsa-sha2-nistp256, x509v3-ecdsa-sha2-nistp384, x509v3-ecdsa-sha2-nistp521 or x509v3-rsa2048-sha256 are selected, then the FIA_X509_EXT SFRs in Appendix B are applicable.

It is recommended to configure the TOE to reject presented RSA keys with a key length below 2048 bit.

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 Documentation

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).

Tests

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 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: The evaluator shall configure an SSH client to only allow a public key algorithm that is not included in the ST selection. 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, hmac-sha2-256, hmac-sha2-512 as its MAC algorithm(s) and rejects all other MAC algorithm(s).

Application Note 101

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.

The ST author selects "implicit" when, and only when, aes-gcm@openssh.com is selected as an encryption algorithm. When aes*-gcm@openssh.com is negotiated as the encryption algorithm, the MAC algorithm field is ignored and GCM is implicitly used as the MAC. "implicit" is not an SSH algorithm identifier and will not be seen on the wire; however, the negotiated MAC might be decoded as "implicit".*

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 Documentation

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).

Tests

Test 1: (conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST) The evaluator shall establish an SSH connection using each of the algorithms, except "implicit", 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.

Note: To ensure the observed algorithm is used, the evaluator shall ensure a non-aes*-gcm@openssh.com encryption algorithm is negotiated while performing this test.

Test 2: (conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST) The evaluator shall configure an SSH client to only allow a MAC algorithm that is not included in the

ST selection. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

Note: To ensure the proposed MAC algorithm is used, the evaluator shall ensure a non-aes*-gcm@openssh.com encryption algorithm is negotiated while performing this test.

FCS_SSHS_EXT.1.7

The TSF shall ensure that diffie-hellman-group14-sha1, ecdh-sha2-nistp256 and ecdh-sha2-nistp384 are the only allowed key exchange methods used for the SSH protocol.

Assurance Activity

TSS

The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms, and that that list corresponds to the list in this component.

Guidance Documentation

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.

Tests

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[14]

The TSF shall ensure that within SSH connections the same session keys are used for a threshold of no longer than one hour, and no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

Application Note 102

This SFR defines two thresholds - one for the maximum time span the same session keys can be used and the other one for the maximum amount of data that can be transmitted using the same session keys. Both thresholds need to be implemented and a rekey needs to be performed on whichever threshold is reached first. For the maximum transmitted data threshold, the total incoming and outgoing data needs to be counted. The rekey applies to all session keys (encryption, integrity protection) for incoming and outgoing traffic.

It is acceptable for a TOE to implement lower thresholds than the maximum values defined in the SFR.

For any configurable threshold related to this requirement the guidance documentation needs to specify how the threshold can be configured. The allowed values must either be specified in the guidance documentation and must be lower or equal to the thresholds specified in this SFR or the TOE must not accept values beyond the thresholds specified in this SFR.

[14] Assurance activities for this SFR were modified by TD0336

Assurance Activity

TSS

The evaluator shall check that the TSS specifies the following:

1. Both thresholds are checked by the TOE.
2. Rekeying is performed upon reaching the threshold that is hit first.

Guidance Documentation

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, then the evaluator shall check that the guidance documentation describes how to configure those thresholds. Either the allowed values are specified in the guidance documentation and must not exceed the limits specified in the SFR (one hour of session time, one gigabyte of transmitted traffic) or the TOE must not accept values beyond the limits specified in the SFR. The evaluator shall check that the guidance documentation describes that the TOE reacts to the first threshold reached.

Tests

The evaluator needs to perform testing that rekeying is performed according to the description in the TSS. The evaluator shall test both, the time-based threshold and the traffic-based threshold.

For testing of the time-based threshold the evaluator shall use an SSH client to connect to the TOE and keep the session open until the threshold is reached. The evaluator shall verify that the SSH session has been active longer than the threshold value and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one hour of sesion time but the value used for testing shall not exceed one hour. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH client that is connected to the TOE.

For testing of the traffic-based threshold the evaluator shall use an SSH client to connect to the TOE, and shall transmit data from and to the TOE within the active SSH session until the threshold for transmitted traffic is reached. The transmitted traffic is the total traffic comprising incoming and outgoing traffic.

The evaluator shall verify that more data has been transmitted within the SSH session than the threshold allows and shall verify that the TOE initiated a rekey (the method of verification shall be reported by the evaluator).

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one gigabyte o transferred traffic but the value used for testing shall not exceed one gigabyte. The evaluator needs to ensure that the rekeying has been initiated by the TOE and not by the SSH client that is connected to the TOE.

If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, the evaluator needs to verify that the threshold(s) can be configured as described in the guidance documentation and the evaluator needs to test that modification of the thresholds is restricted to Security Administrators (as required by FMT_MOF.1/Functions).

6.1.3 Identification and Authentication (FIA)

6.1.3.1 FIA_AFL.1 Authentication Failure Management (Refinement)

FIA_AFL.1.1

The TSF shall detect when an Administrator configurable positive integer within **1-255** unsuccessful authentication attempts occur related to Administrators attempting to authenticate remotely.

FIA_AFL.1.2

When the defined number of unsuccessful authentication attempts has been met, the TSF shall prevent the offending remote Administrator from successfully authenticating until **manual account unlocking** is taken by a local Administrator.

Application Note 16

This requirement applies to a defined number of successive unsuccessful authentication attempts and does not apply to an Administrator at the local console, since it does not make sense to lock a local Administrator's account in this fashion. This could be addressed by (for example) requiring a separate account for local Administrators or having the authentication mechanism implementation distinguish local and remote login attempts. The "action" taken by a local Administrator is implementation specific and would be defined in the Administrator guidance (for example, lockout reset or password reset). The ST author chooses one of the selections for handling of authentication failures depending on how the TOE has implemented this handler.

The TSS describes how the TOE ensures that authentication failures by remote Administrators cannot lead to a situation where no Administrator access is available, either permanently or temporarily (e.g. by providing local logon which is not subject to blocking). The Operational Guidance describes, and identifies the importance of, any actions that are required in order to ensure that Administrator access will always be maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

Assurance Activity

TSS

The evaluator shall examine the TSS to determine that it contains a description, for each supported method for remote administrative actions, of how successive unsuccessful authentication attempts are detected and tracked. The TSS shall also describe the method by which the remote administrator is prevented from successfully logging on to the TOE, and the actions necessary to restore this ability.

The evaluator shall examine the TSS to confirm that the TOE ensures that authentication failures by remote administrators cannot lead to a situation (e.g. by providing local logon which is not subject to blocking).

Guidance Documentation

The evaluator shall examine the guidance documentation to ensure that instructions for configuring the number of successive unsuccessful authentication attempts and time period (if implemented) are provided, and that the process of allowing the remote administrator to once again successfully log on is described for each "action" specified (if that option is chosen). If different actions or mechanisms are implemented depending on the secure protocol employed (e.g., TLS vs. SSH), all must be described.

The evaluator shall examine the guidance documentation to confirm that it describes, and identifies the importance of, any actions that are required in order to ensure that administrator access will always be maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

Tests

The evaluator shall perform the following tests for each method by which remote administrators access the TOE (e.g. any passwords entered as part of establishing the connection protocol or the remote administrator application):

a) Test 1: The evaluator shall use the operational guidance to configure the number of successive unsuccessful authentication attempts allowed by the TOE (and, if the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall also use the operational guidance to configure the time period after which access is re-enabled). The evaluator shall test that once the authentication attempts limit is reached, authentication attempts with valid credentials are no longer successful.
b) Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows.

   If the administrator action selection in FIA_AFL.1.2 is included in the ST then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

   If the time period selection in FIA_AFL.1.2 is included in the ST then the evaluator shall wait for just less than the time period configured in Test 1 and show that an authorisation attempt using valid credentials does not result in successful access. The evaluator shall then wait until just after the time period configured in Test 1 and show that an authorisation attempt using valid credentials results in successful access.

6.1.3.2 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 configurable to **15** and **128**.

Application Note 17

The ST author selects the special characters that are supported by the 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.

The second assignment should be configured with the largest minimum password length the Security Administrator can configure.

Assurance Activity

Guidance Documentation

The evaluator shall examine the guidance documentation to determine that it:

a) identifies the characters that may be used in passwords and provides guidance to security administrators on the composition of strong passwords, and
b) provides instructions on setting the minimum password length and describes the valid minimum password lengths supported.

Tests

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, and a minimum length listed in the requirement are supported, and justify the subset of those characters chosen for testing.

6.1.3.3 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;
- **Respond to ICMP requests;**
- **Send and receive DNS request;**
- **Send and receive ARP requests;**
- **Send and receive SSH packets;**

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 18

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 (an audit 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.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

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".

The evaluator shall examine the TSS to determine that it describes which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration.

For distributed TOEs the evaluator shall examine that the TSS details how Security Administrators are authenticated and identified by all TOE components. If not all TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the TSS shall describe how the overall TOE functionality is split between TOE components including how it is ensured that no unauthorized access to any TOE component can occur.

For distributed TOEs, the evaluator shall examine the TSS to determine that it describes for each TOE component which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration. For each TOE component that does not support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 the TSS shall describe any unauthenticated services/services that are supported by the component.

Guidance Documentation

The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps (e.g., establishing credential material such as pre- shared keys, tunnels, certificates, etc.) to logging in are described. For each supported the login method, the evaluator shall ensure the 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.

Tests

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.
d) Test 4: For distributed TOEs where not all TOE components support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the evaluator shall test that the components authenticate Security Administrators as described in the TSS.

6.1.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1

The TSF shall provide a local password-based authentication mechanism, and no other authentication mechanism to perform local administrative user authentication.

Application Note 19

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/Admin.

According to the application note for FMT_SMR.2, for distributed TOEs at least one TOE component has to support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 but not necessarily all TOE components. In case not all TOE components support this way of authentication for Security Administrators the TSS shall describe how Security Administrators are authenticated and identified.

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.

6.1.3.5 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 20

"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

Tests

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.

6.1.4 Security Management (FMT)

Assurance Activity

General requirements for distributed TOEs

TSS

For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Guidance Documentation

For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

Tests

Tests defined to verify the correct implementation of security management functions shall be performed for every TOE component. For security management functions that are implemented centrally, sampling should be applied when defining the evaluator's tests (ensuring that all components are covered by the sample).

6.1.4.1 FMT_MOF.1/Functions Management of security functions behavior

FMT_MOF.1.1/Functions

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, handling of audit data to Security Administrators.

Application Note 126

FMT_MOF.1/Functions should be chosen if one or more of the following scenarios apply:

- *If the transmission protocol for transmission of audit data to an external IT entity as defined in FAU_STG_EXT.1.1 is configurable, "transmission of audit data to an external IT entity" shall be chosen.*
- *If the handling of audit data is configurable, "handling of audit data" shall be chosen. 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/LocSpace.*
- *If the behaviour of the audit functionality is configurable when Local Audit Storage Space is full, "audit functionality when Local Audit Storage Space is full" shall be chosen.*

The first selection for "determine the behaviour of" and "modify the behaviour of" should be done as appropriate. It might be necessary to have different selections for the first selection depending on the second selection (e.g. "handling of audit data" might require "determine the behaviour of" and "modify the behaviour of" for the first selection on the one hand and "TOE Security Functions" might require "modify the behaviour of" only). In that case FMT_MOF.1/Functions should be iterated with increasing number appended (i.e. FMT_MOF.1/Functions1, FMT_MOF.1/Functions2, etc.).

Assurance Activity

TSS

For distributed TOEs see chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Tests

Test 1 (if 'transmission of audit data to external IT entity' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity without prior authentication as security administrator (by authentication as a user with no administrator privileges or without user authentication at all). Attempts to modify parameters without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to modify the security related parameters can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

Test 2 (if 'transmission of audit data to external IT entity' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related 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 have to test all possible values of the security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity but at least one allowed value per parameter.

Test 1 (if 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the handling of audit data without prior authentication as security administrator (by authentication as a user with no administrator privileges or without user authentication at all). Attempts to modify parameters without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator. 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/LocSpace.

Test 2 (if 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related 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/LocSpace.

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

Test 1 (if 'audit functionality when Local Audit Storage Space is full' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify the behaviour when Local Audit Storage Space is full without prior authentication as security administrator (by authentication as a user with no administrator privileges or without user authentication at all). This attempt should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

Test 2 (if 'audit functionality when Local Audit Storage Space is full' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify the behaviour when Local Audit Storage Space is full with prior authentication as security administrator. This attempt should be successful. The effect of the change shall be verified.

The evaluator does not necessarily have to test all possible values for the behaviour when Local Audit Storage Space is full but at least one change between allowed values for the behaviour.

Test 3 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen from the second selection without prior authentication as security administrator (by authentication as a user with no administrator privileges or without user authentication at all). This can be done in one test or in separate tests. The attempt(s) to determine the behaviour of the selected functions without administrator authentication shall fail. According to the implementation no other users than the Security Administrator might be defined and without any

user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

Test 4 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen from the second selection with prior authentication as security administrator. This can be done in one test or in separate tests. The attempt(s) to determine the behaviour of the selected functions with administrator authentication shall be successful.

6.1.4.2 FMT_MOF.1/ManualUpdate Management of security functions behaviour

FMT_MOF.1.1/ManualUpdate

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

Application Note 21

FMT_MOF.1/ManualUpdate restricts the initiation of manual updates to Security Administrators.

Assurance Activity

TSS

For distributed TOEs see chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Guidance Documentation

The evaluator shall examine the guidance documentation to determine that any necessary steps to perform manual update are described. The guidance documentation shall also provide warnings regarding functions that may cease to operate during the update (if applicable).

For distributed TOEs the guidance documentation shall describe all steps how to update all TOE components. This shall contain description of the order in which components need to be updated if the order is relevant to the update process. The guidance documentation shall also provide warnings regarding functions of TOE components and the overall TOE that may cease to operate during the update (if applicable).

Tests

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). The attempt to update the TOE shall fail.

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

6.1.4.3 FMT_MTD.1/CoreData Management of TSF Data

FMT_MTD.1.1/CoreData

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

Application Note 22

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. The identifier “CoreData” has been added here to separate this iteration of

FMT_MTD.1 from the optional iteration of FMT_MTD.1 defined in Appendix A.4.2.1 (FMT_MTD.1/CryptoKeys).

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 Documentation

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 c PP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

6.1.4.4 FMT_MTD.1/CryptoKeys Management of TSF data

FMT_MTD.1.1/CryptoKeys

The TSF shall restrict the ability to manage the cryptographic keys to Security Administrators.

Application Note 45

FMT_MTD.1.1/CryptoKeys restricts management of cryptographic keys to Security Administrators. It should only be chosen if cryptographic keys can be managed (e.g. modified, deleted or generated/imported) by the Security Administrator. The identifier "CryptoKeys" has been added here to separate this iteration of FMT_MTD.1 from the mandatory iteration of FMT_MTD.1 defined in Chapter 6.6.2.1 (FMT_MTD.1/CoreData).

Assurance Activity

TSS

For distributed TOEs see chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Tests

The evaluator shall try to perform at least one of the related actions (modify, delete, generate/import) without prior authentication as security administrator (either by authentication as a non-administrative user, if supported, or without authentication at all). Attempts to perform related actions without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to manage cryptographic keys can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

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

6.1.4.5 FMT_MOF.1/Services

FMT_MOF.1.1/Services

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

Application Note 44

FMT_MOF.1/Services should only be chosen if the Security Administrator has the ability to start and stop services. In this case the option 'starting and stopping services' shall be chosen in the selection in FAU_GEN.1.1. The term "services" is defined as for FAU_GEN.1.1 (see related Application Notes for FAU_GEN.1.1).

Assurance Activity

TSS

For distributed TOEs see chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Test

The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) without prior authentication as security administrator (either by authenticating as a user with no administrator privileges, if possible, or without prior authentication at all). The attempt to enable/disable this service/these services should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to enable/disable this service/these services can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) with prior authentication as security administrator. The attempt to enable/disable this service/these services should be successful.

6.1.4.6 FMT_SMF.1 Specification of Management Functions [TD0179[15], TD0319[16]]

FMT_SMF.1.1

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

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- Ability to configure the authentication failure parameters for FIA_AFL.1;
- Ability to configure audit behaviour;
- 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;
- Ability to set the time which is used for time-stamps;
- Ability to re-enable administrator account

Application Note 23

[15] Bullet-point number four of FMT_SMF.1.1 was modified by TD0179.

[16] TD0319 replaced and updated FMT_SMF.1.1. Only the final modified form is shown here.

The TOE must provide functionality for both local and remote administration in general. This cPP does not mandate, though, a specific security management function to be available either through the local administration interface, the remote administration interface or both. The TSS shall detail which security management functions are available through which interface(s). The TOE must provide functionality to configure the access banner for FTA_TAB.1 and the session inactivity time(s) for FTA_SSL_EXT.1 and FTA_SSL.3. 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/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_X509_EXT.2.2 and FPT_TUD_EXT.1.2 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 behaviour" includes the relevant management functions from FMT_MOF.1/Services and FMT_MOF.1/Functions, (for all of these SFRs that are included in the ST). If the TOE offers the ability for a remote Administrator account to be disabled in line with FIA_AFL.1 them the ST author should select "Ability to re-enable an Administrator account" to allow the account to be re-enabled by a local Administrator. 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, or if the ST is describing a distributed TOE, then the ST author makes the appropriate choice or choices in the second selection, otherwise select "No other capabilities" (in the latter case the selection may alternatively be left blank in the ST).

The selection "Ability to configure thresholds for SSH rekeying" shall be included in the ST if the TOE supports configuration of the thresholds for the mechanisms used to fulfil FCS_SSHC_EXT.1.8 or FCS_SSHS_EXT.1.8 (such configuration then requires the inclusion of FMT_MOF.1/Functions in the ST). If the TOE places limits on the values accepted for the thresholds, then this is stated in the TSS.

The selection "Ability to configure lifetime for IPsec SAs" shall be included in the ST if the TOE supports secure communication via IPsec and the FCS_IPSEC_EXT.1 requirements are included in the ST. The configuration of the lifetime for IPsec SAs needs to be in line with the selection in FCS_IPSEC_EXT.1.7 (such configuration then requires the inclusion of FMT_MOF.1/Functions in the ST).

The selection "Ability to set the time which is used for time-stamps" shall be included in the ST if the TOE allows the Administrator to set the time of the device which is then used in time stamps. This option shall not be selected if the TOE does not allow manual time setting but only relies on synchronization with external time sources like NTP servers.

The selection "Ability to configure the reference identifier for the peer" shall be included in the ST if the TOE supports secure communications via the IPsec protocol and the FCS_IPSEC_EXT.1 requirements are included in the ST. For TOEs that support only IP address and FQDN identifier types, configuration of the reference identifier may be the same as configuration of the peer's name for the purposes of connection.

For distributed TOEs the interaction between TOE components will be configurable (see FCO_CPC_EXT.1). Therefore, the ST author includes the selection "Ability to configure the interaction between TOE components" for distributed TOEs. A simple example would be the change of communication protocol according to FPT_ITT.1. Another example would be changing the management of a TOE component from direct remote administration to remote administration through another TOE component. A more complex use case would be if the realization of an SFR is achieved through two or more TOE components and the responsibilities between the two or more components could be modified.

For distributed TOEs that implement a registration channel (as described in FCO_CPC_EXT.1.2), the ST author uses the selection "Ability to configure the cryptographic functionality" in this SFR, and its corresponding mapping in the TSS, to describe the configuration of any cryptographic aspects of the registration channel that can be modified by the operational environment in order to improve the channel security (cf. the description of the content of Preparative Procedures in [SD, 3.6.1.2]).

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_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1/Services, and FMT_MOF.1/Functions (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.

TSS (containing also requirements on Guidance Documentation and Tests)

The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

For distributed TOEs with the option 'ability to configure the interaction between TOE components' the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

Guidance Documentation

See section 2.4.4.1.

Tests

The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

6.1.4.7 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 24

FMT_SMR.2.3 requires that a Security Administrator be able to administer the TOE through the local console and through a remote mechanism. The ST Author must select FTP_ITC.1, FPT_ITT.1 and/or FTP_TRP.1/Admin to demonstrate how secure communication is achieved.

For distributed TOEs not every TOE component is required to implement its own user management to fulfil this SFR. At least one component has to support authentication and identification of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. For the other TOE components authentication as Security Administrator can be realized through the use of a trusted channel (either according to FTP_ITC.1 or FPT_ITT.1) from a component that supports the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. The identification of users according to FIA_UIA_EXT.1.2 and the association of users with roles according to FMT_SMR.2.2 is done through the components that support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. TOE components that authenticate Security Administrators through the use of a trusted channel are not required to support local administration of the component as defined in FMT_SMR.2.3.

Assurance Activity

Guidance Documentation

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.

Tests

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 Protection of the TSF (FPT)

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 26

The intent of the requirement is that raw password authentication data is 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 could directly read memory to capture a password but is trusted not to do so. Passwords should be obscured during entry on the local console in accordance with FIA_UAU.7.

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 pre-shared, symmetric and private keys)

FPT_SKP_EXT.1.1

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

Application Note 25

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 TSF Testing (Extended)

FPT_TST_EXT.1.1

The TSF shall run a suite of the following self-tests during initial start-up (on power on), to demonstrate the correct operation of the TSF: **digital signature verification of the TOE firmware, cryptographic known answer tests for all cryptographic algorithms**.

Application Note 27

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-tests 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.

Non-distributed TOEs may internally consist of several components that contribute to enforcing SFRs. Self-testing shall cover all components that contribute to enforcing SFRs and verification of integrity shall cover all software that contributes to enforcing SFRs on all components.

For distributed TOEs all TOE components have to perform self-tests. This does not necessarily mean that each TOE component has to carry out the same self-tests: the ST describes the applicability of the selection (i.e. when self-tests are run) and the final assignment (i.e. which self-tests are carried out) to each TOE component.

Application Note 28

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/Rev 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.

For distributed TOEs the evaluator shall examine the TSS to ensure that it details which TOE component performs which self-tests and when these self- tests are run.

Guidance Documentation

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.

For distributed TOEs the evaluator shall ensure that the guidance documentation describes how to determine from an error message returned which TOE component has failed the self-test.

Tests

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 fulfil 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. Note that the testing is not restricted to the cryptographic functions of the TOE.
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 either verify that the self-tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

For distributed TOEs the evaluator shall perform testing of self-tests on all TOE components according to the description in the TSS about which self-test are performed by which component.

6.1.5.4 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1

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

Application Note 29

If a trusted update can be installed on the TOE with a delayed activation the version of both the currently executing image and the installed but inactive image must be provided. In this case the option 'the most recently installed version of the TOE firmware/software' needs to be chosen from the selection in FPT_TUD_EXT.1.1 and the TSS needs to describe how and when the inactive version becomes active. If all trusted updates become active as part of the installation process, only the currently executing version needs to be provided. In this case the option 'no other TOE firmware/software version' shall be chosen from the selection in FPT_TUD_EXT.1.1.

For a distributed TOE, the method of determining the installed versions on each component of the TOE is described in the operational guidance.

Assurance Activity

TSS

The evaluator shall verify that the TSS describe how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the TSS needs to describe how and when the inactive version becomes active. The evaluator shall verify this description.

The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system firmware and software (for simplicity the term 'software' will be used in the following although the requirements apply to firmware and 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 options ‘support automatic checking for updates’ or ‘support automatic updates’ are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

For distributed TOEs, the evaluator shall examine the TSS to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component. Alternatively, this description can be provided in the guidance documentation. In that case the evaluator should examine the guidance documentation instead.

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.

If a published hash is used to protect the trusted update mechanism, then the evaluator shall verify that the trusted update mechanism does involve an active authorization step of the Security Administrator, and that download of the published hash value, hash comparison and update is not a fully automated process involving no active authorization by the Security Administrator. In particular, authentication as Security Administration according to FMT_MOF.1/ManualUpdate needs to be part of the update process when using published hashes.

Guidance Documentation

The evaluator shall verify that the guidance documentation describes how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the guidance documentation needs to describe how to query the loaded but inactive version.

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.

If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

For distributed TOEs the evaluator shall verify that the guidance documentation describes how the versions of individual TOE components are determined for FPT_TUD_EXT.1, how all TOE components are updated, and the error conditions that may arise from checking or applying the update (e.g. failure of signature verification, or exceeding available storage space) along with appropriate recovery actions. . The guidance documentation only has to describe the procedures relevant for the user; it does not need to give information about the internal communication that takes place when applying updates.

If this was information was not provided in the TSS: For distributed TOEs, the evaluator shall examine the Guidance Documentation to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component.

If this was information was not provided in the TSS: If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the Guidance Documentation contains a description of how the certificates are contained on the device. The evaluator also ensures that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

Tests

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. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). 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 (if digital signatures are used): The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. 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
2) An image that has not been signed
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)
4) If the TOE allows a delayed activation of updates the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs depending on the point in time when an attempted update is rejected. 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.

c) Test 3 (if published hash is verified on the TOE): If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.
    1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the user to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE
    2) The evaluator uses a legitimate update and tries to perform verification of the hash value without storing the published hash value on the TOE. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax check of the command prevents the execution of the command without providing a hash value. In that case the mechanism that prevents the execution of this check shall be tested accordingly, e.g. that the syntax check rejects the command without providing a hash value, and the rejection of the attempt is regarded as sufficient verification of the correct behaviour of the TOE in failing to verify the hash. The evaluator then attempts to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE, the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE
    3) If the TOE allows delayed activation of updates, the TOE must be able to display both the currently executing version and most recently installed version. 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.

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

The evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all methods supported (manual updates, automatic checking for updates, automatic updates).

For distributed TOEs the evaluator shall perform Test 1, Test 2 and Test 3 (if applicable) for all TOE components.

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 30

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 administrative 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.

The TSS explains what actions are involved in the TOE support when using the "support automatic checking for updates" or "support automatic updates" selections.

When published hash values (see FPT_TUD_EXT.1.3) are used to protect the trusted update mechanism, the TOE must not automatically download the update file(s) together with the hash value (either integrated in the update file(s) or separately) and automatically install the update without any active authorization by the Security Administrator, even when the calculated hash value matches the published hash value. When using published hash values to protect the trusted update mechanism, the option "support of automatic updates" must not be used (automated checking for updates is permitted, though). The TOE may automatically download the update file(s) themselves but not to the hash value. For the published hash approach, it is intended that a Security Administrator is always required to give active authorisation for installation of an update (as described in more detail under FPT_TUD_EXT.1.3) below. Due to this, the type of update mechanism is regarded as "manually initiated update", even if the update file(s) may be downloaded automatically. A fully automated approach (without Security Administrator intervention) can only be used when "digital signature mechanism" is selected in FPT_TUD_EXT.1.3 below.

FPT_TUD_EXT.1.3

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

Application Note 31

The digital signature mechanism referenced in the selection of FPT_TUD_EXT.1.3 is one of the algorithms specified in FCS_COP.1/SigGen. The published hash referenced in FPT_TUD_EXT.1.3 is generated by one of the functions specified in FCS_COP.1/Hash. The ST author should choose the mechanism implemented by the TOE; it is acceptable to implement both mechanisms.

When published hash values are used to secure the trusted update mechanism, an active authorization of the update process by the Security Administrator is always required. The secure transmission of an authentic hash value from the developer to the Security Administrator is one of the key factors to protect the trusted update mechanism when using published hashes and the guidance documentation needs to describe how this transfer has to be performed. For the verification of the trusted hash value by the Security Administrator different use cases are possible. The Security Administrator could obtain the published hash value as well as the update file(s) and perform the verification outside the TOE while the hashing of the update file(s) could be done by the TOE or by other means. Authentication as Security Administrator and initiation of the trusted update would in this case be regarded as "active authorization" of the trusted update. Alternatively, the Administrator could provide the TOE with the published hash value together with the update file(s) and the hashing and hash comparison is performed by the TOE. In case of successful hash verification, the TOE can perform the update without any additional step by the Security Administrator. Authentication as Security Administrator and sending the hash value to the TOE is regarded as "active authorization" of the trusted update (in case of successful hash verification), because the Administrator is expected to load the hash value only to the TOE when intending to perform the update. As long as the transfer of the hash value to the TOE is performed by the Security Administrator, loading of the update file(s) can be performed by the Security Administrator or can be automatically downloaded by the TOE from a repository.

If the digital signature mechanism is selected, the verification of the signature shall be performed by the TOE itself. For the published hash option, the verification can be done by the TOE itself as well as by the Security Administrator. In the latter case use of TOE functionality for the verification is not mandated, so verification could be done using non-TOE functionality of the device containing the TOE or without using the device containing the TOE.

For distributed TOEs all TOE components shall support Trusted Update. The verification of the signature or hash on the update shall either be done by each TOE component itself (signature verification) or for each TOE component (hash verification).

Updating a distributed TOE might lead to the situation where different TOE components are running different software versions. Depending on the differences between the different software versions the impact of a mixture of different software versions might be no problem at all or critical to the proper functioning of the TOE. The TSS shall detail the mechanisms that support the continuous proper functioning of the TOE during trusted update of distributed TOEs.

Application Note 32

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

Application Note 33

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

Application Note 34

"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 firmware and software elements (e.g. applications, drivers, and kernel) of the TSF need to be protected, i.e. they should either be digitally signed by the corresponding manufacturer and subsequently verified by the mechanism performing the update or a hash should be published for them which needs to be verified before the update.

6.1.5.5 FPT_STM_EXT.1 Reliable Time Stamps

FPT_STM_EXT.1.1

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

FPT_STM_EXT.1.2

The TSF shall allow the Security Administrator to set the time,

Application Note 35

Reliable time stamps are expected to be used with other TSF, e.g. for the generation of audit data to allow the Security Administrator to investigate incidents by checking the order of events and to determine the actual local time when events occurred. The decision about the required level of accuracy of that information is up to the Administrator. The TOE depends on external time and date information, either provided manually by the Security Administrator or through the use of one or more external time sources like NTP servers. The corresponding option(s) shall be chosen from the selection in FPT_STM_EXT.1.2. The use of a local real-time clock and the automatic synchronisation with an external time source (e.g. NTP server) is recommended but not mandated. If a Security Administrator is modifying the system time remotely they must use a protected communication path as specified in FPT_TRP.1/Admin. If the TOE uses an external entity to modify the system time (NTP Server, or non-NTP external entity), such connections must be performed in accordance with FTP_ITC.1. External time source entities that do not use cryptography for authentication and integrity verification are not allowed. The ST author describes in the TSS how the external time and date information is received by the TOE and how this information is maintained.

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 discontinuous 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 determined. Note, that all discontinuous time changes, Administrator actuated or changed via an automated process, must be audited. No audit is needed when time is changed via use of kernel or system facilities – such as daytime (3) – that exhibit no discontinuities in time.

For distributed TOEs it is expected that the Security Administrator ensures synchronization between the time settings of different TOE components. All TOE components shall either be in sync (e.g. through synchronisation between TOE components or through synchronisation of

different TOE components with external NTP servers) or the offset should be known to the Administrator for every pair of TOE components. This includes TOE components synchronized to different time zones. **Assurance Activity**

TSS

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

Guidance Documentation

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.

Tests

The evaluator shall perform the following tests:

a) Test 1: If the TOE supports direct setting of the time by the Security Administrator then 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 TOE Access (FTA)

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

Guidance Documentation

The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Tests

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 (Refinement)

FTA_SSL.3.1

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

Assurance Activity

Guidance Documentation

The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Tests

For each method of remote administration, 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 (Refinement)

FTA_SSL.4.1

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

Assurance Activity

Guidance Documentation

The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Tests

For each method of remote administration, 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 (Refinement)

FTA_TAB.1.1 [TD0338[17]]

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.

[17] The TSS Assurance Activity for FTA_TAB.1 was modified by TD0338

Application Note 36

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 administrative method of access (local and remote) available to the Security Administrator (e.g., serial port, SSH, HTTPS). The evaluator shall check the TSS to ensure that all administrative methods of access available to the Security Administrator are listed and that the TSS states that the TOE is displaying an advisory notice and a consent warning message for each administrative method of access. The advisory notice and the consent warning message might be different for different administrative methods of access, and might be configured during initial configuration (e.g. via configuration file).

Guidance Documentation

The evaluator shall check the guidance documentation to ensure that it describes how to configure the bnner message.

Tests

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 Trusted path/channels (FTP)

6.1.7.1 FTP_ITC.1 Inter-TSF trusted channel (Refinement)

FTP_ITC.1.1

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

FTP_ITC.1.2

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

FTP_ITC.1.3

The TSF shall initiate communication via the trusted channel for **Audit Server (syslog)**.

ST Application Note

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 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.

Application Note 37

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 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.

While there are no requirements on the party initiating the communication, the ST author lists in the assignment for FTP_TC.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.

Where public key certificates are used in support of an FTP_ITC.1 channel, FIA_X509_EXT.1/Rev is to be used (this requires checking certificate revocation), and not the iteration FIA_X509_EXT.1/ITT which is only for use in inter-component channels of a distributed TOE.

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, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST.

Guidance Documentation

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.

Tests

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.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of external secure channels to TOE components in the Security Target.

6.1.7.2 FTP_TRP.1/Admin Trusted Path (Refinement)

FTP_TRP.1.1/Admin

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/Admin

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

FTP_TRP.1.3/Admin

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

Application Note 38

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 is 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 Documentation

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

Tests

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: The evaluator shall ensure, for each communication channel, the channel data is not sent in plaintext.

c) Test 3: The evaluators shall ensure that, for each protocol 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.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.

6.2 Security Assurance Requirements

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

Table 4: 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 – conformance (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 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.

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

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[18]: 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

[18] 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.

an adequate 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[19].

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.

[19] 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 Data Generation

The TOE saves audit logs in a local audit log file store. Each audit log file is rotated at approximately 10MB in size, but due to the lag between the appending to the log and the rotation of the log, the size may grow larger than this. Each log will never grow larger than 20MB in size. On rotation of the log file, the relevant log file is deleted and a new log file is created. No log files other than the current log file are kept by the TOE.

For each administrator-driven audit event, the date/time, username, command and arguments, and success/failure status of the command are logged. Cryptographic key operations are audited by referring to an administrator-configured “name” of the key, and do not contain the actual key value.

FAU_GEN.1

FAU_GEN.2

7.1.2 Audit Storage

The TOE manages a number of log files as follows:

- Audit log: Stores CLI commands entered by the user.
- System log: Stores general system log messages.

Each log file is rotated at approximately 10MB in size but due to the lag between the appending to the log and the rotation of the log, the size may grow larger than this. Each log will never grow larger than 20MB in size. The log files cannot be modified, but may be read by an authorized security administrator. Log files are stored in volatile memory, and are not preserved between reboots - an authorized security administrator may issue a reboot command to delete all log files.

The TOE transmits audit logs to an RFC 5424 compliant syslog server over a trusted channel as described in Section 6.1.7.1.

The TOE log files store up to approximately 10 megabytes of data locally in each log file. When a log file reaches 10 megabytes in size, the TOE deletes the existing file and creates a new one. Note that due to the lag between the appending to the log and the rotation of the log, the size may grow larger than this. But each log will never grow larger than 20MB in size. While the TOE is

configured to send data to an external RFC 5424-compliant syslog server, all log messages are sent to the syslog server over the trusted channel as described in Section 6.1.7.1.

FAU_STG_EXT.1

7.2 Cryptographic Support

7.2.1 Cryptographic Key Generation

The TOE generates, stores, and destroys cryptographic keys as follows:

Table 5: Cryptographic Key Table					
Key Type	Length (bits)	Scheme	Usage	Storage	Destruction
EC Session Keys	256, 384	FIPS 186-4 App B.4	Ephemeral Session Key for SSH session establishment.	Ephemeral; stored in RAM	Overwritten with zeroes at end of session.
DH G14 Session Keys	2048	RFC 3526 Section 3	Ephemeral Session Key for SSH session establishment.	Ephemeral; stored in RAM	Overwritten with zeroes at end of session
RSA Key	2048, 3072, 4096	FIPS 186-4 Section 5.5	Signature Generation, Signature Verification for SSH public key authentication.	Restricted key partition in plaintext.	Block Erase with read-verify when any of the designated cryptographic key zeroization commands identified in AGD are executed by the administrator. Cryptographic key zeroize Cryptographic key zeroize rsa The administrator can also destroy the key by generating a new key with the same name as the old key.
				While in use, RSA keys are held in RAM	Overwritten with zeroes when the key is no longer in use (after performing a cryptographic operation) or overwritten with a new value of the key when a new key

Table 5: Cryptographic Key Table					
Key Type	Length (bits)	Scheme	Usage	Storage	Destruction
ECDSA Key	256, 384	ISO/IEC 14888-3 Section 6.4	Signature Generation. Signature Verification for SSH public key authentication and verification of trusted updates.	Restricted key partition in plaintext.	Block Erase with read-verify when any of the designated cryptographic key zeroization commands identified in AGD are executed by the administrator. Cryptographic key zeroize Cryptographic key zeroize ec
				While in use, RSA keys are held in RAM	Overwritten with zeroes when the key is no longer in use (after performing a cryptographic operation)
HMAC Key	160, 256, 512	ISO/IEC 9797-2:2011 Section 7	Keyed Hashing for SSH	While in use, keys for HMAC keyed hashing are held in RAM	Overwritten with zeroes when the key is no longer in use (after performing a cryptographic operation)
AES Session Keys	128, 256	ISO 18033-3, 10116	SSH data encryption	Ephemeral; stored in RAM	Overwritten with zeroes at end of session

The TOE generates asymmetric RSA and EC keys using the DRBG described in section 7.2.3, with RSA key sizes of 2048-bits and EC key sizes of 256 or 384 bits. The TOE generates asymmetric AES keys using the DRBG described in section 7.2.3, with AES key sizes of 128 or 256 bits. When the TSF forms DH Group14 exchanges, the implementation is SP800-56A rev 3 compliant. The TOE utilizes domain parameters “p” and “g” from RFC 3526 section 3 and q=(p-1) / 2. A previous version of this ST written prior to the release of SP800-56A revision 3 noted that the TOE’s behavior was not compliant with SP800-56A revision 2.

The TOE is both the initiator and responder to DH key establishment based on the whether the TOE is acting as a client or a server.

FCS_CKM.1, FCS_CKM.2, FCS_CKM.4

7.2.2 Cryptographic Operations

All ECDSA, and RSA operations performed by the TOE are compliant with FIPS 186-4.

Table 6: Cryptographic Operations			
Function	Certificate Number(s)	SFR	Comments
AES - Encryption / Decryption	AES 5668, implemented OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_COP.1 /DataEncryption	The TOE performs AES in the CBC mode with key sizes 128 or 256 bits.
ECDH	CVL 2053 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_CKM.1 FCS_CKM.2	ECDH is used in EC SSH session key establishment.
ECDSA	ECDSA 1534 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h DSA 1467, implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_CKM.1 FMC_CKM.2	The TOE performs ECDSA with NIST P-256 and NIST P-384 curves for SSH public key authentication. EC key pairs are generated for SSH public/private authentication.
RSA	RSA 3050 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_CKM.1	The TOE performs RSA with 2048 bit moduli for SSH public/private authentication
HMAC	HMAC 3773 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_COP.1 /KeyedHash	The TOE performs HMAC using the following: HMAC-SHA1, with a key size of 160 bits and an output digest size of 160 bits. HMAC-SHA1 uses a 512 bit block size. HMAC-SHA2-256 with a key size of 256 bits and an output digest size of 256 bits. HMAC-SHA2-256 uses a 512 bit block size. HMAC-SHA2-512 with a key size of 512 bits and an output digest size of 512 bits. HMAC-SHA2-512 uses a 1024 bit block size.
Cryptographic Hashing	CVL 2054 and SHS 4542 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_COP.1 /Hash	The TOE performs cryptographic hashing as follows: SHA1, SHA2-512 - used in SSH HMAC. SHA-256 - used in SSH HMAC, and in digital signature generation.

			Appropriate hash algorithms (including SHA2-384) are also used for digital signature generation and verification and the key-derivation functions of SSH.
DRBG	DRBG#2290 implemented via OpenSSL 1.0.1h without a FIPS object module 1.0.1h	FCS_RBG_EXT.1	

7.2.3 Random Bit Generation

The TOE produces random bits in accordance with the NIST recommendations. The TOE utilizes a combination of multiple software sources to generate entropy. The TOE deterministic random bit generator that complies with ISO 18031:2011 and utilizes the AES-based CTR_DRBG. The operation of this generator is described in complete detail in the non-public Entropy Assessment Report [ENT]. The DRBG is seeded with at least 256 bits of entropy.

FCS_RBG_EXT.1

7.2.4 SSH Client and Server Protocols

TOEs SSH implementation is conformant to RFCs 4251, 4252, 4253, 4254, 5656, and 6668. The TOE provides SSH services for remote administration and for communicating with the audit server.

The TOE supports SSH public key authentication using RSA-2048, ECDSA-SHA2-NISTP256, ECDSA-SHA2-NISTP384 keypairs. All other public key algorithms are rejected. If an administrator does not wish to configure RSA or ECDSA public key authentication, password-based authentication is also supported.

The TOE supports SSH bulk encryption algorithms of AES-128-CBC and AES-256-CBC.

The TOE SSH implementation contains a counter for received packet size. The TOE implements a counter for each SSH session. As a packet is being received is, this counter increments for each byte received. If the counter exceeds 33,292 bytes in a single packet (inclusive of padding data, header data, etc.) the packet is dropped.

The TOE supports AES-CBC-128 and AES-CBC-256 encryption algorithms to protect the content of the SSH session. No optional characteristics are specified.

The TOE supports the use of HMAC-SHA1, HMAC-SHA2-256, and HMAC-SHA2-512 as the only integrity algorithms for SSH sessions.

The TOE allows only the following key exchange methods: DH-Group14 with SHA1, ECDH over NIST p256 with SHA2, and ECDH over NIST p384 with SHA2.

The TOE implements a counter to keep track of the number of packets transmitted or received through an SSH session. The TOE ensures that the same session keys are used for a threshold of no longer than one hour and no more than 950 MB of data has been exchanged. These thresholds are not configurable.

When acting as an SSH client, the TOE authenticates the identity of the remote SSH server using the corresponding public key. The TOE supports the use of SSH-RSA, ECDSA-SHA2-nistp256, and ECDSA-SHA2-nistp384 public keys for server authentication.

FCS_SSHC_EXT.1

FCS_SSHS_EXT.1

7.3 Identification and Authentication

7.3.1 Authentication Failure Management

FIA_AFL.1.1

The TOE allows an administrator to configure the number of successive failed authentication attempts that are permissible before account locking occurs. Each time a user attempts authentication, a counter is implemented which tracks the number of successive failed authentication attempts. When a user fails to authenticate a number of times equal to the configured limit, the TOE locks the claimed user identity (if it exists). The account remains locked until such time as a local administrator manually unlocks the account.

While the account is locked, the TOE will refuse all authentication attempts claiming the locked account as the user identity.

FIA_AFL.1

7.3.2 Password Management

The TOE allows administrators to set passwords of between 15 and 128 characters. Minimum password length can be configured by the administrator, and the administrator can configure the TOE to require passwords of at least 15 characters.

Configured passwords can use special characters - these characters include the following: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”.

FIA_PMG_EXT.1

7.3.3 User Identification and Authentication

The TOE supports local authentication via the local serial console. Authentication is performed by providing the username and password. Successful authentication will be indicated by presenting the console prompt; unsuccessful authentication is indicated by an error message and a prompt to re-authenticate.

The TOE supports remote authentication via SSH. Password based authentication is performed by providing the username and password. Successful authentication will be indicated by presenting the console prompt; unsuccessful authentication is indicated by an error message and a prompt to re-authenticate. Public-Key based authentication is performed by configuring an SSH client to send the correct public key to the TOE for comparison with the stored authentication data. Successful authentication will be indicated by presenting the console prompt; unsuccessful authentication is indicated by an error message and a prompt to provide a password for the claimed user identity. If the password authentication fails, the user will receive an error message and the SSH session will fail.

FIA_UIA_EXT.1

7.3.4 Password-based Authentication Mechanism

The TOE provides a local authentication mechanism that is password protected for administrative functions. This is the local serial console, which requires authentication as an authorized security administrator before administration of the TOE takes place.

FIA_UAU_EXT.2

7.3.5 Protected Authentication Feedback

The TOE does not echo back authentication data during authentication attempts. Password prompts provide neither the authentication data nor obfuscated data (such as asterisks).

FIA_UAU.7

7.4 Security Management

7.4.1 Management of Security Functions Behaviour

The TOE allows authorized and authenticated security administrators to perform the following functions, which are restricted only to security administrators. Non-security administrators are prevented from performing any of the following functions.

FMT_MOF.1/Functions, FMT_MOF.1/ManualUpdate,

7.4.2 Management of TSF Data

The TOE prevents non-security administrators from modifying any TSF element or security function. The only interfaces available to an unauthenticated user are the login prompts to the TOE. Only authorized security administrators may authenticate to the TOE and interact with the TSF data.

FMT_MTD.1/CoreData

FMT_MTD.1/CryptoKeys

7.4.3 Specification of Management Functions

The TOE allows the authorized security administrator to:

Administrator-configurable functions:

- Initiate manual updates of the firmware image.
- Change the behavior of the TSF with regard to transmission of audit data to external syslog servers.
- Configure the TOE clock used to maintain the accuracy of the real-time hardware clock.
- Management of user accounts, including identification and authentication parameters and credentials.
- Manage the TSF data.
- Create, delete, and configure the use of cryptographic keys and parameters.
- Configure the access banner
- Configure session inactivity timers for local and remote administrative sessions.
- Configure the availability of services and actions available prior to initiating the identification and authentication process, as defined in FIA_UIA_EXT.1.1
- Enable, disable, start or stop any of the following services:
    - SSH server
- Perform administrative activities locally and remotely
- Configure the pre-login access banner in accordance with FTA_TAB
- Configure an inactivity timeout period
- Perform updates to the TOE firmware
- Configure the operation of the cryptographic functions.
- Enable, Disable, View, and Edit the behavior of the security functions
- Configure the security management functions of the TOE

FMT_SMF.1, FMT_MOF.1/Services

7.4.4 Restrictions on Security Roles

The TSF allows one security administrator account to be created on the TOE. All administrator accounts are security administrators.

FMT_SMR.2

7.5 Protection of the TSF

7.5.1 Protection of Administrator Passwords

Local and SSH password authentication data is stored as an MD5 hash in the underlying operating system and running configuration. Public keys used for SSH authentication to the TOE are stored in the underlying operating system and in the running configuration in plaintext form. Non-security administrators are not permitted to view the public keys.

FPT_APW_EXT.1

7.5.2 TSF Testing

The TOE performs an integrity check of the installed firmware by comparing the 4096-bit RSA digital signature of the complete firmware image during the bootup process, before any configuration is loaded or any interfaces are enabled. If signature verification fails, the TOE fails the self-testing process. After verifying the image, the TOE performs cryptographic known-answer tests on all cryptographic algorithms. The correct operation of the TOE is ensured by verifying that the installed firmware image is unmodified and that the cryptographic modules are performing as expected.

During bootup, the TOE also performs cryptographic health testing. All cryptographic algorithms are tested via known-answer testing. If any cryptographic self-testing fails, the TOE will immediately reboot and log an audit message at the local console.

The TOE also performs entropy health testing at startup, and continuously during normal operation. If the entropy noise source health testing fails, the TOE will immediately reboot and log an audit message at the local console.

FPT_TST_EXT.1

7.5.3 Trusted Update

The TOE allows security administrators to initiate manual updates by copying the update candidate to the TOE using non-TOE provided methods (SCP or TFTP). The candidate updates are first obtained from the vendor support channel. The update candidate is stored in local flash memory, and the administrator is instructed to verify the digital signature of the update candidate manually to verify that the digital signature matches the expected result. If signature verification fails, the administrator is notified that the firmware update cannot continue, and the update candidate is automatically deleted by the TOE. If signature verification succeeds, the administrator is instructed that the update may continue, and the TOE proceeds to apply the update.

The administrator may run the “show version” command to query the currently executing version of software. Pending updates (software updates which have been verified as described above) become active when the TOE reboots.

FPT_TUD_EXT.1

7.5.4 Protection of TSF Data

The TOE stores private portions of keypairs in individual micropartitions of the local flash storage. There is no standard interface for reading the stored private keys. Private keys may be destroyed (zeroized) or replaced (zeroized and overwritten with new data), but not read.

FPT_SKP_EXT.1

7.5.5 Reliable Time Stamps

The TOE updates the local real-time hardware. The time is used for the auditing and timestamping process. The time interval (counter) is used for session timers and audit logging.

The hardware real-time clock has a drift estimate of less than 1.5 minutes per month, which does not represent a security concern in the context of any of the stated security functions.

FPT_STM_EXT.1

7.6 TOE Access

7.6.1 TSF-initiated Session Locking

The TOE terminates local administrative sessions when the session inactivity timer expires.

FTA_SSL_EXT.1

7.6.2 TSF-initiated Termination

The TOE terminates remote administrative sessions when the session inactivity timer expires.

FTA_SSL.3

7.6.3 User-initiated Termination

The TOE allows the administrator to terminate their own session for both remote and local sessions.

FTA_SSL.4

7.6.4 Default TOE Access Banners

The TOE provides a pre-authentication access banner that is configurable by the administrator and is displayed in local and remote authentication sessions prior to allowing the user to authenticate. Other permitted unauthenticated actions are: respond to ICMP requests, send and receive DNS requests, send and receive ARP requests, send and receive Spanning Tree Protocol packets, send and receive NTP packets. The TOE permits SSH packets to be sent and received as part of the initial phase of identification and authentication.

FTA_TAB.1

7.7 Trusted Path/Channels

7.7.1 Inter-TSF Trusted Channel

The TOE utilizes SSH to provide a trusted communication channel between itself and authorized IT entities that is logically distinct from other communication channels and provides assured identification of the endpoints. The Trusted Channel prevents detection or disclosure of the channel data.

The TOE uses the trusted channel to protect communication between itself and the audit / syslog server.

The TOE initiates communication via the trusted channel for syslog. The TOE is the client when communicating to the remote audit server. Assured identification of the remote endpoint is provided by verifying the remote SSH server's public key.

FTP_ITC_EXT.1

7.7.2 Trusted Path

The TOE provides SSH as the remote administration mechanisms between itself and authorized administrators to provide assured identification of the endpoints and protection of the channel data against detection or disclosure.

FTP_TRP.1

8. Terms and Definitions

Table 7: TOE Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
AES	Advanced Encryption Standard
ASCII	American Standard Code for Information Interchange
BIOS	Basic Input/Output System
CA	Certificate Authority
CBC	Cipher Block Chaining
CLI	Command Line Interface
CMOS	Complementary Metal–Oxide–Semiconductor
CRL	Certificate Revocation List
CSP	Critical Security Parameter
CSR	Certificate Signing Request
CTR	Counter
DH	Diffie-Hellman
DHE	Diffie-Hellman Ephemeral
DNS	Domain Name System
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
ECDH	Elliptic Curve Diffie-Hellman
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
ESP	Encapsulating Security Payload
GCM	Galois Counter Mode
GUI	Graphical User Interface
HMAC	Hash Message Authentication Code
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
ICMP	Internet Control Message Protocol
IKE	Internet Key Exchange
IP	Internet Protocol
KAT	Known Answer Test
KDF	Key Derivation Function
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
PCT	Pairwise Consistency Test
PKCS	Public Key Cryptography Standards
RFC	Requests for Comments
RSA	Rivest-Shamir-Adleman
SA	Security Association
SAN	Subject Alternative Name
SFP	Small Form-Factor Pluggable
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol
SPD	Security Policy Database
SSH	Secure Shell
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UI	User Interface
URI	Uniform Resource Identifier
VIK	Voyager® Ignition Key

Table 8: CC Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
CC	Common Criteria
CCRA	Arrangement on the Recognition of Common Criteria Certificates in the field of IT Security
DOD	Department of Defense
EAL	Evaluation Assurance Level
FIPS	Federal Information Processing Standards
OSP	Organizational Security Policy
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SFP	Security Function Policy
ST	Security Target
TOE	Target of Evaluation
TSF	TOE Security Functionality
TSFI	TSF Interface
TSS	TOE Summary Specification

9. References

Table 9: TOE Guidance Documentation		
Reference	Description	Date
[T1]	Operational User Guidance Version 1.0	July 2018

Table 10: Common Criteria v3.1 References			
Reference	Description	Version	Date
[C1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model CCMB-2009-07-001	V3.1 R4	July 2009
[C2]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components CCMB-2009-07-002	V3.1 R4	July 2009
[C3]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components CCMB-2009-07-003	V3.1 R4	July 2009
[C4]	Common Criteria for Information Technology Security Evaluation Evaluation Methodology CCMB-2009-07-004	V3.1 R4	July 2009

Table 11: Supporting Documentation			
Reference	Description	Version	Date
[cPP]	Collaborative Protection Profile for Network Devices + Errata 20180314	2.0E	March 14, 2018
[SD]	Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP	2.0	May 5, 2017

Annex A Algorithm Validation Requirements

FCS_CKM.1.1 [TD0291[20]]

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

[20] Algorithm testing for FCS_CKM.1.1 was modified by TD0291.

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.

Testing for FFC Schemes using Diffie-Hellman group 14 is done as part of testing in CKM.2.1.

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.

Diffie-Hellman Group 14

The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses Diffie-Hellman group 14.

FCS_COP.1/DataEncryption

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/SigGen

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 generates or obtains 10 messages for each modulus size/SHA combination supported by the TOE. The TOE generates and returns the corresponding signatures.

The evaluator shall verify the correctness of the TOE's signature using a trusted reference implementation of the signature verification algorithm and the associated public keys to verify the signatures.

Signature Verification Test

For each modulus size/hash algorithm selected, the evaluator generates a modulus and three associated key pairs, (d, e). Each private key d is used to sign six pseudorandom messages each of 1024 bits using a trusted reference implementation of the signature generation algorithm. Some of the public keys, e, messages, or signatures are altered so that signature verification should fail. For both the set of original messages and the set of altered messages: the modulus, hash algorithm, public key e values, messages, and signatures are forwarded to the TOE, which then attempts to verify the signatures and returns the verification results.

The evaluator verifies that the TOE confirms correct signatures on the original messages and detects the errors introduced in the altered messages.

FCS_COP.1/Hash

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.