

Copyright Notice

Copyright © 2020 Teron Labs Pty Ltd.

This document contains information protected by copyright. TERON LABS PTY LTD, registered in Australia under Australian Business Number 38 627 752 836.

Teron Labs AISEF

Unit 3, 10 Geils Court
Deakin, ACT 260
Australia

+61 2 6172 1261
info@teronlabs.com
www.teronlabs.com

Table of Contents

1	References.....	6
1.1	Evaluation Requirements.....	6
1.2	Evaluation Evidence	6
2	Introduction	7
2.1	Evaluation Identifiers.....	7
2.2	ST Identifier	8
2.3	TOE Overview	8
2.4	Test Equivalency	10
3	CAVP Certificates	11
4	Security Functional Requirements Assurance Activities	14
4.1	Technical Decisions.....	14
4.2	Security Audit (FAU)	17
4.2.1	FAU_GEN.1 Audit data generation.....	17
4.2.2	FAU_GEN.2 User identity association	18
4.2.3	FAU_STG_EXT.1 Protected audit event storage.....	19
4.3	Cryptographic Support (FCS)	22
4.3.1	FCS_CKM.1 Cryptographic Key Generation.....	22
4.3.2	FCS_CKM.2 Cryptographic Key Establishment.....	24
4.3.3	FCS_CKM.4 Cryptographic Key Destruction	27
4.3.4	FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption).....	28
4.3.5	FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)	33
4.3.6	FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm).....	34
4.3.7	FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm).....	35
4.3.8	FCS_RBG_EXT.1 Extended: Cryptographic Operation (Random Bit Generation) .	36
4.4	Identification and Authentication (FIA)	37
4.4.1	FIA_AFL.1 Authentication Failure Management	37
4.4.2	FIA_PMG_EXT.1 Password Management.....	39
4.4.3	FIA_UIA_EXT.1 User Identification and Authentication	40
4.4.4	FIA_UAU_EXT.2 Password-based Authentication Mechanism	42

4.4.5	FIA_UAU.7 Protected Authentication Feedback	42
4.5	Security management (FMT)	43
4.5.1	FMT_MOF.1/ManualUpdate	43
4.5.2	FMT_MTD.1/CoreData Management of TSF Data	43
4.5.3	FMT_SMF.1 Specification of Management Functions	44
4.5.4	FMT_SMR.2 Restrictions on security roles.....	45
4.5.5	Protection of the TSF (FPT).....	45
4.5.6	FPT_SKP_EXT.1 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys).....	45
4.5.7	FPT_APW_EXT.1 Protection of Administrator Passwords	46
4.5.8	FPT_TST_EXT.1 TSF testing	46
4.5.9	FPT_TUD_EXT.1 Trusted Update	47
4.5.10	FPT_STM_EXT.1 Reliable Time Stamps	51
4.6	TOE Access (FTA).....	52
4.6.1	FTA_SSL_EXT.1 TSF-initiated Session Locking	52
4.6.2	FTA_SSL.3 TSF-initiated Termination	52
4.6.3	FTA_SSL.4 User-initiated Termination	53
4.6.4	FTA_TAB.1 Default TOE Access Banners.....	54
4.7	Trusted path/channels (FTP)	55
4.7.1	FTP_ITC.1 Inter-TSF trusted channel.....	55
4.7.2	FTP_TRP.1/Admin Trusted Path	56
5	Evaluation Activities for Optional Requirements	58
5.1	Security Audit (FAU)	58
5.1.1	FAU_STG.1 Protected audit trail storage	58
6	Evaluation Activities for Selection-Based Requirements	60
6.1	Cryptographic Support (FCS)	60
6.1.1	FCS_SSHS_EXT.1 SSH Server	60
6.2	Security management (FMT)	67
6.2.1	FMT_MOF.1/Functions Management of security functions behaviour	67
6.2.2	FMT_MOF.1/Services.....	68
6.2.3	FMT_MTD.1/CryptoKeys Management of TSF Data	69
7	Evaluation Activities for SARs.....	70
7.1	ADV: Development	70

7.1.1	Basic Functional Specification (ADV_FSP.1).....	70
7.2	AGD: Guidance Documents.....	71
7.2.1	Operational User Guidance (AGD_OPE.1).....	71
7.2.2	Preparative Procedures (AGD_PRE.1).....	73
7.3	ALC: Life-cycle Support	74
7.3.1	Labelling of the TOE (ALC_CMC.1).....	74
7.3.2	TOE CM coverage (ALC_CMS.1).....	74
7.4	ATE: Tests	75
7.4.1	Independent Testing – Conformance (ATE_IND.1)	75
7.5	AVA: Vulnerability Assessment.....	78
7.5.1	Vulnerability Survey (AVA_VAN.1)	78
8	Glossary.....	80

1 References

1.1 Evaluation Requirements

- [1] Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, Version 3.1, Revision 5
- [2] Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, Version 3.1, Revision 5
- [3] Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, version 3.1 Revision 5
- [4] Common Methodology for Information Technology Security Evaluation, Evaluation methodology, Version 3.1, Revision 5
- [5] Collaborative Protection Profile for Network Devices (NDcPP), Version 2.1, 24 September 2018
- [6] Supporting Document, Evaluation Activities for Network Device cPP, Version 2.1, September-2018

1.2 Evaluation Evidence

- [7] Security Target Junos OS 19.3R1 for QFX5120, QFX5210 and EX4650, Version 1.1, 14-Jul-2020
- [8] Test Report NDcPP - Junos 19.3R1 for QFX5120, QFX5210 and EX4650 (EFT-T007-TR-NDcPP), Version d7, 30-Apr-2020
- [9] Junos OS Common Criteria Evaluated Configuration Guide for EX4650-48Y, QFX5120-48Y, QFX5120-32C and QFX5210-64C Devices, Release 19.3R1 16-Mar-2020
- [10] Junos OS CLI User Guide, 25 September 2019
- [11] Junos OS Installation and Upgrade Guide, 09 January 2020
- [12] Junos OS User Access and Authentication User Guide, 30 December 2019
- [13] Evaluation Workbook: AVA (EFT-T007-EWB-AVA), Version 1.0, 30-Apr-2020
- [14] Evaluation Workbook: ASE (EFT-T007-EWB-ASE), Version 1.0, 30-Apr-2020
- [15] Evaluation Workbook: AGD (EFT-T007-EWB-AGD), Version 1.0, 30-Apr-2020
- [16] Evaluation Workbook: ATE (EFT-T007-EWB-ATE), Version 1.0, 30-Apr-2020
- [17] Seeding of the Kernel RBG in EX4650, QFX5120 and QFX5210 Platforms Running Junos 19.3R1, Version 1.3, 25-Mar-2020

2 Introduction

This report documents the assurance activities performed by Teron Labs as part of the Common Criteria evaluation of Junos OS 19.3R1 for QFX5120, QFX5210 and EX4650 developed by Juniper Networks. The product was evaluated against the requirements of the Collaborative Protection Profile for Network Devices Version 2.1 (NDcPP) and comprised of the assurance activities defined in the Supporting Document, Evaluation Activities for Network Device cPP, Version 2.1 (SD), revised as per the applicable Technical Decisions (TDs) listed in Section 4.1. Based on the results of these activities, Teron Labs determined that Junos OS 19.3R1 for QFX5120, QFX5210 and EX4650 and supporting evidence documentation passes all requirements of the NDcPP.

2.1 Evaluation Identifiers

Task Identifier	EFT-T007
TOE Name	Junos OS 19.3R1 for QFX5120, QFX5210 and EX4650
TOE Version	19.3R1
Sponsor	Juniper Networks, Inc. 1133 Innovation Way, Sunnyvale California 94089 United States
Developer	Juniper Networks, Inc. 1133 Innovation Way, Sunnyvale California 94089 United States
Evaluation Facility	Teron Labs Unit 3, 10 Geils Court, Deakin, ACT 2600, Australia
Scheme	Australasian Information Security Evaluation Program (AISEP)
PP(s)	Collaborative Protection Profile for Network Devices (NDcPP), Version 2.1, 24-September-2018.
CC Version	3.1 Revision 5

2.2 ST Identifier

ST Title	Security Target Junos OS 19.3R1 for QFX5120, QFX5210 and EX4650
ST Version	1.1
ST Date	14-Jul-2020

2.3 TOE Overview

The Target of Evaluation (TOE) is Juniper Networks, Inc. Junos OS 19.3R1 executing on QFX5120, QFX5210 and EX4650 Ethernet Switches.

Each of the Ethernet Switches is a secure network device that protects itself largely by offering only a minimal logical interface to the network and attached nodes. All switching platforms are powered by the Junos OS software, Junos OS 19.3R1, which is a special purpose OS that provides no general-purpose computing capability. Junos OS provides both management and control functions as well as all IP switching.

The Ethernet Switches primarily support the definition of, and enforce, information flow policies among network nodes. All information flow from one network node to another passes through an instance of the TOE. Information flow is controlled on the basis of network node addresses and protocol. In support of the information flow security functions, the TOE ensures that security-relevant activity is audited and provides the security tools to manage all of the security functions.

Each instance of the TOE consists of the following two major architectural components:

- The Routing Engine (RE) runs the Junos firmware and provides Layer 2 and Layer 3 switching services and network management for all operations necessary for the configuration and operation of the TOE and controls the flow of information through the TOE.
- The Packet Forwarding Engine (PFE) provides all operations necessary for packet forwarding.

The functions of the appliances can all be managed through the Junos firmware, either from a connected terminal console or via a network connection. Network management is secured using the SSH protocol. All management, whether from a user connecting to a terminal or from the network, requires successful authentication. In the evaluated deployment the TOE is managed and configured via Command Line Interface, either via a directly connected console or over the network secured using the SSH protocol

The TOE is contained within the physical boundary of the appliance chassis listed in Table 1. The physical boundary of the QFX5120, QFX5210 and EX4650 appliance instances of the TOE includes the KVM Hypervisor, which provides the virtualization layer in which Junos OS VM executes, as shown in Figure 1.

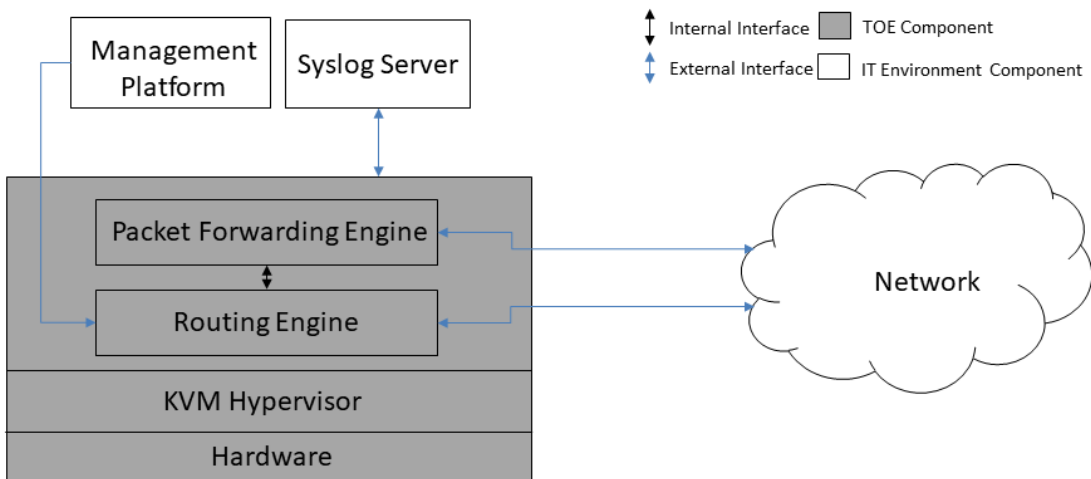


Figure 1 QFX5120, QFX5210 and EX4650 TOE Boundary

The TOE interfaces are comprised of the following:

- Network interfaces which pass traffic
- Management interface to handle administrative actions.

- Table 1 TOE Chassis Details

Switch Model	Network ports	Firmware
QFX5120	<ul style="list-style-type: none"> • QFX5120-48Y: <ul style="list-style-type: none"> ○ A 25GbE/100GbE data center access switch offering 48 SFP+ transceiver ports and eight QSFP28 ports that can be configured as 8x40GbE or 8x100GbE ports, with an aggregate throughput of 2 Tbps or 1.31 Bpps per switch. ○ Each QSFP28 port can be configured as 4x25GbE ports using breakout cables while each QSFP+ port can be configured as 4x10GbE ports using breakout cables, increasing the number of 25GbE and 10GbE ports per switch to 80. • QFX5120-32C: <ul style="list-style-type: none"> ○ A compact 100GbE data center spine switch offering 32 QSFP28 or 32 QSFP+ ports. ○ 100GbE ports can be configured as 4x25GbE ports using breakout cables. 	Junos OS 19.3R1

Switch Model	Network ports	Firmware
EX4650	<ul style="list-style-type: none"> • EX4650-48Y: <ul style="list-style-type: none"> ○ A 25GbE/100GbE campus distribution switch offering 48 SFP28 transceiver ports and eight QSFP28 ports that can be configured as 8x40GbE or 8x100GbE ports, with an aggregate throughput of 2 Tbps or 1.49 Bpps per switch. ○ Each QSFP28 port can be configured as 4x25GbE ports using breakout cables, increasing the total number of 25GbE ports to 80 per switch. 	
QFX5210	<ul style="list-style-type: none"> • QFX5210-64C: <ul style="list-style-type: none"> ○ A compact 100GbE data center spine switch offering 64 QSFP28 or 64 QSFP+ ports. ○ 100GbE ports can be channelized to support 2x50GbE, 4x25GbE, or 4x10GbE downlinks 	

2.4 Test Equivalency

The TOE is Junos OS 19.3R1 running on four possible hardware platforms:

- QFX5120-48Y
- QFX5120-32C
- QFX5210-64C
- EX4650-48Y

All four platforms share the same Intel Xeon D1518 processor and switching fabric. The only differences between the 4 platforms is the number and type of network ports as described in Table 2 of the Security Target. QFX5120-48Y and EX4650-48Y are identical hardware. Furthermore, all four platforms run the same binary firmware image. Based on this, the evaluators deemed the four platforms to be equivalent for the purpose of NDcPP testing and chose a single EX4650-48Y platform to conduct all testing.

3 CAVP Certificates

Some of the tests of cryptographic functionality of the TOE were carried out via verification of CAVP certification claims. For these tests, the evaluators checked that the CAVP certificate numbers provided in the ST matched the TOE and cryptographic testing requirements specified in NDcPP-SD (Ref. [6]). To map NDcPP testing requirements against equivalent CAVP claims, the evaluators referred to Addendum #2 of NIAP Policy Letter #51 of 6 December 2019.

Table 2 lists the cryptographic algorithms used by the TOE along with the supported SFRs and CAVP certificate numbers.

Table 2 - CAVP certification claims

Library	Function, Usage, Algorithm, Mode, Keysize	NIST standard	SFRs Supported	Certificate Number
OpenSSL (OpenSSH)	SSH AES Data Encryption/Decryption <ul style="list-style-type: none"> - AES-CBC with key sizes 128 bit, 192 bit and 256 bit - AES-CTR with key sizes 128 bit, 192 bit and 256 bit 	FIPS 197, SP800-38A	FCS_COP.1/DataEncryption	<u>C1543</u>
	SSH Hashing: <ul style="list-style-type: none"> - SHA1 - SHA2-256 - SHA2-384 - SHA2-512 	FIPS 180-4	FCS_COP.1/Hash	<u>C1543</u>
	SSH Keyed Hashing: <ul style="list-style-type: none"> - HMAC-SHA1 - HMAC-SHA2-256 - HMAC-SHA2-512 	FIPS 198-1	FCS_COP.1/KeyedHash	<u>C1543</u>

¹ https://www.niap-ccvcs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

Library	Function, Usage, Algorithm, Mode, Keysize	NIST standard	SFRs Supported	Certificate Number
	SSH Signature Generation and Verification using ECDSA: <ul style="list-style-type: none"> - ECDSA (P-256 w/ SHA-256) - ECDSA (P-384 w/ SHA-384) - ECDSA (P-521 w/ SHA-512) 	FIPS 186-4	FCS_COP.1/SigGen	<u>C1543</u>
	SSH Key Generation and Verification for ECDH: <ul style="list-style-type: none"> - ECDSA (P-256) - ECDSA (P-384) - ECDSA (P-521) 	FIPS 186-4	FCS_CKM.1	<u>C1543</u>
	SSH Signature Generation and Verification using RSA: <ul style="list-style-type: none"> - RSA PKCS1_V1_5 (n=2048 (SHA 256), n=3072 (SHA 256)) 	FIPS 186-4	FCS_COP.1/SigGen	<u>C1543</u>
	RSA Key Generation: <ul style="list-style-type: none"> - RSA PKCS1_V1_5 (n=2048, n=3072) 	FIPS 186-4	FCS_CKM.1	<u>C1543</u>
	SSH Keys Establishment using ECC: <ul style="list-style-type: none"> - KAS-ECC Component (P-256 w/SHA-256, P-384 w/SHA-384, P-521 w/SHA-512) 	SP800-56A	FCS_CKM.2	<u>C1543</u>
	SSH Key Derivation: <ul style="list-style-type: none"> - CVL SSH v2 KDF (SHA-1, SHA-384, SHA-512) 	SP800-135	FCS_CKM.2	<u>C1543</u>
OpenSSL	Random bit generation with HMAC-DRBG <ul style="list-style-type: none"> - DRBG (HMAC-SHA-2-256) 	SP800-90A	FCS_RBG_EXT.1	<u>C1543</u>

Library	Function, Usage, Algorithm, Mode, KeySize	NIST standard	SFRs Supported	Certificate Number
LibMD	Hashing for password conditioning, password hashing, and self-testing (verifying integrity of system files): <ul style="list-style-type: none"> - SHA1, SHA2-256, SHA2-512 	FIPS 180-4	FCS_COP.1/Hash	<u>C1542</u>
	Keyed hashing for password conditioning, password hashing, and self-testing (verifying integrity of system files): <ul style="list-style-type: none"> - HMAC-SHA1, HMAC-SHA2-256 	FIPS 198-1	FCS_COP.1/KeyedHash	<u>C1542</u>
Kernel	Hashing for Random bit generation with HMAC-DRBG: <ul style="list-style-type: none"> - SHA1, SHA2-256, SHA2-384, SHA2-512 	FIPS 180-4	FCS_COP.1/Hash	<u>C1541</u>
	Keyed hashing for Random bit generation with HMAC-DRBG: <ul style="list-style-type: none"> - HMAC-SHA1, HMAC-SHA2-256 	FIPS 198-1	FCS_COP.1/KeyedHash	<u>C1541</u>
	Random bit generation with HMAC-DRBG: <ul style="list-style-type: none"> - DRBG (HMAC-SHA-2-256) 	SP800-90A	FCS_RBG_EXT.1	<u>C1541</u>

4 Security Functional Requirements Assurance Activities

This section describes the evaluation activities defined in the NDcPP-SD (Ref. [6]) regarding TOE summary specification (TSS), guidance and functional testing requirements. Descriptions of the requirements are formatted within a coloured box. The descriptions have been taken from the SD with revisions specified by the relevant technical decisions listed in the ST and reproduced in Table 1 below. Each requirement is followed by the corresponding evaluation findings. Note that only evaluation activities applicable to the TOE are included. In particular, as the TOE is not a distributed system, evaluation activities defined in the SD for distributed systems are omitted.

4.1 Technical Decisions

The following technical decisions (TDs) are applicable to the evaluated TOE:

ITEM	TITLE	REFERENCE	PUBLICATI ON DATE	Relevant to ST
TD0484	NIT Technical Decision for Interactive sessions in FTA_SSL_EXT.1 & FTA_SSL.3	FTA_SSL_EXT.1, FTA_SSL.3	2019.12.18	Yes
TD0483	NIT Technical Decision for Applicability of FPT_APW_EXT.1	FPT_APW_EXT.1	2019.12.18	Yes
TD0482	NIT Technical Decision for Identification of usage of cryptographic schemes	FCS_CKM.1.1, FCS_CKM.2.1	2019.12.18	Yes
TD0481	NIT Technical Decision for FCS_(D)TLSC_EXT.X.2 IP addresses in reference identifiers	FAU_GEN.1, FCS_(D)TLSC_EXT.X.2	2019.12.18	No
TD0480	NIT Technical Decision for Granularity of audit events	FIA_AFL.1	2019.12.18	Yes
TD0478	NIT Technical Decision for Application Notes for FIA_X509_EXT.1 iterations	FIA_X509_EXT.1/Rev, FIA_X509_EXT.1/ITT	2019.12.18	No
TD0477	NIT Technical Decision for Clarifying FPT_TUD_EXT.1 Trusted Update	FPT_TUD_EXT.1	2019.12.18	Yes

ITEM	TITLE	REFERENCE	PUBLICATI ON DATE	Relevant to ST
TD0475	NIT Technical Decision for Separate traffic consideration for SSH rekey	FCS_SSHC_EXT.1.1, FCS_SSHS_EXT.1.1	2019.12.18	Yes
TD0453	NIT Technical Decision for Clarify authentication methods SSH clients can use to authenticate SSH servers	FCS_SSHC_EXT.1.9	2019.09.16	No
TD0451	NIT Technical Decision for ITT Comm UUID Reference Identifier	FCS_TLSS_EXT.1.2, FCS_TLSS_EXT.2.2	2019.09.16	No
TD0450	NIT Technical Decision for RSA-based ciphers and the Server Key Exchange message	FCS_TLSS_EXT.*.3, FCS_DTLSS_EXT.*.4	2019.09.16	No
TD0447	NIT Technical Decision for Using 'diffie-hellman-group-exchange-sha256' in FCS_SSHC/S_EXT.1.7	FCS_SSHC_EXT.1.7, FCS_SSHS_EXT.1.7	2019.09.16	Yes
TD0425	NIT Technical Decision for Cut-and-paste Error for Guidance AA	FTA_SSL.3	2019.05.31	Yes
TD0424	NIT Technical Decision for NDcPP v2.1 Clarification - FCS_SSHC/S_EXT1.5	FCS_SSHC_EXT.1.5, FCS_SSHS_EXT.1.5	2019.05.31	Yes
TD0423	NIT Technical Decision for Clarification about application of Rfl#201726rev2	N/A	2019.05.31	No
TD0412	NIT Technical Decision for FCS_SSHS_EXT.1.5 SFR and AA discrepancy	FCS_SSHS_EXT.1.5	2019.03.22	Yes
TD0411	NIT Technical Decision for FCS_SSHC_EXT.1.5, Test 1 - Server and client side seem to be confused	FCS_SSHC_EXT.1.5	2019.03.22	No
TD0410	NIT technical decision for Redundant assurance activities associated with FAU_GEN.1	FIA_AFL.1	2019.03.22	Yes

ITEM	TITLE	REFERENCE	PUBLICATI ON DATE	Relevant to ST
TD0409	NIT decision for Applicability of FIA_AFL.1 to key-based SSH authentication	FIA_AFL.1	2019.03.22	Yes
TD0408	NIT Technical Decision for local vs. remote administrator accounts	FIA_AFL.1, FIA_UAU_EXT.2, FMT_SMF.1	2019.03.22	Yes
TD0407	NIT Technical Decision for handling Certification of Cloud Deployments	N/A	2019.03.22	Yes
TD0402	NIT Technical Decision for RSA-based FCS_CKM.2 Selection	FCS_CKM.2	2019.02.24	Yes
TD0401	NIT Technical Decision for Reliance on external servers to meet SFRs	FTP_ITC.1	2019.02.24	Yes
TD0400	NIT Technical Decision for FCS_CKM.2 and elliptic curve-based key establishment	FCS_CKM.1, FCS_CKM.2	2019.02.24	Yes
TD0399	NIT Technical Decision for Manual installation of CRL (FIA_X509_EXT.2)	FIA_X509_EXT.2	2019.02.24	No
TD0398	NIT Technical Decision for FCS_SSH*EXT.1.1 RFCs for AES-CTR	FCS_SSHC_EXT.1.1, FCS_SSHS_EXT.1.1	2019.02.24	Yes
TD0397	NIT Technical Decision for Fixing AES-CTR Mode Tests	FCS_COP.1/DataEncrypti on	2019.02.24	Yes
TD0396	NIT Technical Decision for FCS_TLSC_EXT.1.1, Test 2	FCS_DTLSC_EXT.1.1, FCS_DTLSC_EXT.2.1, FCS_TLSC_EXT.1.1, FCS_TLSC_EXT.2.1	2019.02.24	No
TD0395	NIT Technical Decision for Different Handling of TLS1.1 and TLS1.2	FCS_TLSS_EXT.2.4, FCS_TLSS_EXT.2.5	2019.02.24	No

Table 3 Applicable NIAP Technical Decisions

4.2 Security Audit (FAU)

4.2.1 FAU_GEN.1 Audit data generation

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.

Section 7.4 of the ST specifies the information that is logged by the TOE in order to identify the key being operated on for SSH session keys and SSH authentication keys.

Guidance Documentation

The evaluator shall check the guidance documentation and ensure that it provides an example of each auditable event required by FAU_GEN.1 (i.e. at least one instance of each auditable event – comprising the mandatory, optional and selection-based SFR sections as applicable – shall be provided from the actual audit record).

The evaluator examined the provided operational guidance (Ref. [9]) and determined that it lists all auditable events and provides a format for audit records.

All audit event types mandated by the cPP are described and the description of the field contains the required information as per FAU_GEN.1.2.

Table 5 in Chapter 7 of the guidance (Ref. [9]) describes each of the fields in the event logs. These include:

- Timestamp;
- Hostname;
- Process;
- Process ID;
- TAG;
- Username; and
- Message Text.

Table 4 in Chapter 6 of the guidance (Ref. [9]) contains audit examples of all 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.

The evaluator has examined the provided operational guidance and determined that it provides for sufficient details to implement the mechanisms in the TOE that are necessary to enforce the requirements specified in the cPP.

The Configuration Guide (Ref. [9]) provides the CLI commands and configuration examples necessary to place the device into its evaluated configuration and to enforce the requirements specified in the Security Target (Ref. [7]). The evaluator has found that the guide provides the necessary information for the TOE to operate in its evaluated configuration.

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.

Note that the testing here can be accomplished in conjunction with the testing of the security mechanisms directly.

The evaluator tested the ability of the TOE to generate audit records for the relevant events listed in the ST. The evaluator tested separately the logging of administrative clearing of logs and reboot of the system. To this end, the evaluator cleared the logs and requested a reboot of the system. After the reboot, the evaluator examined the logs and determined that the steps of clearing the logs and rebooting the TOE were logged.

The verification of audit-log functions for other events were tested throughout the rest of the test-plan.

4.2.2 FAU_GEN.2 User identity association

TSS & Guidance Documentation

The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

Tests

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

4.2.3 FAU_STG_EXT.1 Protected audit event storage

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.

Section 7.4 of the ST indicates that Syslog can be configured to store the audit logs locally, and optionally to send them to one or more syslog log servers via Netconf over SSH.

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.

As per Sect. 7.4 of the ST, "Local audit log are stored in /var/log/ in the underlying filesystem. Only a Security Administrator can read log files, or delete log and archive files through the CLI interface or through direct access to the filesystem having first authenticated as a Security Administrator. The syslogs are automatically deleted locally according to configurable limits on storage volume. The default maximum size is 1Gb. The default maximum size can be modified by the user, using the "size" argument for the "set system syslog" CLI command".

The evaluator shall examine the TSS to ensure it describes whether the TOE is a standalone TOE that stores audit data locally or a distributed TOE that stores audit data locally on each TOE component or a distributed TOE that contains TOE components that cannot store audit data locally on themselves but need to transfer audit data to other TOE components that can store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs it contains a list of TOE components that store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs that contain components which do not store audit data locally but transmit their generated audit data to other components it contains a mapping between the transmitting and storing TOE components.

As per Sect. 7.4 of the ST, the TOE consists of a single standalone component that stores logs locally and optionally to send them to one or more external syslog log servers.

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.

As per Sect. 7.4 of the ST: "The Junos OS defines an active log file and a number of "archive" files (10 by default, but configurable from 1 to 1000). When the active log file reaches its maximum size, the logging utility closes the file, compresses it, and names the compressed archive file 'logfile.0.gz'. The logging utility then opens and writes to a new active log file. When the new active log file reaches the configured maximum size, 'logfile.0.gz' is renamed 'logfile.1.gz', and the active log file is closed, compressed, and renamed 'logfile.0.gz'. When the maximum number of archive files is reached and when the size of the active file reaches the configured maximum size, the contents of the oldest archived file are deleted so the current active file can be archived

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.

As per Sect. 7.4 of the ST, when Syslog is configured to send logs to one or more syslog log servers, this done in real time, via NETCONF over SSH.

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.

Chapter 5 of the guidance (Ref. [9]) provides the necessary documentation required to set up remote logging using SSH NETCONF.

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 guidance (Ref. [9]) describes the relationship between the local audit data and the audit data that are sent to the audit log server. This is demonstrated via the following sentences in Chapter 5 of the guidance “A secure Junos OS environment requires the auditing of events and storing them in a local audit file. The recorded events are simultaneously sent to an external syslog 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.

The evaluator has examined the provided operational guidance and determined that it provides clarity on the fact that audit data is overwritten when space for audit data is full as per selection for FAU_STG_EXT.1.3 in the ST (Ref. [7]). According to the guidance (Ref. [9]) the user is able to configure the number of archived log files. These archived log files shall be overwritten when the maximum number of archived log files have been created.

Tests

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.

The evaluator established an SSH connection to the TOE and executed an XML RPC to request the transmission of logs to a dedicated log-server. The evaluator, via monitoring of the packet-capture data, verified that all logging is transmitted encrypted between the TOE and the log-server. The evaluator also verified that connectivity between the log-server and the TOE is restored after the connection is physically interrupted.

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).

The evaluators generated audit data and confirmed that these audit files were stored within the TOE file system. The evaluators confirmed that, upon exhausting the local storage space, the TOE deleted the oldest log file and created a new file to write to. This behaviour is consistent with FAU_STG_EXT.1.

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

The TOE does not claim compliance with FAU_STG_EXT.2/LocSpace.

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.

The TOE is not distributed and, as such, this test was not applicable.

4.3 Cryptographic Support (FCS)

4.3.1 FCS_CKM.1 Cryptographic Key Generation

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.

Sect. 7.1.1 of the ST indicates: "Asymmetric keys are also generated in accordance with FIPS PUB 186-4 Appendix B.3 for RSA Schemes and Appendix B.4 for ECC Schemes for SSH communications. Key lengths are listed in Table 11. The TOE implements Diffie-Hellman group 14, using the modulus and generator specified by Section 3 of RFC3526."

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.

The Configuration Guide (Ref. [9]) describes how the administrator can configure SSH (in Chapter 4). As part of these configuration guides, the available cryptographic methods and associated key sizes are indicated with configuration examples for how to set these values appropriately.

Tests

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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfies the requirements for FCS_CKM.1 RSA key generation contained in Addendum #2 of NIAP Policy Letter #5² of 6 December 2019.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-384 and 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-384 and 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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfy the requirements for FCS_CKM.1 ECC key generation and verification contained in Addendum #2 of NIAP Policy Letter #5³ of 6 December 2019.

² https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

³ https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

Diffie-Hellman Group 14

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

4.3.2 FCS_CKM.2 Cryptographic Key Establishment**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.

Table 7 of the ST, Sect. 7.1.1, lists the following DH key exchange protocols (for SSHv2):

- ecdh-sha2-nistp256
- ecdh-sha2-nistp384
- ecdh-sha2-nistp521
- Diffie-Hellman group 14 (modp 2048)

This is consistent with FCS_CKM.1.1 and FCS_CKM.2.1. Further, it is stated that “The TOE implements Diffie-Hellman group 14, using the modulus and generator specified by Section 3 of RFC3526.”

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).

The Evaluated Configuration Guide (Ref. [9]) describes how the administrator can configure SSH (in Chapter 4). As part of these configuration guides, the available cryptographic methods and associated key sizes are indicated with configuration examples for how to set these values appropriately.

Tests**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 MAC tags 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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfy the requirements for FCS_CKM.2 ECC-based key establishment schemes contained in Addendum #2

of NIAP Policy Letter #5⁴ of 6 December 2019. Specifically, CAVP certificate C1543 includes appropriate KAS-ECC Component and KDFs functions.

SP800-56B 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 KTSOAEP 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 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 KTSOAEP 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

⁴ https://www.niap-ccvcs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

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.

The TOE does not claim SP800-56B key establishment and, hence, this test is not applicable.

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.

The evaluators performed this test as part of the test for FCS_SSHS_EXT.1, where the SSH service on the TOE is configured to only use Diffie Hellman group 14 against a known, good implementation of the Diffie Hellman group 14 provided by the SSH client that is provided with Kali Linux.

4.3.3 FCS_CKM.4 Cryptographic Key Destruction

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 for2). 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.

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.

Table 8 of the ST provides all the required information. For each CSPs (SSH Private Host Key, SSH Session Keys, User Password and DRBG State), Table 8 describes the following:

- usage,
- method of storage (plaintext or hashed),
- storage location (SSD or memory);
- zeroization method (several methods).

The set of CSP covers all cryptographic uses specified in the ST.

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.

No exception is identified.

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 and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Chapter 2 of the Evaluated Configuration Guide (Ref. [9]) describes how the administrator can perform a zeroisation of the TOE. This will ensure that all Critical Security Parameters (CSPs) are wiped from the TOE. There are no instances where key destruction may be delayed at the physical layer.

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

Tests

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 $1 < i \leq 10$. 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 $1 < i \leq 10$. 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.
- a) 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.
- b) 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 evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

AES-CTR Known Answer Tests:

There are four Known Answer Tests (KATs) described below. For all KATs, the plaintext, IV, and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator 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, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from 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, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input.

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

KAT-3 To test the encrypt functionality, the evaluator shall supply the two sets of key values described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second 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, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from decryption of the given ciphertext using the given key values 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 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, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from encryption of the given plaintext using a 128-bit key value of all zeros and using a 256 bit key value of all zeros, respectively, and an IV of all zeros. Plaintext value i in each set shall have the leftmost bits be

ones and the rightmost 128-i bits be zeros, for i in [1, 128]. To test the decrypt functionality, 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.

AES-CTR Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key, IV, and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. 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 by decrypting an i-block message where 1 less-than i less-than-or-equal to 10. For each i the evaluator shall choose a key and a ciphertext message of length i blocks and decrypt the message, using the mode to be tested, with the chosen key. The plaintext shall be compared to the result of decrypting the same ciphertext message with the same key using a known good implementation.

AES-CTR Monte-Carlo Test

The evaluator shall test the encrypt functionality using 200 plaintext/key pairs. 100 of these shall use 128 bit keys, and 100 of these shall use 256 bit keys. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

Input: PT, Key

for i = 1 to 1000:

CT[i] = AES-CTR-Encrypt(Key, PT) PT = CT[i]

The ciphertext computed in the 1000th iteration 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.

There is no need to test the decryption engine.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfies the requirements for FCS_COP.1 for AES (CBC, CTR) contained in Addendum #2 of NIAP Policy Letter #5⁵ of 6 December 2019.

⁵ https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

4.3.5 FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)

Tests

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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfies the requirements for FCS_COP.1 ECDSA signature generation and verification contained in Addendum #2 of NIAP Policy Letter #5⁶ of 6 December 2019.

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.

⁶ https://www.niap-ccv.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

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

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificate C1543 claimed in the ST satisfies the requirements for FCS_COP.1 RSA signature generation and verification contained in Addendum #2 of NIAP Policy Letter #5⁷ of 6 December 2019.

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

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.

Table 6 of the ST provides complete mapping between primitives and hash functions.

Guidance Documentation

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

The Evaluated Configuration Guide (Ref. [9]) describes how the administrator can configure SSH (in Chapter 4). As part of these configuration guides, the available cryptographic methods and associated key sizes are indicated with configuration examples for how to set these values appropriately.

Tests

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

⁷ https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

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 i th message is $m + 99*i$, where $1 \leq i \leq m$. 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 i th message is $m + 8*99*i$, where $1 \leq i \leq m/8$. 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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificates C1541, C1542, and C1543 claimed in the ST satisfy the requirements for FCS_COP.1 SHS contained in Addendum #2 of NIAP Policy Letter #5⁸ of 6 December 2019.

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

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.

⁸ https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

Table 13 of the ST lists the supported HMAC functions, lengths, hash functions, block sizes and output MAC sizes.

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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificates C1541, C1542, and C1543 claimed in the ST satisfy the requirements for FCS_COP.1 HMAC-DRBG contained in Addendum #2 of NIAP Policy Letter #5⁹ of 6 December 2019.

4.3.8 FCS_RBG_EXT.1 Extended: Cryptographic Operation (Random Bit Generation)

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.

Section 7.1.2 specifies the DRBG of the TOE as HMAC-DRBG using SHA-256, seeded from five software-based noise sources. Further details, including an entropy estimation analysis, are provided in the Entropy Assessment Report which was submitted to the evaluation lab and scheme. The minimum bits of entropy needed to seed is 256 bits.

Guidance Documentation

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

The DRBG utilised by the TOE is non-configurable by the Administrator and is automatically used by the TOE. As such, this requirement is non-applicable.

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

⁹ https://www.niap-ccv.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

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) unstantiate. 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 \leq 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.

This assurance activity was carried out via validation of CAVP certification claims, as discussed in Section 7.4.1. The evaluator confirmed that CAVP certificates C1541 and C1543 claimed in the ST satisfy the requirements for FCS_RBG_EXT.1 contained in Addendum #2 of NIAP Policy Letter #5¹⁰ of 6 December 2019.

4.4 Identification and Authentication (FIA)

4.4.1 FIA_AFL.1 Authentication Failure Management

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.

¹⁰ https://www.niap-ccvcs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

The evaluator shall examine the TSS to confirm that 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).

Section 7.2 of the ST describes how successive unsuccessful authentication attempts are detected and tracked. It explains how the TOE can be configured to specify the action to be taken if the administrator fails to enter valid username/password credentials for password authentication when attempting to authenticate via remote access.

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 has examined the guidance (Ref. [9]) to ensure that instructions for configuring the number of successive unsuccessful authentication attempts for authentication are provided. The "Limiting the Number of User Logins Attempts for SSH Sessions" section in Chapter 4 of the guidance (Ref. [9]) provides detailed commands for the administrator to enter in order to set the correct lock-out period; tries before disconnection; back-off threshold (the delay after an unsuccessful attempt); back-off factor (the factor by which the delay increases after each unsuccessful attempt); and whether the root user may be allowed to login via SSH.

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.

The User Access and Authentication User Guide (Ref. [12]) is clear in indicating that administrator access will always be maintained, even if remote administration is made unavailable due to blocking of accounts as a result of FIA_AFL.1. In the "Junos Login Settings" section it states "The device prevents the locked users to perform activities that require authentication, until a security administrator manually clears the lock or the defined time period for the device to remain locked has elapsed. However, the existing locks are ignored when the user attempts to log in from the local console."

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.

The evaluator configured the TOE to lock-out access for a user after 3 failed attempts at authentication via SSH. The evaluator also configured the number of minutes for a lockout period to be 5 minutes. This configuration was then validated by the evaluator by attempting a login via SSH with incorrect password credentials 3 times. Having been locked out, the evaluator was only able to log in with the correct credentials after 5 minutes, or via the serial console using the credentials of the user that's locked out. Since the locking of a user-account only applies to SSH connections, the evaluator was able to still access the system via the serial console.

4.4.2 FIA_PMG_EXT.1 Password Management

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

Chapter 3 of the Evaluated Configuration Guide (Ref. [9]) identifies the characters that may be used in the password as "both alphanumeric and punctuation characters, composed of any combination of upper and lowercase letters, numbers, and special characters such as, "!", "@", "#", "\$", "%", "^", "&", "*", "(", and ")"". Furthermore, it states that "there should be at least a change in one case, one or more digits, and one or more punctuation marks"

The guide provides commands for administrators to set the password policy via the following commands:

```
set system login password minimum-length 10
set system login password change-type character-sets
set system login password minimum-changes 3
set system login password change-type character-sets
```

In addition to this, the guide states that hashing algorithm for user passwords can be either SHA256 or SHA512 and provides the following command to set it:

```
set system login password format sha512
```

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.

To perform this test, the evaluator configured the maximum and minimum length of the accepted passwords and the minimum change in character sets as per guidance documentation. The evaluator then tried different sets of passwords that are expected to pass and fail the password requirements enforced by the TOE and confirmed that the TOE behaved as expected.

4.4.3 FIA_UIA_EXT.1 User Identification and Authentication

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".

Section 7.2 of the ST describes the logon process for each logon method allowed (local console and SSH). The Authentication process and library are login() and PAM Library module.

Following TOE initialization, the login() process is listening and can be accessed through either direct connection to the local console or following successful establishment of a remote

management connection over SSH, when a login prompt is displayed. For password authentication, login() interacts with a user to request a username and password to establish and verify the user's identity. The SSH daemon also supports public key authentication of clients.

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.

Section 7.2 of the ST specifies that prior to authentication, the only Junos OS managed responses provided to the administrator are:

- Negotiation of SSH session
- Display of the access banner
- ICMP echo responses.

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.

The Evaluated Configuration Guide provides:

- Guidance on configuring administrative credentials and privileges (Chapter 3); and
- Guidance on configuring SSH and Console Connections (Chapter 4).

An administrator successfully authenticates to the TOE by providing a username and password combination matching the stored credentials (for both console and SSH).

There is no configuration required to limit services available prior to login.

The evaluator was able to successfully configure the administrative credentials and privileges using the information presented in the guidance.

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.

The evaluator tested all methods of authentication to the TOE: console, password-based SSH login and public-key based SSH login. The evaluator carried out a successful login using each one of these methods. The evaluator also carried out an unsuccessful attempt at login using each one of these methods. A login-banner was displayed where appropriate. Furthermore, all attempts at login were recorded in the TOE's syslog in the appropriate format.

- 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.

The evaluator carried out an nmap scan on the IP address of the TOE for all available protocols and ports. The only protocols identified were TCP and ICMP. The only TCP service that was available on the TOE was SSH and NETCONF (830) as expected.

- 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.

The only other service prior to the local administrator logging in to the TOE is the OAM shell. The evaluator accessed the shell and confirmed that no services are available to the administrator before logging in.

4.4.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

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.

4.4.5 FIA_UAU.7 Protected Authentication Feedback

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.

The evaluator confirmed that the TOE does not provide any display output/feedback when passwords are entered as part of the user login process.

4.5 Security management (FMT)

4.5.1 FMT_MOF.1/ManualUpdate

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).

As per Chapter 2 of the guidance (Ref. [9]) the system software can be updated via the following commands:

```
request system software add <filename>
request system reboot
```

The TOE is not in a distributed form. As such, the second requirement is not 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.

To execute this test the evaluator copied the firmware image on to a USB flash drive. This was then mounted on the TOE by a user with admin privileges. The evaluator then connected to the TOE, via SSH, as a user with no administrator privileges and attempted to install the update by entering the corresponding CLI commands. The attempt failed.

4.5.2 FMT_MTD.1/CoreData Management of TSF Data

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.

As per Section 7.2 of the ST, prior to authentication, the only Junos OS managed responses provided to the administrator are:

- Negotiation of SSH session
- Display of the access banner
- ICMP echo responses.

If the TOE supports handling of X.509v3 certificates and implements a trust store, the evaluator shall examine the TSS to determine that it contains sufficient information to describe how the ability to manage the TOE's trust store is restricted.

This EA is not applicable.

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

The documentation (Ref. [9]) groups functionality into chapters (administrative credentials and privileges, SSH, event logging, etc.), which allows for simple identification of which functions are applicable to the requirements of the cPP.

The TOE implements a single role, that of the authorised administrator. As such, no configuration is required to restrict access to TOE functions and TSF data.

If the TOE supports handling of X.509v3 certificates and provides a trust store, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to configure and maintain the trust store in a secure way. If the TOE supports loading of CA certificates, the evaluator shall review the guidance documentation to determine that it provides sufficient information for the administrator to securely load CA certificates into the trust store. The evaluator shall also review the guidance documentation to determine that it explains how to designate a CA certificate a trust anchor.

The TOE does not support the handling of X.509v3 certificates. As such, this requirement is not applicable

4.5.3 FMT_SMF.1 Specification of Management Functions

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).

Section 7.6 of the ST specifies the security management functions available via the serial port on the device or remotely over SSH. These functions correspond to those described in the guidance documentation as well as those observed by the evaluators during exercising of the TOE.

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.

4.5.4 FMT_SMR.2 Restrictions on security roles

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.

The TOE is administered locally via the console port or remotely via SSH. The evaluator found the guidance document (Ref. [9]) to provide all the necessary instructions for administering the TOE both locally and remotely.

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.

This SFR is addressed by all the tests carried out by the evaluator.

4.5.5 Protection of the TSF (FPT)

4.5.6 FPT_SKP_EXT.1 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)

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.

As per Section 7.1.1, "Junos OS does not provide a CLI interface to permit the viewing of keys. Cryptographic keys are protected through the enforcement of kernel-level file access rights,

limiting access to the contents of cryptographic key containers to processes with cryptographic rights or shell users with root permission. Security Administrators do not have root permission in shell.”

4.5.7 FPT_APW_EXT.1 Protection of Administrator Passwords

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.

As per Section 7.2 of the ST, Locally stored authentication credentials are protected:

- The password is hashed when stored using hmac-sha1, sha256 or sha512.
- Authentication data for public key-based authentication methods are stored in a directory owned by the user (and typically with the same name as the user). This directory contains the files '.ssh/authorized_keys' and '.ssh/authorized_keys2' which are used for SSH public key authentication.

4.5.8 FPT_TST_EXT.1 TSF testing

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.

Section 7.3 of the ST describes the self-tests that the TOE runs. The description of the self-test is sufficient to demonstrate the correct operation of the TSF. Specifically, the self-tests ensure that only authorized executables are allowed to run thus ensuring the correct operation of the TOE.

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.

The evaluator was able to ensure that the guidance documentation (Ref. [9]) describes the possible errors that may result from such tests and the actions the administrator should take in response.

The TOE is not in a distributed form. As such, the second requirement is not applicable.

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.

The evaluator confirmed that when the TOE is rebooted, the FIPS self-tests, including firmware integrity and cryptographic functions self-tests, are performed during the reboot process.

4.5.9 FPT_TUD_EXT.1 Trusted Update

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.

Security Administrators are able to query the current version of the TOE firmware using the CLI command "show version".

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.

As per Section 7.5 of the ST, updates are downloaded and applied manually (there is no automatic updating of the Junos OS). The installable firmware package containing the Junos OS has a digital signature that is checked when the Security Administrator attempts to install the package. If verification fails, the TOE uses the last known verified image.

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.

N/A.

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.

N/A.

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.

N/A

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.

N/A

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.

Per the guidance document (Ref. [9]) the currently running version of the TOE can be queried via the show version local command.

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.

As per Chapter 1 of the Junos OS Software Installation and Upgrade Guide (Ref. [11]):

“Juniper Networks routing platforms run only binaries supplied by Juniper Networks, and currently do not support third-party binaries. Each Junos OS image includes a digitally signed manifest of executables that are registered with the system only if the signature can be validated. Junos OS will not execute any binary without a registered signature.”

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.

The TOE does not use published hashes to protect the trusted update mechanism. As such, this requirement is not applicable.

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.

The TOE is not in a distributed form. As such, this requirement is not applicable.

If this 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.

The TOE is not in a distributed form. As such, this requirement is not applicable.

If this 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.

The TOE is not in a distributed form. As such, this requirement is not applicable.

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.

The evaluator executed the 'show version' command and confirmed that the TOE output the current version of the firmware. The evaluator loaded a legitimate update file onto the device via USB and, using the commands specified in the Installation and Upgrade Guide, confirmed that the TOE successfully installed the new firmware image. The TOE does not support delayed activation of updates.

- 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.

The evaluator executed the 'show system version' command via the CLI and confirmed that it indicated a version different to that of the update file to be applied. The evaluator attempted to apply modified updates (modified via hex editor, unsigned firmware file or signed with an invalid development key) and confirmed that, in each instance, the TOE rejected the update file. The TOE does not support delayed activation of updates.

4.5.10 FPT_STM_EXT.1 Reliable Time Stamps

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.

Section 7.4 of the ST explains that the clock function of Junos OS provides a source of date and time information for the appliance, used in audit timestamps, which is maintained using the hardware Time Stamp Counter as the clock source.

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.

Per the CLI guide, the date/time can be set via the CLI using `set date YYYYMMDDHHMM.ss` command.

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.

The evaluator set the time manually on the TOE using the operational guidance. The evaluator waited for a period of 10 minutes and confirmed that the time is in agreement with respect to the value that was originally set.

4.6 TOE Access (FTA)

4.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

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.

As per the CLI guide (Ref. [10]) the timeout period for local (serial) connections can be set via the `set cli idle-timeout <minutes>` command.

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.

The evaluator configured several idle timeout periods for the local console connection. The evaluators confirmed that, for each time period defined, the TOE terminated the session after the period of inactivity had expired. The evaluators confirmed that, once a session had been terminated, re-authentication was required before access to the TOE was restored.

4.6.2 FTA_SSL.3 TSF-initiated Termination

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.

As per the CLI guide (Ref. [10]) the timeout period for local (serial) connections can be set via the `set cli idle-timeout <minutes>` command. This method terminates the session with the user having to log back in.

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.

The evaluators configured several idle timeout periods for the remote SSH connection. The evaluators confirmed that, for each time period defined, the TOE terminated the session after the period of inactivity had expired. The evaluators confirmed that, once a session had been terminated, re-authentication was required before access to the TOE was restored.

4.6.3 FTA_SSL.4 User-initiated Termination

Guidance Documentation

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

As per the CLI guide (Ref. [10]) any active CLI session can be closed from the CLI via the exit command.

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

The evaluator established an administrative session via the local console. Once the session had been established, the evaluator executed the 'exit' command and confirmed that the session was terminated.

The evaluator established an administrative session via the remote SSH. Once the session had been established, the evaluator executed the 'exit' command and confirmed that the session was terminated.

4.6.4 FTA_TAB.1 Default TOE Access Banners

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).

As per Section 7.2 of the ST, "Junos enables Security Administrators to configure an access banner for local and remote SSH connections provided with the authentication prompt. The banner can provide warnings against unauthorized access to the secure switch as well as any other information that the Security Administrator wishes to communicate."

Guidance Documentation

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

As per Chapter 4 of the Evaluated Configuration Guide (Ref. [9]) the login banner can be set via the <set system login message login-message-banner-text> command.

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.

The evaluator configured a warning and consent message using the command specified in the guidance documentation. The evaluator confirmed that the configured message was displayed when connecting to the TOE via both local and remote administrative channels.

4.7 Trusted path/channels (FTP)

4.7.1 FTP_ITC.1 Inter-TSF trusted channel

TSS

The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

As described in Section 7.1.3, Junos OS provides an SSH server to support Trusted Channels using SSHv2 protocol which ensures the confidentiality and integrity of communication with the remote audit server. Export of audit information to a secure, remote server is achieved by setting up an event trace monitor that sends event log messages by using NETCONF over SSH to the remote system event logging server. The remote audit server initiates the connection. The SSHv2 protocol ensures that the data transmitted over a SSH session cannot be disclosed or altered by using the encryption and integrity mechanisms of the protocol with the FIPS cryptographic module.

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.

The TOE utilises SSH for communication between itself and remote identities. External logging is supported and logs are transferred over SSH to a remote server using NETCONF.

The Evaluated Configuration Guide (Ref. [9]) provides instructions for configuring SSH and the transfer of logs using NETCONF via SSH. In the event that the connections are broken, the TOE shall attempt to reconnect to the remote device.

Tests

The vendor shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

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.

Testing of SSH is performed as part of other evaluation activities.

- 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.

The evaluator confirmed by observing the traffic data using Wireshark that the TOE can initiate communication via the trusted channel (NETCONF over SSH) to send audit records to the syslog server and that the data is encrypted

- d) Test 4: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities

The evaluator established an SSH connection between the TOE and a peer before physically interrupting communications. The connection was able to be recovered after a physical interruption when the disconnection period is shorter than the application layer timeout. The connection is terminated by the TOE if the disconnection duration is greater than the application layer timeout. No data was transmitted during the periods of disconnection.

4.7.2 FTP_TRP.1/Admin Trusted Path

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.

Section 7.1.3 of the SP indicates the TOE implements SSHv2 protocol as an administrative trusted path, in order to ensure the confidentiality and integrity of user remote sessions. This matches the cryptographic SFRs defined 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.

The TOE only supports SSH for establishing the remote administrative sessions. For SSH connections, a valid username and password or SSH key must be provided to access the TSF. The SSH client that is used must support the ciphers/key exchange methods used by the TOE in its evaluated configuration. This is confirmed by the Evaluated Configuration Guide (Ref. [9]) and the CLI Guide (Ref. [10]).

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.

Testing of remote administration via SSH is performed as part of other evaluation activities

- b) Test 2: The evaluator shall ensure, for each communication channel, the channel data is not sent in plaintext.

The evaluators performed Wireshark monitoring for SSH and confirmed that data sent via this channel was not transmitted in plaintext.

5 Evaluation Activities for Optional Requirements

5.1 Security Audit (FAU)

5.1.1 FAU_STG.1 Protected audit trail storage

TSS

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

As per Section 7.4 of the ST, “local audit logs are stored in /var/log/ in the underlying filesystem. Only a Security Administrator can read log files or delete log and archive files through the CLI interface or through direct access to the filesystem having first authenticated as a Security Administrator. The syslogs are automatically deleted locally according to configurable limits on storage volume. The default maximum size is 1Gb. The default maximum size can be modified by the user, using the “size” argument for the “set system syslog” CLI command.”

Guidance Documentation

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

There is no configuration required to protect locally stored audit data. As such, this requirement is not applicable.

Tests

The evaluator shall perform the following tests:

- a) Test 1: The evaluator shall access the audit trail without authentication as Security Administrator (either by authentication as a non-administrative user, if supported, or without authentication at all) and attempt to modify and delete the audit records. The evaluator shall verify that these attempts 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 access the audit trail 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 TOE, in the evaluated configuration, defines a single role, that of the Security Administrator. The evaluators confirmed that access control mechanisms prevent non-authenticated users from accessing the audit logs.

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

The evaluator connected via SSH to the TOE as an authorised administrator. From the CLI, the evaluator entered the command to clear the logs selected by the evaluator. The evaluator confirmed, through accessing the shell on the TOE, that only the log files previously selected had been removed from the TOE

6 Evaluation Activities for Selection-Based Requirements

6.1 Cryptographic Support (FCS)

6.1.1 FCS_SSHS_EXT.1 SSH Server

TSS

FCS_SSHS_EXT.1.2

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.

Section 7.1.3 of the ST describes the acceptable public key algorithms for authentication, namely:

- ssh-rsa
- rsa-sha2-256,
- rsa-sha2-512,
- ecdsa-sha2-nistp256
- ecdsa-sha2-nistp384
- ecdsa-sah2-nistp521

The TOE supports password-based authentication for SSH as described in Section 7.2.

FCS_SSHS_EXT.1.3

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

Table 10, Section 7.1.3 of the ST, indicates that packets greater than 256Kbytes in an SSH transport connection are dropped and the connection is terminated by Junos OS.

FCS_SSHS_EXT.1.4

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.

Table 10, Section 7.1.3 of the ST, describes the optional characteristics of SSH The described supported encryption algorithms (AES-CBC-128, AES-CBC-256, AES-CTR-128, AES-CTR-256) correspond to those selected in FCS_SSHS_EXT.1.4.

FCS_SSHS_EXT.1.5

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.

Per Table 10, Section 7.1.3 of the ST, the TOE uses keys generated in accordance with “ssh-rsa”, “rsa-sha2-256”, “rsa-sha2-512”, “ecdsa-sha2-nistp256”, “ecdsa-sha2-nistp384” or “ecdsa-sha2-nistp521” to perform public-key based device authentication

FCS_SSHS_EXT.1.6

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.

As per Table 10, Section 7.1.3 of the ST, TOE permits negotiation of HMAC-SHA1 in each direction for SSH transport, according to RFC 4253, as well as HMAC-SHA2-256 and HMAC-SHA2-512, according to RFC 6668.

FCS_SSHS_EXT.1.7

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.

The TSS in Section 7.1.3 of the ST indicates that “Key exchange is performed only using one of the supported key exchange algorithms, which are ordered as follows: ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521 (all specified in RFC 5656), diffie-hellman-group14-sha1 (specified in RFC 4253).” This list corresponds to the definition of FCS_SSHS_EXT.1.7.

FCS_SSHS_EXT.1.8

The evaluator shall check that the TSS specifies the following:

- a) Both thresholds are checked by the TOE.
- b) Rekeying is performed upon reaching the threshold that is hit first.

The TSS in Section 7.1.3 of the ST indicates that “For ciphers whose blocksize ≥ 16 , the TOE rekeys every $(2^{32}-1)$ bytes. The client may explicitly request a rekeying event as a valid SSHv2message at any time and the TOE will honor this request.

Re-keying of SSH session keys can be configured using the sshd_config knob. The data-limit must be between 51200 and 4294967295 $(2^{32}-1)$ bytes and the time-limit must be between 1 and 1440 minutes.

Guidance Documentation

FCS_SSHS_EXT.1.4

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).

The evaluator was satisfied with the level of detail provided by the instructions in Chapter 4 of the Evaluated Configuration Guide (Ref. [9]) on configuring the TOE for SSH. This chapter provided steps on:

- Configuring the host-key algorithms;
- Configuring the key-exchange algorithms for Diffie-Hellman;
- Configuring the message authentication codes;
- Configuring the ciphers; and
- Configuring the maximum number of user-login attempts.

Sufficient detail was provided by these steps such that the SSH functionality conformed to the descriptions in the TSS.

FCS_SSHS_EXT.1.5

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).

Refer to guidance evaluation activity for SSHS_EXT.1-4

FCS_SSHS_EXT.1.6

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).

Sufficient detail was provided by the steps in Chapter 4 of the Evaluated Configuration Guide (Ref. [9]) such that the SSH functionality conformed to the descriptions in the TSS.

The guidance provides the following set of statements to configure the allowed data integrity algorithms:

```
set system services ssh macs hmac-sha1
set system services ssh macs hmac-sha2-256
set system services ssh macs hmac-sha2-512
```

FCS_SSHS_EXT.1.7

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.

The Evaluated Configuration Guide (Ref. [9]) describes how the administrator can configure SSH (in Chapter 4). As part of these configuration guides, the available cryptographic methods and associated key sizes are indicated with configuration examples for how to set these values appropriately.

FCS_SSHS_EXT.1.8

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.

The guidance (Ref. [9]) provides for details on the configuration of thresholds for data-based and time-based rekeying for SSH.

Tests

FCS_SSHS_EXT.1.2

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.

The evaluator configured the TOE to accept only password-based authentication for SSH connections. The evaluators then connected to the TOE from a client device and confirmed that password-based authentication could be successfully completed.

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.

The evaluator attempted to connect to the TOE via SSH and, when prompted, entered an incorrect password. The evaluator confirmed that authentication failed, and the TOE did not permit access

FCS_SSHS_EXT.1.3

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

The evaluators established an SSH between a client and the TOE. The evaluators then sent a packet of just over 400KB in size and confirmed that the packet was dropped by the TOE.

FCS_SSHS_EXT.1.4

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.

Per the guidance documentation, the evaluators configured the TOE to only offer those algorithms and cryptographic primitives specified in this requirement. The evaluators then commenced session establishment between a remote client and the TOE while monitoring network traffic between the two. Evaluators confirmed that the server KEXINIT packet contained only those algorithms specified in this requirement.

FCS_SSHS_EXT.1.5

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.

The evaluator connected to the TOE from an SSH client using each of the public key algorithms specified in this requirement (ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521). Via Wireshark analysis, the evaluator was able to confirm successful authentication, negotiation and establishment of an SSH session.

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.

The evaluators attempted to authenticate to the TOE via SSH using a private key (ecdsa-sha2-nistp256) whose corresponding public key was not configured on the TOE for authentication. The evaluators confirmed that the TOE rejected the provided key and did not permit access to TSF data or services.

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.

The evaluators created a 1024-bit DSA key pair for use in public key authentication. Attempts to load this key onto the TOE for use in SSH public-key authentication were met with an error (as the TOE only permits RSA and ECDSA keys of sizes specified in FCS_SSHS_EXT.1).

FCS_SSHS_EXT.1.6

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.

The evaluator opened multiple SSH connections to the TOE with each connection restricted to one of the supported integrity algorithms. The evaluator examined the packet-capture of the negotiated connections and confirmed that the TOE permitted the exclusive use of hmac-sha1, hmac-sha2-256 and hmac-sha2-512 for SSH connections.

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.

The evaluator attempted to establish an SSH connection to the TOE by restricting the connection attempt to the hmac-sha1-96 integrity algorithm. The TOE did not permit the use of the MAC algorithm.

FCS_SSHS_EXT.1.7

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.

The evaluator configured an SSH client to only use diffie-hellman-group1-sha1 for key exchange and attempted to connect to the TOE. The evaluators confirmed that the TOE rejected this authentication attempt.

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.

The evaluator configured an SSH client to use each of the specified key exchange methods (dh-group14-sha1, ecdh-sha2-nistp521, ecdh-sha2-nistp256 and ecdh-sha2-nistp384) in turn. The evaluator confirmed that, for each specified key exchange method, the TOE permitted the connection and successfully established an SSH session

FCS_SSHS_EXT.1.8

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 session 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 of 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).

In cases where data transfer threshold could not be reached due to hardware limitations it is acceptable to omit testing of this (SSH rekeying based on data transfer threshold) threshold if both the following conditions are met:

- a) An argument is present in the TSS section describing this hardware-based limitation and
- b) All hardware components that are the basis of such argument are definitively identified in the ST. For example, if specific Ethernet Controller or WiFi radio chip is the root cause of such limitation, these chips must be identified.

The evaluator configured the TOE to have an SSH session rekey time of 60 minutes. The evaluators established a session from an SSH client and ensured that the session was kept alive for longer than 60 minutes. Via Wireshark traffic capture and audit log examination, the evaluators confirmed that a) the TOE initiated an SSH rekey upon reaching the 60-minute threshold; and b) an audit log was generated to indicate that the rekey event took place.

The evaluator then configured the TOE to have an SSH session rekey data limit of 1 gigabyte. The evaluator established a session from an SSH client and began to transmit traffic to exceed this threshold (via transfer of a large file). Via Wireshark traffic capture and audit log examination, the evaluators confirmed that a) the TOE initiated an SSH rekey upon reaching the 1 gigabyte threshold; and b) an audit log was generated to indicate that the rekey event took place.

The evaluator confirmed that both traffic-based and time-based thresholds for SSH rekey are configurable (as specified in the guidance documentation) and are only accessible to authorised administrators.

6.2 Security management (FMT)

6.2.1 FMT_MOF.1/Functions Management of security functions behaviour

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.

No access to TOE services and TSF data is permitted prior to authentication as a Security Administrator. As a non-authenticated user, access extends as far as the login prompt and the user must successfully authenticate before any further access is granted.

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.

The evaluator authenticated to the TOE as a Security Administrator and confirmed that, when in configuration mode, the functionality was provided to alter security-related parameters (e.g. cipher suites, authentication methods) for transmission of audit logs to an external entity. Evaluators confirmed that, upon committing the configuration and establishing the secure tunnel for audit log transmission, the revised configuration was used.

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.

No access to TOE services and TSF data is permitted prior to authentication as a Security Administrator. As a non-authenticated user, access extends as far as the login prompt and the user must successfully authenticate before any further access is granted.

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.

The evaluator authenticated to the TOE as a Security Administrator and confirmed that, when in configuration mode, the functionality was provided to alter security-related parameters (such as audit log size, the number of audit log files to be stored on the device, etc.) related to audit log storage. The evaluator confirmed that, once these parameters had been adjusted, the TOE took action (e.g. creating new log files, deleting older log files, etc.) as expected.

6.2.2 FMT_MOF.1/Services

Tests

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 authenticated to the device as a user that does not have security administrator privileges and attempted to run the commands to execute the FIPS self-tests and carry out a reboot. The evaluator confirmed that the attempt failed.

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.

The evaluator authenticated to the device as a security administrator and attempted to run the commands to execute the FIPS self-tests and carry out a reboot. The evaluator confirmed that the attempt was successful.

6.2.3 FMT_MTD.1/CryptoKeys Management of TSF Data

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.

The evaluator was able to confirm that no operation can be carried out without prior authentication in Junos, and that a user without security administrator privileges is not able to enter the configuration mode. Therefore, this user is unable to set any configuration material relevant to the SSH public keys for a user.

7 Evaluation Activities for SARs

7.1 ADV: Development

7.1.1 Basic Functional Specification (ADV_FSP.1)

The evaluator shall examine 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 set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

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 evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

The evaluator uses the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 3, including the EAs associated with testing of the interfaces.

It should be noted that there may be some SFRs that do not have an interface that is explicitly "mapped" to invoke the desired functionality. For example, generating a random bit string, destroying a cryptographic key that is no longer needed, or the TSF failing to a secure state, are capabilities that may be specified in SFRs, but are not invoked by an interface.

However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

Relevant TSFIs, per NDcPP-SD (Ref. [6]), are those used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates), as well as interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs).

According to guidance documentation (Ref.[9]) and ST (Ref. [7]) the relevant TSFIs are the Junos CLI, which can be accessed by administrators either via a directly attached serial connection or remotely via SSH;

The purpose of these interfaces is clear from the ST and guidance documentation. Furthermore, the guidance documentation contains detailed instructions on how to administer the TOE via the CLI.

7.2 AGD: Guidance Documents

7.2.1 Operational User Guidance (AGD_OPE.1)

The evaluator shall ensure the Operational guidance documentation is 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 evaluators were able to verify that there is reasonable guarantee that administrators and users will be made aware of the existence and role of the documentation in maintaining an evaluated configuration. The documentation relating to maintaining an evaluated configuration is made publicly available on Juniper's website.

The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

The Evaluated Configuration Guide (Ref. [9]) requires the TOE to operate in FIPS mode and covers all the operational environments that the product supports.

The evaluator shall ensure that the Operational guidance contains 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.

In order to satisfy the evaluated configuration, the TOE may only be operated in FIPS mode. When configured to operate in FIPS mode as per Evaluated Configuration Guide (Ref. [9]), a "fips" indicator is present on the CLI prompt. Also, as per the Evaluated Configuration Guide, once configured to operate in FIPS mode, the only means of not operating in FIPS mode is to zeroize the TOE. This will remove all CSPs and revert the device to factory setting.

The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.

The evaluator was able to confirm that the operational guidance was clear in identifying the functionality and interfaces that were assessed and tested. The Evaluated Configuration Guide (Ref. [9]) covers the subjects of: authentication methods, administrator credentials and privileges, SSH and console connection, remote logging, audit and event logging options and the carrying out of self-tests on the TOE.

In addition the evaluator shall ensure that the following requirements are also met.

- 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 instructions that describe at least one method of validating 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.
-
- a) In order to satisfy the evaluated configuration, the TOE may only be operated in FIPS mode. When configured to operate in FIPS mode as per Evaluated Configuration Guide (Ref. [9]), a “fips” indicator is present on the CLI prompt. As per the same guidance document, once configured to operate in FIPS mode, the only means of not operating in FIPS mode is to zeroize the TOE. This will remove all CSPs and revert the device to factory setting.
 - b)
 1. The “Downloading Software Packages from Juniper Networks” section of the of the Evaluated Configuration Guide (Ref. [9]) provides instructions on how an update can be obtained. The “Installing Software on EX Series and QFX Series devices with Single Routing Engine” section outlines the steps on how the update can be installed.
 2. The Installation and Upgrade guide (Ref. [11]) provides detailed instructions on initiating the update process as well as examples on how to determine if the upgrade was successful or otherwise. The provided examples allow for the user to check the messages that are output on successful validation of the digital signature.
 - c) The Evaluated Configuration Guide (Ref. [9]) is clear in stating to the reader that it provides for “the steps required to duplicate the configuration of the device running Junos OS when the device is evaluated.” That is, only the security functionality that is covered by the guide is covered by the Evaluation Activities.

7.2.2 Preparative Procedures (AGD_PRE.1)

The evaluator shall examine the Preparative procedures to ensure they 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 evaluator was able to identify the following methods on how the administrator is able to verify that the operational environment can fulfil its role to support the security objectives of the TOE:

- OE.PHYSICAL: The installation and upgrade guide (Ref. [11]) guide addresses this in detail.
- OE.NO_GENERAL_PURPOSE: this requirement is met implicitly as the TOE is a dedicated router platform and does not permit the installation of any general-purpose software.
- OE.NO_THRU_TRAFFIC_PROTECTION: as stated in the Security Target (Ref. [7]) the TOE does not provide any protection of the traffic that traverses it.
- OE.TRUSTED_ADMIN: This requirement is met when the administrator configures the TOE according to the Evaluated Configuration Guide (Ref. [9]). Maintenance of the TOE (for e.g. the removal of expired certificates) is subject to the update procedures and frequency employed by the administrator.
- OE.UPDATES: The Evaluated Configuration Guide (Ref. [9]) and the Junos OS Software Installation and Upgrade Guide (Ref.[11]) provide a great level of detail on how the administrator can provide updates to the TOE.
- OE.ADMIN_CREDENTIALS_SECURE: The procedures that are provided as part of the Evaluated Configuration Guide (Ref. [9]) ensure that the behaviour of the administrator and the overall security of the environment.
- OE.RESIDUAL_INFORMATION: The zeroisation procedures that are provided in the Evaluated Configuration Guide (Ref. [9]) support this objective.

The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

The evaluator was able to find that the preparative procedures are provided for every Operational Environment that the product supports.

The evaluator shall examine the preparative procedures to ensure they include instructions to successfully install the TSF in each Operational Environment.

The evaluator was able to ensure that the Evaluated Configuration Guide (Ref. [9]) provided the procedures to successfully install the TSF in the supported Operating Environments.

The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.

The instructions provided in the Evaluated Configuration Guide (Ref. [9]) allowed for the security of the TSF to be managed as a component of the operational environment.

In addition the evaluator shall ensure that the following requirements are also met.

The preparative procedures must

- a) include instructions to provide a protected administrative capability; and
- b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

- a) The “Configuring Administrative Credentials and Privileges” section of the Evaluated Configuration Guide (Ref. [9]) provides instructions on configuring protected administrative capability.
- b) The TOE does not have default values for passwords. However, as per the installation and upgrade guide (Ref. [11]), the TOE requires the root password to be configured before any changes are made to the configuration. The installation and upgrade guide (Ref. [11]) provides instructions on how the root password may be set.

7.3 ALC: Life-cycle Support

7.3.1 Labelling of the TOE (ALC_CMC.1)

When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

During testing, the evaluators issued a *show version* command. The output provided by the TOE indicated that the TOE version was 19.3R1. This output was consistent with the expected version. The TOE reference provided in the Security Target was found to be consistent with the TOE versioning used in other documentation and on the TOE itself.

7.3.2 TOE CM coverage (ALC_CMS.1)

When evaluating the developer’s coverage of the TOE in their CM system, the evaluator performs the work units as presented in the CEM.

The evaluators examined the Configuration List provided within Security Target and confirmed that it uniquely identifies the TOE and its associated evaluation evidence.

7.4 ATE: Tests

7.4.1 Independent Testing – Conformance (ATE_IND.1)

The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in 3, 5 and 6 in [NDcPP-SD]

The evaluator should consult Appendix B of the [NDcPP-SD] when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

The evaluators conducted independent testing as per ATE_IND.1. The description of the test cases and results are documented in a detailed test report (Ref. [8]) produced by the evaluators and submitted to the ACA. The test report includes all the required tests specified in NDcPP-SD (Ref. [6]) with all revisions specified by the relevant technical decisions listed in the ST. A summary of the test cases can be found in Sections 4, 5 and 6 of this AAR. All independent tests passed. Some of the tests for cryptographic functionality of the TOE were carried out via verification of CAVP certification claims. For these tests, the evaluators checked that the CAVP certificate numbers provided in the ST matched the TOE and cryptographic testing requirements specified in NDcPP-SD (Ref. [9][6]). To map NDcPP testing requirements against equivalent CAVP claims, the evaluators referred to Addendum #2 of NIAP Policy Letter #5¹¹ of 6 December 2019.

Testing was performed at Teron Labs' evaluation facility in Canberra under the oversight of the ACA. The test environment used by the evaluators is depicted in the diagram below.

¹¹ https://www.niap-ccevs.org/Documents_and_Guidance/policy-ltr-5-add2.pdf

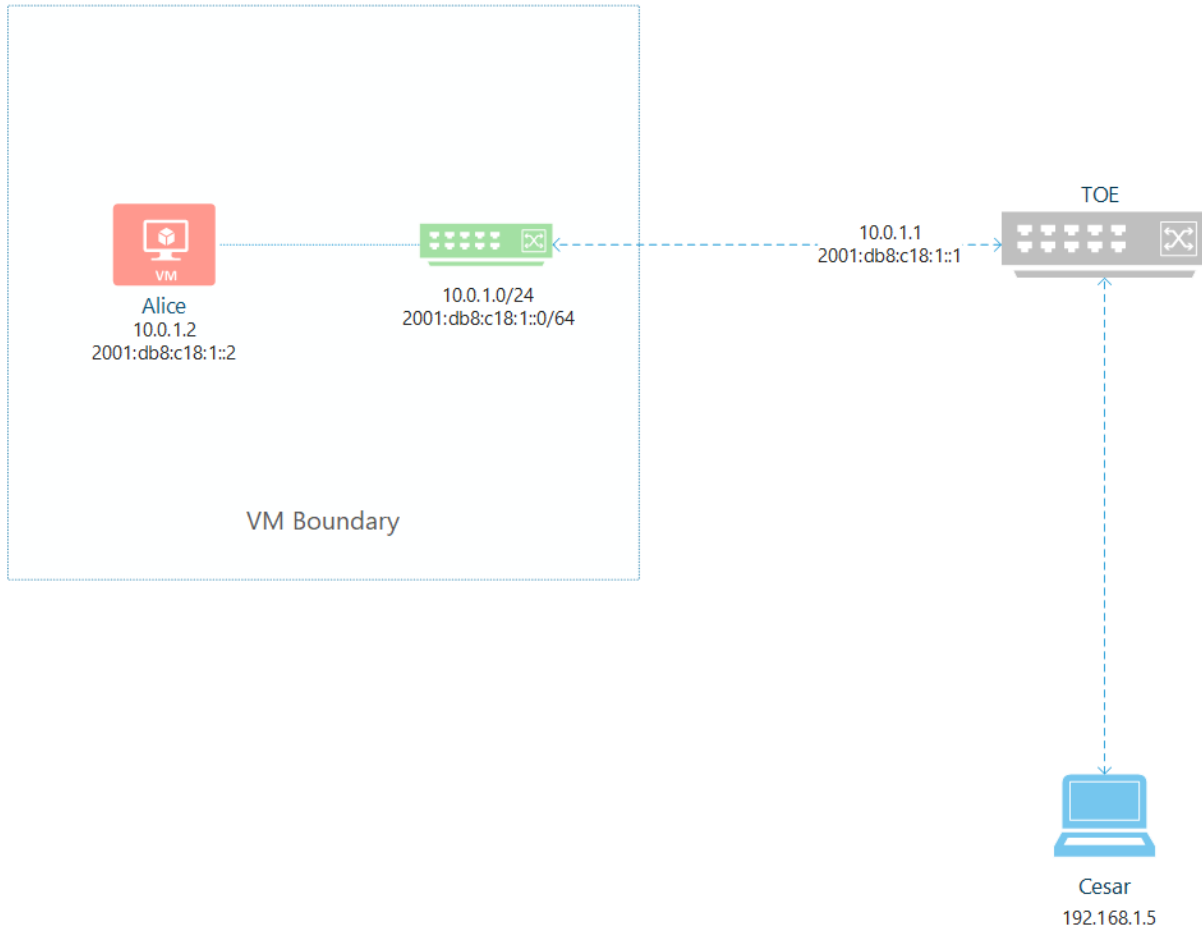


Figure 2 - Test Environment

The test environment meets the requirements set out in the NDcPP-SD (Ref. [6]). It consists of a virtual machine, Alice, a physical machine, Cesar, from which the TOE may be accessed via the console, and the TOE itself. Alice is instantiated on the VMWare ESXi 6.x server running on a Dell R810 hardware. The network connection between the TOE and the VMs is managed by a virtual switch that is also configured on the VMWare ESXi server on which Alice is hosted. Alice's prime function is to generate or instigate network traffic.

Identifier	Software	Hardware	Purpose
Alice	Kali Linux 2019.1 Rolling Release (Debian 4.19.20)	Dell Poweredge R810	<ul style="list-style-type: none"> • Network traffic generation • Syslog server

TOE	Junos 19.3R1	EX4650-48Y	• Switch
Switch 1	Virtual Switch on VMWare ESX 6.0 U3	Dell Poweredge R810	• Switch
Cesar	Windows 10 Professional	Dell Latitude 5490	• Console access • Report generation

Table 4 – Test hardware

The software used in the test environment is considered in the table below.

Name	Type	Source
Junos 19.3R1	TOE Operating System	Vendor
Scapy 2.4.0	Packet generation software	Kali Linux (pre-installed)
Python 2.7	Language run-time (used by Scapy)	Kali Linux (pre-installed)
Wireshark 2.6.6	Packet sniffing software	Kali Linux (pre-installed)
rsyslog (installed via package rsyslog version 8.40.0-1)	Syslog receiver	Kali Linux (pre-installed)
PuTTY 0.71	Console software	Internet
VMware ESXi Server 6.0 U3 (Kernel Release 5050593)	Virtualisation Host	Internet
Netmiko and Paramiko for Python 2.7	Python SSH library	Internet

7.5 AVA: Vulnerability Assessment

7.5.1 Vulnerability Survey (AVA_VAN.1)

The evaluator shall examine the documentation outlined below provided by the developer to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.

The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside the TOE) such as a web server and protocol or cryptographic libraries. This additional documentation is merely a list of the name and version number of the components, and will be used by the evaluators in formulating hypotheses during their analysis.

The evaluator formulates hypotheses in accordance with process defined in Appendix A of [NDcPP-SD]. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3 of [NDcPP-SD]. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2 of [NDcPP-SD]. The results of the analysis shall be documented in the report according to Appendix A.3 of [NDcPP-SD].

The evaluator followed the flaw hypothesis methodology. Accordingly, four types of flaw hypotheses have been considered.

Type 1 Hypotheses – Public-Vulnerability-Based

The evaluators performed a search on the sources listed in Section A.4 of the NDcPP SD (Ref. [6]) to determine a list of potential flaw hypotheses that are more recent than the publication date of the cPP, and those that are specific to the TOE and its components.

The search terms used were as follows: “router”, “switch”, “TCP”, “SSH”, “Junos”, “19.3R1”, “Juniper EX”, “Juniper EX4650”, “Juniper QFX5120”, “Juniper QFX5210”, “OpenSSL 1.0.2r” and “FreeBSD 11”.

The evaluators analysed the public vulnerabilities that resulted from the above search and concluded that no vulnerability was applicable to the TOE.

Type 2 Hypotheses – iTC-Sourced

- None defined.

Type 3 Hypotheses – Evaluation-Team-Generated

As per the NDcPP SD, Type 3 flaws are formulated by the evaluator based on information presented by the product (through on-line help, product documentation and user guides, etc.) and product behaviour during the (functional) testing activities.

During functional testing of the TOE, the evaluators have not observed any behaviour that would point to anomalous functionality or vulnerability. Similarly, the evaluators have not found elements in the product documentation that would be indicative of potential vulnerabilities, beyond what was already explored in the conducted Type 1 survey. Still, based on behaviour

observed in previous evaluations of Juniper devices performed by the evaluators, the evaluators decided to test the following race condition vulnerability:

- The evaluator shall attempt to take advantage of any race conditions in the TOE's boot-up process. This shall involve the evaluator attempting to access the system via the serial console without providing a password.

The above test passed.

Type 4 Hypotheses – Tool-Generated

As per NDcPP SD, the following protocol fuzzing vulnerability testing was considered by the evaluators:

- Examine the effects of sending mutated packets carrying each 'Type' and 'Code' value that is undefined in the relevant RFC for each of ICMPv4 (RFC 792) and ICMPv6 (RFC 4443).
- Examine the effects of mutated packets carrying each 'Transport Layer Protocol' value that is undefined in the respective RFC for IPv4 (RFC 791) IPv6 (RFC 2460) should also be covered if it is supported and claimed by the TOE.
- Examine the effect of fuzzing the remaining fields in the required protocol headers.

The above fuzz tests were performed by the evaluators. These tests passed. Thus, no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential

8 Glossary

Acronym/Term	Description
AAR	Assurance Activity Report
ACA	Australasian Certification Authority
AES	Advanced Encryption Standard
AISEF	Australasian Information Security Evaluation Facility
CA	Certificate Authority
CAVP	Cryptographic Algorithm Validation Program
CAVS	Cryptographic Algorithm Validation System
CBC	Cipher Block Chaining
CLI	Command Line Interface
cPP	Collaborative Protection Profile
CRL	Certificate Revocation List
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS 140-2	Federal Information Processing Standard 140-2
FTP	File Transfer Protocol
GCM	Galois Counter Mode
HMAC	Hash-based Message Authentication Code
ICMP	Internet Control Message Protocol
ICMPv6	Internet Control Message Protocol version 6
IKE	Internet Key Exchange
IP	Internet Protocol
IPsec	Internet Protocol Security
IPv6	Internet Protocol version 6
NDcPP	Network Device collaborative Protection Profile
RSA	Rivest-Shamir-Adleman
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm

SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
ST	Security Target
TCP	Transmission Control Protocol
TSF	TOE Security Functionality
TSFI	TSF Interface
TOE	Target of Evaluation
UDP	User Datagram Protocol
VPN	Virtual Private Network