Dell

Dell EMC Networking SmartFabric OS10.5.4

Assurance Activity Report

Version 1.2

September 2023

Document prepared by

www.lightshipsec.com

Table of Contents

1 Introduction

1 This Assurance Activity Report (AAR) documents the evaluation activities performed by Lightship Security for the evaluation identified in Table 1. The AAR is produced in accordance with National Information Assurance Partnership (NIAP) reporting guidelines.

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	NIAP Common Criteria Evaluation and Validation Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	Dell Technologies, Inc.
TOE	Dell EMC Networking SmartFabric OS10.5.4
Security Target	Dell EMC Networking SmartFabric OS10.5.4 Security Target, v2.0 [ST]
Protection Profile	collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020 [NDcPP]

1.2 Evaluation Methods

2 The evaluation was performed using the methods, tools and standards identified in Table 2.

Table 2: Evaluation Methods

Evaluation Criteria	CC v3.1R5
Evaluation Methodology	CEM v3.1R5
Supporting Documents	Evaluation Activities for Network Device cPP, December-2019, Version 2.2 [ND-SD]

Table 3: Interpretations

NDcPP v2.2e Technical Decisions	Applicable
TD0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1)	
TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	N/A: The ST does not include FCS_NTP_EXT.1
TD0536: NIT Technical Decision for Update Verification Inconsistency	

NDcPP v2.2e Technical Decisions	Applicable
TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	
TD0546: NIT Technical Decision for DTLS – clarification of Application Note 63	N/A: The TOE does not claim FCS_DTLSC_EXT.1
TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	
TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test	N/A: The ST does not claim FCS_TLSS_EXT.1
TD0556: NIT Technical Decision for RFC 5077 question	N/A: The ST does not claim FCS_TLSS_EXT.1
TD0563: NiT Technical Decision for Clarification of audit date information	
TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria	
TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7	N/A: The TOE does not claim FCS_TLSS_EXT.1 or FCS_TLSS_EXT.1
TD0570: NiT Technical Decision for Clarification about FIA_AFL.1	
TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1	
TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	
TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	
TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	
TD0591: NIT Technical Decision for Virtual TOEs and hypervisors	N/A. The TOE is not a virtual TOE
TD0592: NIT Technical Decision for Local Storage of Audit Records	
TD0631: NIT Technical Decision for Clarification of public key authentication for SSH Server	
TD0632: NIT Technical Decision for Consistency with Time Data for vNDs	

NDcPP v2.2e Technical Decisions	Applicable
TD0633: NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	N/A: The ST does not claim FCS_IPSEC_EXT.1
TD0634: NIT Technical Decision for Clarification required for testing IPv6	
TD0635: NIT Technical Decision for TLS Server and Key Agreement Parameters	N/A: The ST does not claim FCS_TLSS_EXT.1
TD0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH	N/A: The ST does not claim FCS_SSHC_EXT.1
TD0638: NIT Technical Decision for Key Pair Generation for Authentication	
TD0639: NIT Technical Decision for Clarification for NTP MAC Keys	N/A: The ST does not include FCS_NTP_EXT.1
TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	
TD0738: NIT Technical Decision for Link to Allowed-With List	

1.3 Reference Documents

Table 4: List of Reference Documents

Ref	Document
[ST]	Dell EMC Networking SmartFabric OS10 Security Target, v2.0
[AGD]	Dell EMC Networking SmartFabric OS10 Common Criteria Guide, v1.1
[ADMIN]	Dell SmartFabric OS10 User Guide Release 10.5.4, 12 2022 Rev. A05

2 Test Overview

3 Testing was performed by Kevin Steiner, Kenji Yoshino, and Nhien Truong from March 2023 through August 2023. Testing was performed in the Lightship Baltimore facility that has been accredited by NVLAP. The TOE and test setup was physically and logically protected from unauthorized access, so the integrity TOE and testing results can be assured.

2.1 TOE Components

Table 5: TOE models

Type	Model	CPU	Software	CAVP
Physical	S4112F-ON S4112T-ON S4128F-ON S4128T-ON S4148F-ON S4148T-ON MX5108n	Intel Atom C2338 (Silvermont)	Dell Networking SmartFabric OS 10.5.4	A1949
	MX9116n	Intel Atom C2538 (Silvermont)		
	S5212F-ON N3248TE-ON	Intel Atom C3338 (Goldmont)		
	S5224F-ON S5232F-ON S5248F-ON S5296F-ON Z9264F-ON	Intel Atom C3538 (Goldmont)		
	Z9432F-ON S5448F-ON	Intel Atom C3758 (Goldmont)		
	E3224F-ON	Intel Atom C3558/C3558R (Goldmont)		
	Z9332F-ON	Intel Pentium D1508 (Broadwell)		

2.2 TOE Version

4 Testing was performed on version 10.5.4.3P1.

2.3 Non-TOE Components

5 The TOE operates with the following components in the environment:

a) **Audit Server.** The TOE can send audit events to a Syslog server.

2.4 Test Environment

6 Figure 1 shows a logical view of the test setup.

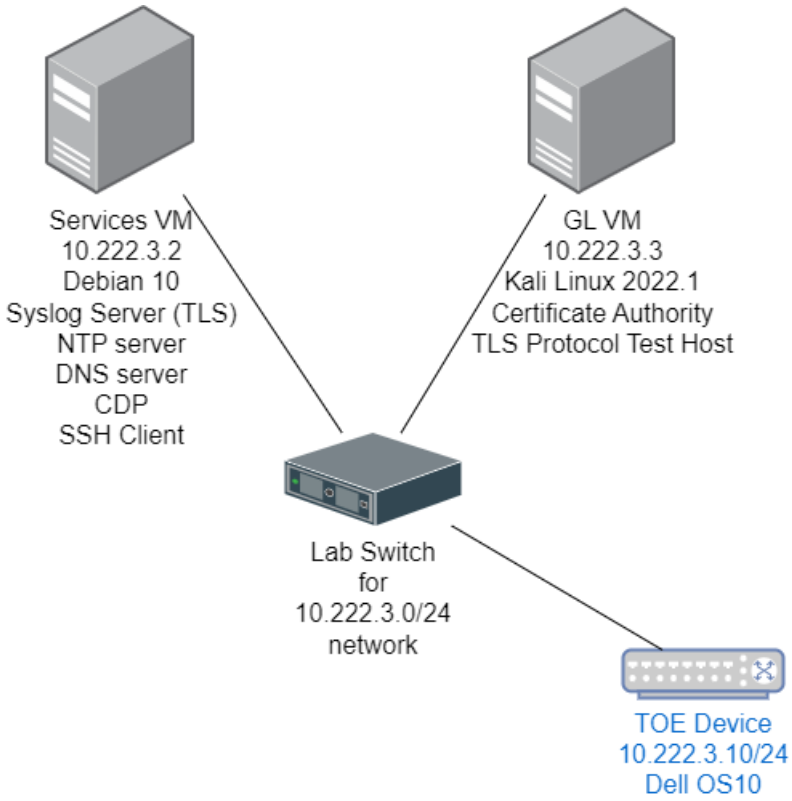

Figure 1 - Test setup

7 All DNS name resolution is under the “example.com” domain. The Services VM operated as the DNS resolver for the test setup.

2.4.1 Logging

8 The Services VM was used as the logging server.

2.4.2 Time

9 The Services VM was used as NTP server utilizing its system time. The GL VM synchronizes time with the Services VM. The time on the TOE was manually configured to match the Services VM. The Services VM, GL VM, and the TOE are configured to be in the ET time zone.

Note	The Services VM NTP server is not considered a test tool, because it does not interact with the TOE.

2.4.3 Systems

Table 6: Test Systems

Name / HW / SW	Description / Functions	Test Tools
Z9432F HW: Z9432F-ON SW: OS10.5.4.3	Fully tested TOE model	N/A
Services VM HW: Test Hypervisor SW: Debian 10	SSH Client (SSH) Perform Packet Captures Syslog Server (TLS) DNS Server CRL Distribution Point	OpenSSH 7.9p1 syslog-ng 3.19.1 dnsmasq 2.80 Wireshark 2.6.20 Python 2.7.16
GL VM Host name: lightship-USCC2203 HW: Test Hypervisor SW: Kali 2022.1	SSH Client (SSH) Protocol Test Host (TLS/SSH) Certification Authority Perform Packet Captures	Greenlight 3.0.34+0 Greenlight 3.0.35 Python 3.9.10 OpenSSL 1.1.1m Wireshark 3.6.0 OpenSSH 8.8p1 tcpdump 4.99.1
Test Hypervisor HW: Dell PowerEdge R440 SW: ESXi, 7.0.3	Hosting Services VM and GL VM	N/A
Lab Switch HPE OfficeConnect 1920S Series Switch JL382A	Connect the TOE with the testing environment.	N/A
NETGEAR Switch	Physical disconnect packet captures	N/A

Name / HW / SW	Description / Functions	Test Tools
HW: ProSafe Plus GS105E		
Packet Capture Laptop HW: Lenovo ThinkPad T15 SW: Windows 10 Pro	Physical disconnect packet captures	Wireshark 4.0.4

2.5 Test Platform Equivalency

2.5.1 Hardware Differences

10 Section 2.1 identifies the TOE models included in the evaluation.

11 All models of the TOE run the same firmware: OS10.5.4.3P1.

12 The team used the [ND-SD] Network Device Equivalency Considerations as the basis for the following equivalency rationale:

Table 7: Equivalency Factors

Factor	Evaluator Guidance	Description
Platform/ Hardware Dependencies	If there are no identified platform/hardware dependencies, the evaluator shall consider testing on multiple hardware platforms to be equivalent. If there are specified differences between platforms/hardware, the evaluator must identify if the differences affect the cPP-specified security functionality or if they apply to non-cPP-specified functionality. If functionality specified in the cPP is dependent upon platform/hardware provided services, the product must be tested on each of the different platforms to be considered validated on that particular hardware combination. In these cases, the evaluator has the option of only retesting the functionality dependent upon the platform/hardware provided functionality. If the differences only affect non-cPP-specified functionality, the variations may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP-specified functionality.	Equivalent: There are no significant platform/hardware differences that would affect the operation of the TOE. The different models of the TOE use different CPU models with different microarchitectures; however, the TOE does not utilize any microarchitecture specific compile options, so the TOE executes the same on all CPUs. Additionally, the cryptographic algorithm implementations were tested on each microarchitecture. There are differences in the network hardware and associated drivers. These differences are limited to network speed and physical layer differences. The higher level network operations remain the same on all models.

Factor	Evaluator Guidance	Description
Differences in TOE Software Binaries	If the model binaries are identical, the model variations shall be considered equivalent. If there are differences between model software binaries, a determination must be made if the differences affect cPP-specified security functionality. If cPP-specified functionality is affected, the models are not considered equivalent and must be tested separately. The evaluator has the option of only retesting the functionality that was affected by the software differences. If the differences only affect non-PP specified functionality, the models may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP specified functionality.	Equivalent: All models run the same binary.
Differences in Libraries Used to Provide TOE Functionality	If there are no differences between the libraries used in various TOE models, the model variations shall be considered equivalent. If the separate libraries are used between model variations, a determination of whether the functionality provided by the library affects cPP-specified functionality must be made. If cPP-specified functionality is affected, the models are not considered equivalent and must be tested separately. The evaluator has the option of only retesting the functionality that was affected by the differences in the included libraries. If the different libraries only affect non-PP specified functionality, the models may still be considered equivalent. For each different library, the evaluator must provide an explanation of why the different libraries do or do not affect cPP specified functionality.	Equivalent: There are no differences between the libraries used in the different TOE models.
TOE Management Interface Differences	If there are no differences in the management interfaces between various TOE models, the model variations shall be considered equivalent. If the product provides separate interfaces based on the model variation, a determination must be made of whether cPP-specified functionality can be configured by the different interfaces. If	Equivalent: There are no differences between the management interfaces for different TOE models.

Factor	Evaluator Guidance	Description
	the interface differences affect cPP-specified functionality, the variations are not considered equivalent and must be separately tested. The evaluator has the option of only retesting the functionality that can be configured by the different interfaces (and the configuration of said functionality). If the different management interfaces only affect non-PP specified functionality, the models may still be considered equivalent. For each management interface difference, the evaluator must provide an explanation of why the different management interfaces do or do not affect cPP specified functionality.	
TOE Functional Differences	If the functionality provided by different TOE model variation is identical, the models variations shall be considered equivalent. If the functionality provided by different TOE model variations differ, a determination must be made if the functional differences affect cPP specified functionality. If cPP-specific functionality differs between models, the models are not considered equivalent and must be tested separately. In these cases, the evaluator has the option of only retesting the functionality that differs model-to-model. If the functional differences only affect non-cPP specified functionality, the model variations may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP specified functionality.	Equivalent: There are no differences in functionality provided by different TOE models

13 In summary, the evaluation team performed full testing on Z9432F-ON. All other models are considered equivalent to the tested model.

3 Evaluation Activities for Mandatory SFRs

3.1 Security Audit (FAU)

3.1.1 FAU_GEN.1 Audit data generation

3.1.1.1 TSS

14 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.

Findings
PASS
[ST] section 6.1.1 indicates the file path or CN is logged to identify the key, or the key is implicitly identified because there is only a single type of the type (i.e., SSH hostkey).

15 For distributed TOEs the evaluator shall examine the TSS to ensure that it describes which of the overall required auditable events defined in FAU_GEN.1.1 are generated and recorded by which TOE components. The evaluator shall ensure that this mapping of audit events to TOE components accounts for, and is consistent with, information provided in Table 1, as well as events in Tables 2, 4, and 5 (where applicable to the overall TOE). This includes that the evaluator shall confirm that all components defined as generating audit information for a particular SFR should also contribute to that SFR as defined in the mapping of SFRs to TOE components, and that the audit records generated by each component cover all the SFRs that it implements.

Findings
PASS
The TOE is not a distributed TOE.

3.1.1.2 Guidance Documentation

16 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).

Findings
PASS
[AGD] section 2.14 provides an example of each auditable event.

17 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.

Findings
PASS
[AGD] sections 2.4 through 2.13 provide instructions for using the TOE according to the requirements specified in the [ST], including commands to run and compliant parameters. [AGD] section 1.3.3 provides general guidance regarding the scope of the evaluated functionality.

3.1.1.3 Tests

18 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.

High-Level Test Description
Examine the audit logs generated by the TOE while performing testing. Verify an audit log is generated for each auditable event and that the audit logs match the example provided in guidance.
Findings
PASS – The evaluator confirmed the TOE generates audit logs for all required auditable events, the audit logs contain the information required by FAU_GEN.1.2 and the Additional Audit Record Contents column, and the audit logs are consistent with the examples provided in guidance.

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

Test Not Applicable	The TOE is not a distributed TOE.

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

3.1.2 FAU_GEN.2 User identity association

3.1.2.1 TSS & Guidance Documentation

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

3.1.2.2 Tests

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

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

Test Not Applicable	The TOE is not a distributed TOE.

3.1.3 FAU_STG_EXT.1 Protected audit event storage

3.1.3.1 TSS

24 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.

Findings
PASS
[ST] section 6.1.2 claims that TOE sends logs to the external audit server using syslog over TLS.

25 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.

Findings
PASS
[ST] section 6.1.2 indicates the TOE rotates log files when they reach 1GB in size and keeps a history of 5 files. The auditable events are contained in the “Audit Log” and “Event Log.”
[ST] section 6.1.2 indicates logs are protected, because, “Only authorized administrators may view audit records and no capability to modify the audit records is provided.”

26 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.

Findings
PASS
[ST] section 6.1.2 states, “The TOE is a standalone TOE that stores data locally.”

27 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.

Findings
PASS
[ST] section 6.1.2 indicates the TOE overwrites the oldest log file according to the log rotation rules.

28 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.

Findings
PASS
[ST] section 6.1.2 indicates that audit information is transmitted in real-time.

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

Findings
PASS
The TOE is not a distributed TOE.

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

Findings
PASS
The TOE is not a distributed TOE.

3.1.3.2 Guidance Documentation

31 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.

Findings
PASS
[AGD] section 2.10.1 and [ADMIN] section ‘System logging over TLS’ describe how to configure a TLS trusted channel with an audit server. These descriptions include configuring mutual authentication for the channel.

32 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.

Findings
PASS
[AGD] section 2.10 indicates logs are sent to the syslog server in real-time.

33 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.

Findings
PASS
[ST] section 6.1.2 only describes overwriting the oldest record when space is exhausted and transmitting logs in real time. The evaluator did not identify any configurations or parameters indicating either behavior is configurable while examining [AGD] and [ADMIN], so this EA is considered satisfied.

3.1.3.3 Tests

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

a) Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator’s choice designed to generate audit data to be transferred to the audit

server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing. The evaluator shall verify that the TOE is capable of transferring audit data to an external audit server automatically without administrator intervention.

High-Level Test Description
Verify traffic to the audit server is not sent in plaintext and identify particular software (name, version) of the audit server used during testing:
Findings
PASS – The evaluator confirmed that audit data is not transferred in the clear; and that they are successfully received by the audit server; and once remote audit logging is enabled and logging server is successfully configured, no manual tasks were needed to transfer audit data to external audit server.

b) Test 2: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behaviour defined in FAU_STG_EXT.1.3. Depending on the configuration this means that the evaluator has to check the content of the audit data when the audit data is just filled to the maximum and then verifies that

1) The audit data remains unchanged with every new auditable event that should be tracked but that the audit data is recorded again after the local storage for audit data is cleared (for the option 'drop new audit data' in FAU_STG_EXT.1.3).

2) The existing audit data is overwritten with every new auditable event that should be tracked according to the specified rule (for the option 'overwrite previous audit records' in FAU_STG_EXT.1.3)

3) The TOE behaves as specified (for the option 'other action' in FAU_STG_EXT.1.3).

High-Level Test Description
Generate audit records to cause the log files to exceed 1GB in size. Verify the TOE performs log rotation and deletes the oldest log file.
Findings
PASS – The evaluator confirmed the TOE performs log rotation when the log exceeds 1GB and the TOE deletes the oldest log file when rotation would result in more than 5 files.

c) Test 3: If the TOE complies with FAU_STG_EXT.2/LocSpace the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2/LocSpace are correct when performing the tests for FAU_STG_EXT.1.3

Test Not Applicable	The TOE does not claim FAU_STG_EXT.2/LocSpace.

d) Test 4: For distributed TOEs, Test 1 defined above should be applicable to all TOE components that forward audit data to an external audit server. For the local storage according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3 the Test 2 specified above shall be applied to all TOE components that store audit data locally. For all TOE

components that store audit data locally and comply with FAU_STG_EXT.2/LocSpace Test 3 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

Test Not Applicable	The TOE is not a distributed TOE.

3.2 Cryptographic Support (FCS)

3.2.1 NIAP Policy 5

35 To demonstrate that all cryptographic requirements are satisfied, the Assurance Activity Report must clearly indicate all SFRs for which a CAVP certificate is claimed and include, at a minimum, the cryptographic operation, the NIST standard, the SFR supported, the CAVP algorithm list name (e.g. AES, KAS, CVL, etc.) and the CAVP Certificate number.

SFR	Cryptographic Operation	NIST Standard	CAVP Certificate (Algorithm)
FCS_CKM.1	Asymmetric Key Generation: RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3;	FIPS PUB 186-4	A1949 (RSA)
FCS_CKM.1	Asymmetric Key Generation: ECC schemes using “NIST curves” [P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4;	FIPS PUB 186-4	A1949 (ECDSA)
FCS_CKM.1	Asymmetric Key Generation FFC Schemes using ‘safe-prime’ groups that meet the following: “NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and [RFC 3526]	NIST SP 800-56A Revision 3	N/A
FCS_CKM.2	RSA-based key establishment schemes that meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447, “Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1”;	N/A	N/A
FCS_CKM.2	Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;	NIST SP 800-56A Revision 3	A1949 (KAS-ECC Component)

SFR	Cryptographic Operation	NIST Standard	CAVP Certificate (Algorithm)
FCS_CKM.2	FFC Schemes using “safe-prime” groups that meet the following: ‘NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and [groups listed in RFC 3526]	NIST SP 800-56A Revision 3	N/A
FCS_COP.1/ DataEncryption	encryption/decryption in accordance with a specified cryptographic algorithm AES used in [CBC, CTR, GCM] mode and cryptographic key sizes [128 bits, 256 bits]	FIPS PUB 197 NIST SP 800-38A NIST SP 800-38D	A1949 (AES)
FCS_COP.1/ SigGen	cryptographic signature services (generation and verification): RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [2048, 3072],	FIPS PUB 186-4	A1949 (RSA)
FCS_COP.1/ SigGen	cryptographic signature services (generation and verification): Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256,384 and 521 bits]	FIPS PUB 186-4	A1949 (ECDSA)
FCS_COP.1/ Hash	cryptographic hashing services in accordance with a specified cryptographic algorithm [SHA-1, SHA-256, SHA-384, SHA-512]	FIPS PUB 180-2	A1949 (SHA-1, 256, 384, 512)
FCS_COP.1/ KeyedHash	keyed-hash message authentication in accordance with a specified cryptographic algorithm [HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512]	FIPS PUB 196-1	A1949 (SHA-1, 256, 512)
FCS_RBG_EXT.1	random bit generation services using [CTR_DRBG (AES)]	NIST SP 800-90A	A1949 (CTR)

3.2.2 FCS_CKM.1 Cryptographic Key Generation

3.2.2.1 TSS

36 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.

Findings
PASS
[ST] section 6.2.1 identifies the RSA, ECC, and Diffie-Hellman key sizes supported by the TOE. Diffie-Hellman key sizes are implicitly specified by the specified groups. The section specifies the usage for each asymmetric key generation scheme.

3.2.2.2 Guidance Documentation

37 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.

Findings
PASS
[AGD] section 2.6 describes generating the SSH hostkey and the restrictions (i.e., RSA) that must be applied. [ADMIN] section 'Regenerate public keys' identifies valid key sizes as 2048 and 3072. [AGD] section 2.10.2 describes generating keys for CSRs. [AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of ephemeral key generation is needed.

3.2.2.3 Tests

38 Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. Generation of long-term cryptographic keys (i.e. keys that are not ephemeral keys/session keys) might be performed automatically (e.g. during initial start-up). Testing of key generation must cover not only administrator invoked key generation but also automated key generation (if supported).

Key Generation for FIPS PUB 186-4 RSA Schemes

39 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*.

40 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

41 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.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements RSA key generation.

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

42 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

43 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.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements ECDSA key generation.

Key Generation for Finite-Field Cryptography (FFC)

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

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

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

46 and two ways to generate the cryptographic group generator g:

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

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

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

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

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

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

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

51 for each FFC parameter set and key pair.

Test Not Applicable	The [ST] does not select FFC Schemes that meet FIPS PUB 186-4.

NIAP TD0580

FFC Schemes using "safe-prime"

NIAP TD0580

52 Testing for FFC Schemes using safe-prime groups is done as part of testing in CKM.2.1.

Note	Testing for FFC Schemes using "safe-prime" groups is done as part of testing in FCS_CKM.2.1.

3.2.3 FCS_CKM.2 Cryptographic Key Establishment

3.2.3.1 TSS

53 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. It is sufficient to provide the scheme, SFR, and service in the TSS.

NIAP TD0580

54 **Removed:** ~~If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall claim the TOE meets RFC 3526 Section 3.~~

55 The intent of this activity is to be able to identify the scheme being used by each service. This would mean, for example, one way to document scheme usage could be:

Scheme	SFR	Service
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
~~Diffie-Hellman (Group 14)~~ **Removed per TD0580**	~~FCS_SSHC_EXT.1~~ **Removed per TD0580**	~~Backup Server~~ **Removed per TD0580**
ECDH	FCS_IPSEC_EXT.1	Authentication Server

56 The information provided in the example above does not necessarily have to be included as a table but can be presented in other ways as long as the necessary data is available.

Findings
PASS
[ST] section 6.2.2 lists RSA, ECC, and Diffie-Hellman key establishment schemes. The schemes are consistent with the keys generated for FCS_CKM.1. The usage of the key establishment scheme is associated with SSH or TLS. FCS_SSHC_EXT.1 is the only SSH claim, and it is only user for administration. FCS_TLSC_EXT.1/2 are the only TLS claims, and it is only used for the Audit Server.

3.2.3.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of ephemeral key generation/key establishment is needed.

3.2.3.3 Tests

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

59 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

60 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.

61 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.

62 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.

63 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.

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

Validity Test

65 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.

66 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).

67 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.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements ECC SP 800-56A key agreement/establishment schemes.

RSA-based key establishment schemes

68 The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 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 RSAES-PKCS1-v1_5.

High-Level Test Description
Verify the TOE correctly implements RSAES-PKCS1-v1_5 by ensuring it can successfully negotiate with a known good implementation.
Findings
PASS – The evaluator confirmed the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 key establishment by successfully connecting to a known-good implementation as part of testing for FCS_TLSC_EXT.1.1. The "known-good" implementations used in these tests was OpenSSL 1.0.2g-LS. TLS_RSA_WITH_AES_128_CBC_SHA, TLS_RSA_WITH_AES_256_CBC_SHA, TLS_RSA_WITH_AES_128_CBC_SHA256, and TLS_RSA_WITH_AES_256_CBC_ SHA256 were successfully negotiated.

NIAP TD0580 Removed:

~~Diffie-Hellman Group 14~~

69 ~~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.~~

FFC Schemes using "safe-prime" groups

70 The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups 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 safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.

High-Level Test Description
Verify the TOE correctly implements FFC schemes using "safe-primes" by ensuring it can successfully negotiate with a known good implementation.
Findings
PASS – The TOE uses safe-prime groups for FTP_TRP.1/Admin SSHS connections and FTP_ITC.1 TLSC connections. The evaluator confirmed the correctness of the TSF's implementation of SSHS safe-prime groups by successfully connecting to a known-good implementation as part of testing for FCS_SSHS_EXT.1.7. The known-good implementation used in these tests was OpenSSH 8.8p1 which was compiled statically with

OpenSSL 1.1.1m. diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, and diffie-hellman-group18-sha512 were successfully negotiated.

The evaluator confirmed the correctness of the TSF's implementation of TLSC safe-prime groups by successfully connecting to a known-good implementation as part of testing for FCS_TLSC_EXT.1.1. The "known-good" implementation used in these tests was OpenSSL 1.0.2g-LS. Diffie-Hellman Group 14 was successfully negotiated.

3.2.4 FCS_CKM.4 Cryptographic Key Destruction

3.2.4.1 TSS

71 The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted for[1]). 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.

Findings

PASS

[ST] section 6.5.1, Table 15 identifies the cryptographic keys stored by the TOE. This description includes the origin, storage locations, key destruction situations, and destruction method.

The evaluator confirmed the keys and storage locations are consistent with the SSH and TLS algorithm selections and the way each protocol uses cryptographic keys. The evaluator did not identify any other functions that involve in scope keys.

72 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).

Findings

PASS

[ST] section 6.5.1, Table 15 indicates keys are destroyed though an overwrite with zeros. For keys stored in NVRAM, a proprietary API is invoked to perform the overwrite.

73 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

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

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.

Findings
PASS
FCS_CKM.4 does not select ‘destruction of reference’.
[ST] section 6.5.1, table 15 identifies how key destruction is invoked. For keys stored in NVRAM, a proprietary API is invoked to perform the overwrite.

74 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.

Findings
PASS
[ST] section 6.5.1, table 15 indicates all keys are stored in plaintext form.

75 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.

Findings
PASS
[ST] section 6 does not identify any configuration or circumstances that may not conform to the key destruction requirement.

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

Findings
PASS
[ST] section 5.3.2, FCS_CKM.4.1 does not select “a…value that does not contain any CSP” for volatile or non-volatile key destruction methods.

3.2.4.2 Guidance Documentation

77 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.

78 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[2] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings
PASS
[ST] and [AGD] do not describe the use of the TRIM command or garbage collection.
[AGD] does not identify any scenarios where key destruction may be delayed by the physical layer.

3.2.4.3 Tests

79 None

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

3.2.5.1 TSS

80 The evaluator shall examine the TSS to ensure it identifies the key size(s) and mode(s) supported by the TOE for data encryption/decryption.

Findings
PASS
[ST] section 6.2.4 identifies 128 and 256 bits a the key sizes supported by the TOE. CBC, CTR, an GCM are identified as the AES modes supported by the TOE.

3.2.5.2 Guidance Documentation

81 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected mode(s) and key size(s) defined in the Security Target supported by the TOE for data encryption/decryption.

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of encryption/decryption algorithms, modes, or key sizes are needed.

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

3.2.5.3 Tests

AES-CBC Known Answer Tests

82 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.

83 **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.

84 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.

85 **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.

86 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.

87 **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].

88 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.

89 **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].

90 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

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

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

AES-CBC Monte Carlo Tests

93 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]
```

94 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.

95 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

96 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.

97 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.

98 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.

99 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

100 The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Since the Counter Mode does not specify the counter that is used, it is not possible to implement an automated test for this mode. The generation and management of the counter is tested through FCS_SSH*_EXT.1.4. If CBC and/or GCM are selected in FCS_COP.1/DataEncryption, the test activities for those modes sufficiently demonstrate the correctness of the AES algorithm. If CTR is the only selection in FCS_COP.1/DataEncryption, the AES-CBC Known Answer Test, AES-GCM Known Answer Test, or the following test shall be performed (all of these tests demonstrate the correctness of the AES algorithm):

101 There are four Known Answer Tests (KATs) described below to test a basic AES encryption operation (AES-ECB mode). 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.

102 KAT-1 To test the encrypt functionality, the evaluator shall supply a set of 5 plaintext values for each selected keysize and obtain the ciphertext value that results from encryption of the given plaintext using a key value of all zeros.

103 KAT-2 To test the encrypt functionality, the evaluator shall supply a set of 5 key values for each selected keysize and obtain the ciphertext value that results from encryption of an all zeros plaintext using the given key value.

104 KAT-3 To test the encrypt functionality, the evaluator shall supply a set of key values for each selected keysize as described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values. A set of 128 128-bit keys, a set of 192 192-bit keys, and/or a set of 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].

105 KAT-4 To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the ciphertext values that result from encryption of the given plaintext using each selected keysize with a key value of all zeros (e.g. 256 ciphertext values will be generated if 128 bits and 256 bits are selected and 384 ciphertext values will be generated if all keysizes are selected). 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]

AES-CTR Multi-Block Message Test

106 The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10 (test shall be performed using AES-ECB mode). For each i the evaluator shall choose a key 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 using a known good implementation. The evaluator shall perform this test using each selected keysize.

AES-CTR Monte-Carlo Test

107 The evaluator shall test the encrypt functionality using 100 plaintext/key pairs. 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-ECB-Encrypt(Key, PT) PT = CT[i]

108 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. The evaluator shall perform this test using each selected keysize.

109 There is no need to test the decryption engine.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements AES.

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

3.2.6.1 TSS

110 The evaluator shall examine the TSS to determine that it specifies the cryptographic algorithm and key size supported by the TOE for signature services.

Findings
PASS
[ST] section 5.3.2, FCS_COP.1.1/SigGen identifies RSA and ECDSA as the signature algorithms.
[ST] section 6.2.5 identifies RSA with a 2048 or 3072-bit key and ECDSA with P-256, P-384, and P-521 as the signature algorithms/key sizes.

3.2.6.2 Guidance Documentation

111 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected cryptographic algorithm and key size defined in the Security Target supported by the TOE for signature services.

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces TLS configuration. This means that no configuration of signature verification algorithms or key sizes are needed for the TLS client to authenticate the remote server.
[AGD] section 2.10.2 describes generating the CSRs with compliant algorithms and key sizes (for the TLS client to authenticate itself to the remote server).
[AGD] section 2.6 describes generating the SSH hostkey which is used to generate signatures.
[AGD] section 2.6 describes configuring complaint SSH userkeys which are used for signature verification.

3.2.6.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

112 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

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

RSA Signature Algorithm Tests

Signature Generation Test

114 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.

115 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

116 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.

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

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements RSA and ECDSA signature generation and verification.

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

3.2.7.1 TSS

118 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.

Findings
PASS
[ST] section 6.2.6 associates the hash function with TLS, SSH, password hashing, Kernel digital signature verification, and update verification.

3.2.7.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of hash sizes is needed.

3.2.7.3 Tests

120 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.

121 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

122 The evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m

bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode

123 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

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

Selected Long Messages Test - Byte-oriented Mode

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

Pseudorandomly Generated Messages Test

126 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.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements Hashing.

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

3.2.8.1 TSS

127 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.

Findings
PASS

[ST] section 6.2.7, Table 14 identifies the key size, hash function, block size, and digest size for each HMAC function. The claimed HMAC functions are consistent with the selections in the SFR.

3.2.8.2 Guidance Documentation

128 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the values used by the HMAC function: key length, hash function used, block size, and output MAC length used defined in the Security Target supported by the TOE for keyed hash function.

Findings

PASS

[AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of HMAC parameters is needed.

3.2.8.3 Tests

129 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.

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements HMAC algorithms.

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

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

3.2.9.1 TSS

131 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.

Findings

PASS

[ST] section 6.2.9 specifies the TOE uses a CTR_DRBG. RDRAND is identified as the entropy source that provides 256-bits of assumed entropy.

3.2.9.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces TLS and SSH configurations. This means that no configuration of RBG functionality is needed.

3.2.9.3 Tests

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

134 If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 – 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. “generate one block of random bits” means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

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

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

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

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

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

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

Note	[ST] Table 4 specifies the CAVP certificate demonstrating the TOE correctly implements Deterministic Random Bit Generation.

3.3 Identification and Authentication (FIA)

3.3.1 FIA_AFL.1 Authentication Failure Management

3.3.1.1 TSS

137 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.

Findings
PASS
[ST] section 6.3.5 indicates the TOE tracks failed authentication attempts using a counter associated with each account. When a user account has sequentially failed authentication the configured number of times, the account will be locked for a Security Administrator defined time period. The TOE automatically unlocks the account once the time period has elapsed.

138 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).

Findings
PASS
[ST] section 6.3.5 indicates that the local console does not lock out users based on failed authentication attempts. This ensures administrative access is always possible.

3.3.1.2 Guidance Documentation

139 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.

Findings
PASS
[AGD] section 2.6 describes using the password-attributes command to configure failed authentication lockouts. SSH is the only secure protocol employed.

140 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.

Findings
PASS
[AGD] section 2.6 requires the “console-exempt” parameter to be specified, ensuring access is always maintained.

3.3.1.3 Tests

141 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.

High-Level Test Description
Configure the TOE to block password authentication attempts after three failed attempts. Perform three failed authentication attempts so that unsuccessful login attempts limit is “met”. Verify audit messages are generated. Attempt to login the fourth time with the correct password so that unsuccessful login attempts limit is “exceeded”, and verify access is denied. Verify audit messages are generated.
Findings
PASS – The evaluator confirmed the TOE prevents authentication with valid credentials once the configured failed authentication threshold has been met.

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.

High-Level Test Description
With the account locked as a result of FIA_AFL.1 Test 1, attempt to authenticate with correct credentials prior to the lockout time period elapsing. Verify access is denied. Once the lockout time period has elapsed, attempt to authentication with correct credentials. Verify access is allowed.
Findings

PASS – The evaluator confirmed the TOE correctly enforces the time period for allowing authentication attempts, denying authentication prior to the time period elapsing and permitting authentication after the time period has elapsed.

3.3.2 FIA_PMG_EXT.1 Password Management

3.3.2.1 TSS

142 The evaluator shall examine the TSS to determine that it contains the lists of the supported special character(s) and minimum and maximum number of charters supported for administrator passwords.

Findings
PASS
[ST] section 6.3.1 indicates identifies “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)” as the supported special characters. This list is consistent with the SFR. The minimum password length can be configured to be a value from 9 to 32 inclusive.

3.3.2.2 Guidance Documentation

143 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.

Findings
PASS
[AGD] section 2.8 identifies the characters that may be used in passwords, provides guidance on the composition of strong passwords, and identifies valid minimum password lengths. [ADMIN] section ‘Password strength’ provides instructions on setting the minimum password length.

3.3.2.3 Tests

144 The evaluator shall perform the following tests.

a) Test 1: The evaluator shall compose passwords that 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.

High-Level Test Description
Configure the TOE to enforce a password minimum length of 9, then set a password that meets the minimum password length exactly. Verify the password is accepted and can be used to login. Configure a password using all claimed characters. Verify the password is accepted and can be used to login.

Findings
PASS – The evaluator confirmed the passwords meeting the minimum length requirement and passwords containing all claimed characters can be configured and used on the TOE.

b) Test 2: The evaluator shall compose passwords that do not meet the requirements in some way. For each password, the evaluator shall verify that the TOE does not support the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the evaluator shall ensure that the TOE enforces the allowed characters and the minimum length listed in the requirement and justify the subset of those characters chosen for testing.

High-Level Test Description
Configure the TOE to enforce a password minimum length of 9, then attempt to set a password that is shorter than the minimum password length. Verify the password change is rejected.
Findings
PASS – The evaluator confirmed that the TOE rejected a password that is shorter than the minimum password length.

3.3.3 FIA_UIA_EXT.1 User Identification and Authentication

3.3.3.1 TSS

145 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”.

Findings
PASS
[ST] sections 6.3.2 and 6.3.3 identify SSH and the local console as the logon methods. [ST] section 6.3.3 identifies the login process, credentials supported, and success criteria for password based and SSH public key based authentication.

146 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.

Findings
PASS
[ST] section 6.3.2 indicates the warning banner can be viewed prior to user identification and authentication. This action applies to SSH and the local console.

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

Findings
PASS
[ST] The TOE is not a distributed TOE.

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

Findings
PASS
[ST] The TOE is not a distributed TOE.

3.3.3.2 Guidance Documentation

149 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.

Findings
PASS
[AGD] section 2.8 describes changing default passwords.
[AGD] section 2.6 and [ADMIN] section ‘username sshkey’ describe configuring SSH pubkey authentication.

3.3.3.3 Tests

150 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.

High-Level Test Description
For each method of administration (serial and SSH) verify that attempting to log into the TOE with correct I&A credentials (username and password or SSH public key) allows access and invalid I&A credentials (invalid username, invalid password, or invalid SSH public key) denies access.
Findings

PASS – The evaluator confirmed that the TOE allows access when correct I&A information is provided and denies access when incorrect I&A information is provided.

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.

High-Level Test Description
Verify no services other than the warning banner are available to a remote entity.
Findings
PASS – The only method of remote administration is SSH. FIA_UIA_EXT.1 and FCS_SSHS_EXT.1.2 show authentication is required, and the SSH protocol enforces the authentication flow so no service other than the warning banner can be offered prior to authentication.

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.

High-Level Test Description
In local console, examine and show that the device does not have any services configured prior to I&A other than a TOE banner by entering common shell key combinations and strings to escape and/or run commands. Verify the user is unable to run any commands or services other than the warning banner.
Findings
PASS – The evaluator confirmed the device does not have any services available to local administrator prior to I&A aside from a TOE banner.

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

Test Not Applicable	The TOE is not a distributed TOE.

3.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

151 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.

3.3.5 FIA_UAU.7 Protected Authentication Feedback

3.3.5.1 TSS

152 None

3.3.5.2 Guidance Documentation

153 The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps to ensure authentication data is not revealed while entering for each local login allowed.

Findings
PASS
[AGD] does not include any configuration steps to ensure authentication data is not revealed at the local console. While performing testing, the evaluator confirmed that no configuration is necessary.

3.3.5.3 Tests

154 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.

High-Level Test Description
Login to the local console and verify at most obscured feedback of the password entry is provided.
Findings
PASS – The evaluator confirmed the TOE does not output any feedback when entering a password at the local console.

3.4 Security management (FMT)

3.4.1 General requirements for distributed TOEs

3.4.1.1 TSS

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

Findings
PASS
The TOE is not a distributed TOE.

3.4.1.2 Guidance Documentation

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

Findings

PASS
The TOE is not a distributed TOE.

3.4.1.3 Tests

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

High-Level Test Description
The TOE is not a distributed TOE.
Findings
PASS – The TOE is not a distributed TOE.

3.4.2 FMT_MOF.1/ManualUpdate

3.4.2.1 TSS

158 For distributed TOEs see [ND-SD] chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings
PASS
The TOE is not a distributed TOE.

3.4.2.2 Guidance Documentation

159 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).

Findings
PASS
[AGD] section 2.4 describes performing manual updates.

160 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).

Findings
PASS
The TOE is not a distributed TOE.

3.4.2.3 Tests

161 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.

Note	The TOE only supports the Security Administrator role and does not allow any administrative actions prior to authentication as a Security Administrator, so this is tested as part of FIA_UIA_EXT.1 Test 2 and 3.

162 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.

Note	This is covered by FPT_TUD_EXT.1 Test 1.

3.4.3 FMT_MTD.1/CoreData Management of TSF Data

3.4.3.1 TSS

163 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.

Findings
PASS
[ST] section 6.4.3 states, “Management of the trust store is an administrative function, which is restricted to authenticated administrators.” The evaluator confirmed that this is consistent with the FIA_UIA_EXT.1 claims of services available prior to authentication (which do not allow the manipulation of TSF data).

164 If 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.

Findings
PASS
[ST] section 6.4.3 states, “Management of the trust store is an administrative function, which is restricted to authenticated administrators.” The evaluator confirmed that this is consistent with the FIA_UIA_EXT.1 claims of services available prior to authentication (which do not allow the manipulation of TSF data).

3.4.3.2 Guidance Documentation

165 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.

Findings
PASS
[AGD] sections 2.6, 2.7, 2.8, 2.9, 2.10, 2.11, and 2.12 describe the user of the TSF-data manipulating functions. [ST] section 6.4.4 indicates all roles are consider Security Administrators, so no restriction of functions is necessary.

166 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.

Findings
PASS
[AGD] section 2.10.2 and [ADMIN] section ‘X.509v3 certificates’ > ‘Manage CA certificates’ provide guidance for the secure management of the trust store.

3.4.3.3 Tests

167 No separate testing for FMT_MTD.1/CoreData is required unless one of the management functions has not already been exercised under any other SFR.

3.4.4 FMT_SMF.1 Specification of Management Functions

168 The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1/Services, and FMT_MOF.1/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

3.4.4.1 TSS (containing also requirements on Guidance Documentation and Tests)

169 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).

Findings
PASS

[ST] section 6.4.6 lists all of the management functions and indicates they are available via the CLI which is accessible via the console and SSH.

[AGD] section 2.6 identifies the CLI (accessible via the local console and SSH) as the only administrative interface.

While performing testing, the evaluator did not identify any additional administrative interfaces.

170 The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Findings

PASS

[ST] section 6.3.2 defines the console as “Directly connecting to the TOE appliance (serial over RJ45).” Serial is inherently local, so no additional warnings are necessary.

[AGD] section 2.6 identifies describes the local serial console and provides a warning for the administrator to ensure the serial console is a local interface.

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

Findings

PASS

The TOE is not a distributed TOE.

3.4.4.2 Guidance Documentation

172 See [ND-SD] section 2.4.4.1.

3.4.4.3 Tests

173 The evaluator tests management functions as part of testing the SFRs identified in [ND-SD] 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.

3.4.5 FMT_SMR.2 Restrictions on security roles

3.4.5.1 TSS

174 The evaluator shall examine the TSS to determine that it details the TOE supported roles and any restrictions of the roles involving administration of the TOE.

Findings

PASS

[ST] section 6.4.4 indicates the product roles of Network Operator, Network Administrator, Security Administrator, and System Administrators are considers Security Administrators for the evaluation. No restrictions on the roles that may be used to administer the TOE are stated.

3.4.5.2 Guidance Documentation

175 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.

Findings
PASS
[AGD] section 2.6 describes configuring the TOE for local and remote administration. No particular configuration needs to be performed on the client for remote administration.

3.4.5.3 Tests

176 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.

Note	There are no explicit test activities and therefore none are recorded here. Both the remote SSH CLI and local console CLI are tested throughout this test plan.

3.5 Protection of the TSF (FPT)

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

3.5.1.1 TSS

177 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.

Findings
PASS
[ST] section 6.5.1 identifies the symmetric and private keys stored by the TOE. All keys are stored plaintext. The keys are protected, because the TOE does not provide an interface specifically for viewing the keys.

3.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

3.5.2.1 TSS

178 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.

Findings

PASS

[ST] section 6.5.2 identifies administrator passwords subject to this requirement. The plaintext passwords are obscured using SHA-256 and there is not an interface to view the passwords. SSH public-key based and password based authentication are the only identified methods of authentication, so administrator passwords are the only data expected to be subject to this requirement.

3.5.3 FPT_TST_EXT.1 TSF testing

3.5.3.1 TSS

179 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.

Findings

PASS

[ST] section 6.5.3 identifies the self-tests as FIPS 140-2 tests, Kernel and file integrity tests, and hardware self-tests. The FIPS 140-2 tests are explained to be known answer, integrity, and conditional self-tests (which are well-defined based on FIPS 140-2) along with the specific algorithms and operations being tested. The Kernel and file integrity tests use digital signature verification and hash verification respectively. The evaluator agrees that the combination of self-tests performed are sufficient to demonstrate that the TSF is operating correctly.

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

Findings

PASS

The TOE is not a distributed TOE.

3.5.3.2 Guidance Documentation

181 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.

Findings
PASS
[AGD] section 2.3 identifies the self-tests, possible errors, and administrative actions to be taken in response to errors.

182 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.

Findings
PASS
The TOE is not a distributed TOE.

3.5.3.3 Tests

183 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.

184 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.

185 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.

186 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.

High-Level Test Description
Reboot the TOE and observe BIOS self-test in the serial CLI console, then verify all required self-tests were performed during startup.
Findings
PASS – The evaluator confirmed that the required self tests are performed during startup.

3.5.4 FPT_TUD_EXT.1 Trusted Update

3.5.4.1 TSS

187 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.

Findings
PASS
[ST] section 6.5.4 indicates the “show version” and “show boot detail” commands can be used to view the version running on the TOE. The TOE supports delayed activation and “show boot detail” shows the active version as well as the most recently installed version (i.e., standby image). [ST] section 6.5.4 also describes how and when an inactive version becomes active.

188 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.

Findings
PASS
[ST] section 6.5.4 identifies the CLI as the only software update mechanism. The image can be copied to the TOE in a variety of methods, but the verification and installation methods remain the same regardless of how the image was transferred to the TOE. The image is verified using a hash or signature with the ‘image secure-install” command.

189 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.

Findings
PASS
[ST] section 5.3.5, FTP_TUD_EXT.1.2 does not select ‘support automatic checking for updates’ or ‘support automatic updates’.

190 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.

Findings
PASS
The TOE is not a distributed TOE.

191 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.

Findings
PASS
[AGD] section 2.4 describes logging into the Dell Support website to obtain software updates. The 'image secure-install' command requires the administrator to manually specify the hash, showing it is not a fully automated process.

3.5.4.2 Guidance Documentation

192 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.

Findings
PASS
[AGD] section 2.1 describes the ‘show version’ and ‘show boot detail’ commands to show the currently active version and inactive version.

193 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.

Findings
PASS
[AGD] section 2.4 describes verifying updates to the TOE using a digital signature or hash.

194 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.

Findings

PASS

[AGD] section 2.4 describes how published hashes are bundled with the software updates.

195 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.

Findings

PASS

The TOE is not a distributed TOE.

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

Findings

PASS

The TOE is not a distributed TOE.

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

Findings

PASS

The TOE is not a distributed TOE.

3.5.4.3 Tests

198 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.

High-Level Test Description
For each method of update verification; show the current version of the TOE, install a legitimate update of the TOE, and verify version is consistent with the newly installed version.
Findings
PASS – The evaluator confirmed the TOE can successfully install updates when a valid hash or signature is provided.

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). 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.

High-Level Test Description
Attempt to install a modified update with a valid signature, an update without a signature, and an update with an invalid signature. Verify each update attempt fails.
Findings
PASS – The evaluator confirmed the TOE will not install an update when verification using a digital signature fails.

c) Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from

outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted. If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the user to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

2) The evaluator uses a legitimate update and tries to perform verification of the hash value without providing the published hash value to the TOE. The evaluator confirms that this attempt fails. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax check of the command prevents the execution of the command without providing a hash value. In that case the mechanism that prevents the execution of this check shall be tested accordingly, e.g. that the syntax check rejects the command without providing a hash value, and the rejection of the attempt is regarded as sufficient verification of the correct behaviour of the TOE in failing to verify the hash. The evaluator then attempts to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE, the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

3) If the TOE allows delayed activation of updates, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

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

High-Level Test Description
Attempt to install an update with an invalid hash and a missing hash. Verify each update attempt fails.
Findings
PASS – The evaluator confirmed the TOE does not instal an update when hash validation fails.

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

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

3.5.5 FPT_STM_EXT.1 Reliable Time Stamps

3.5.5.1 TSS

202 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.

Findings
PASS
[ST] section 6.5.5 identifies audit record timestamps, session timeouts, and certificate validation as the TOE's use of time. The TOE maintains an internal clock to maintain time. A battery maintains the internal time to ensure it is reliable.

NIAP TD0632

203 If "obtain time from the underlying virtualization system" is selected, the evaluator shall examine the TSS to ensure that it identifies the VS interface the TOE uses to obtain time. If there is a delay between updates to the time on the VS and updating the time on the TOE, the TSS shall identify the maximum possible delay.

Findings
PASS
[ST] section 5.3.5, FPT_STM_EXT.1.2 does not select "obtain time from the underlying virtualization system."

3.5.5.2 Guidance Documentation

204 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.

Findings
PASS
[AGD] section 2.9 describes how to set the time.

NIAP TD0632

205 If the TOE supports obtaining time from the underlying VS, the evaluator shall verify the Guidance Documentation specifies any configuration steps necessary. If no configuration is necessary, no statement is necessary in the Guidance Documentation. If there is a delay between updates to the time on the VS and updating the time on the TOE, the evaluator shall ensure the Guidance Documentation informs the administrator of the maximum possible delay.

Findings
PASS
The TOE does not obtain time from the underlying VS.

3.5.5.3 Tests

206 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.

High-Level Test Description
Change the date/time in various combinations of forward/backward including all elements (day, month, year, hour, minute, second, etc.) and verify that time was changed accordingly.
Findings
PASS – The evaluator confirmed the Security Administrator is able to manually set the time on the TOE by following the guidance.

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

Test Not Applicable	The ST does not claim NTP.

NIAP TD0632

c) Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay between the setting the time on the VS and when the time is reflected on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

Test Not Applicable	The TOE does not obtain time from the underlying VS.

207 If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different

parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

Test Not Applicable	The audit component of the TOE does not consist of several parts with independent time information.

3.6 TOE Access (FTA)

3.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.6.1.1 TSS

208 The evaluator shall examine the TSS to determine that it details whether local administrative session locking or termination is supported and the related inactivity time period settings.

Findings
PASS
[ST] section 6.6.1 indicates the TOE performs local administrative session termination and that the inactivity period can be configured from 1 to 65535 seconds.

3.6.1.2 Guidance Documentation

209 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.

Findings
PASS
[AGD] section 2.6 describes configuring local administrative session termination and the associated inactivity period.

3.6.1.3 Tests

210 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.

High-Level Test Description
Configure several inactivity timeout values. Verify the TOE terminates local console sessions when the inactivity period has elapsed.
Findings

PASS – The evaluator confirmed that the session is terminated after the configured time period.

3.6.2 FTA_SSL.3 TSF-initiated Termination

3.6.2.1 TSS

211 The evaluator shall examine the TSS to determine that it details the administrative remote session termination and the related inactivity time period.

Findings
PASS
[ST] section 6.6.2 indicates the TOE terminates remote administrative sessions after an inactivity period. of 1 to 65535 seconds

3.6.2.2 Guidance Documentation

212 The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Findings
PASS
[AGD] section 2.6 describes configuring remote administrative session termination and the associated inactivity period.

3.6.2.3 Tests

213 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.

High-Level Test Description
Configure several inactivity timeout values. Verify the TOE terminates remote SSH sessions when the inactivity period has elapsed.
Findings
PASS – The evaluator confirmed that the session is terminated after the configured time period.

3.6.3 FTA_SSL.4 User-initiated Termination

3.6.3.1 TSS

214 The evaluator shall examine the TSS to determine that it details how the local and remote administrative sessions are terminated.

Findings

PASS
[ST] section 6.6.3 indicates local and remote sessions are terminated with the ‘exit’ command.

3.6.3.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.13 indicates administrators can terminate their own sessions by using the ‘exit’ command.

3.6.3.3 Tests

216 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.

High-Level Test Description
Log into the serial console and immediately log out. Verify that the session has been terminated.
Findings
PASS – The evaluator confirmed the local session was terminated by the user.

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.

High-Level Test Description
Log into the SSH CLI interface and immediately log out. Verify the session has been terminated.
Findings
PASS – The evaluator confirmed the remote session was terminated by the user.

3.6.4 FTA_TAB.1 Default TOE Access Banners

3.6.4.1 TSS

217 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).

Findings
PASS
[ST] section 6.4.6 lists all of the management functions and indicates they are available via the CLI which is accessible via the console and SSH. [ST] section 6.6.4 indicates the banner is displayed prior to authenticating to the CLI.

3.6.4.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.6 and [ADMIN] section 'Login banner' describe how to configure the banner message.

3.6.4.3 Tests

219 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.

High-Level Test Description
Change the banner to any string. Prior to I&A of both local console and SSH, verify that the banner was modified and is presented.
Findings
PASS – The evaluator confirmed that the banner was successfully changed and displayed in both local console and SSH prior to I&A.

3.7 Trusted path/channels (FTP)

3.7.1 FTP_ITC.1 Inter-TSF trusted channel

3.7.1.1 TSS

220 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.

Findings
PASS

[ST] section 6.7.1 identifies the TOE as a TLS client to the Audit server. This is consistent with the selections in the SFR and the inclusion of FCS_TLSC_EXT.1 and FCS_TLSC_EXT.2.

3.7.1.2 Guidance Documentation

221 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.

Findings

PASS

[AGD] section 2.10.1 and [ADMIN] section 'System logging over TLS' describe how to configure a TLS trusted channel with an audit server. These descriptions include configuring mutual authentication for the channel.

[AGD] section 2.10.1 indicates the TOE automatically attempts to reestablish the connection to the syslog server if the connection is broken. Based on this the evaluator determined no additional recovery instructions are necessary.

3.7.1.3 Tests

222 The developer 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.

223 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.

Note The TOE maintains trusted channel to the remote audit server, which is set up as per the evaluated configuration. It is constantly tested throughout the evaluation.

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.

Note FCS_TLSC_EXT.1 testing shows the TOE can initiate the trusted channel to the remote audit server.

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

High-Level Test Description

Examine a packet capture performed in a testing involving data, such as username, being transferred to audit server and verify that they are not sent in plaintext.

Findings
PASS – The evaluator confirmed that no plaintext such as username “admin” is present in the packet capture.

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 shall, for each instance where the TOE acts as a client utilizing a secure communication mechanism with a distinct IT entity, physically interrupt the connection of that IT entity for the following durations: i) a duration that exceeds the TOE’s application layer timeout setting, ii) a duration shorter than the application layer timeout but of sufficient length to interrupt the network link layer.

The evaluator shall ensure that, when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.

In the case where the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

High-Level Test Description
Findings
PASS

High-Level Test Description
Properly establish a connection between the TOE and the syslog server using a mirror switch, then physically disconnect the TOE. While waiting for 5 seconds, trigger the TOE to generate Application Data packets then reconnect. Verify no TSF data is sent in plaintext and required audit messages are generated. Repeat the test for the duration of so the application layer times out.
Findings
PASS – The evaluator confirmed trusted channel data is not sent in plaintext when the channel is physically broken or when physical connectivity is restored.

224 Further assurance activities are associated with the specific protocols.

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

Test Not Applicable	The TOE is not a distributed TOE.

226 The developer 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.

3.7.2 FTP_TRP.1/Admin Trusted Path

3.7.2.1 TSS

227 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.

Findings
PASS
[ST] section 6.7.2 identifies SSH as the only method of remote administrator. This is consistent with the selections in the SFR and the inclusion of FCS_SSHS_EXT.1.

3.7.2.2 Guidance Documentation

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

Findings
PASS
[AGD] section 2.6 identifies SSH as the remote administrative protocol. SSH is an industry standard protocol, so not additional instructions are necessary.

3.7.2.3 Tests

229 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.

Note	The trusted path is the SSH Remote CLI, which is set up as per the evaluated configuration. It is constantly tested throughout the evaluation. SSH is tested in FCS_SSHS_EXT.1.

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

High-Level Test Description
Capture traffic while logging into the TOE over the trusted path. Verify the username and password are not sent in plaintext.
Findings
PASS – The evaluator confirmed SSH data messages in the trusted channel being tested are not in plaintext.

230 Further assurance activities are associated with the specific protocols.

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

Test Not Applicable	The TOE is not a distributed TOE.

4 Evaluation Activities for Optional Requirements

4.1 Cryptographic Support (FCS)

4.1.1 FCS_TLSC_EXT.2 Extended: TLS Client support for mutual authentication

4.1.1.1 TSS

FCS_TLSC_EXT.2.1

232 The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Findings
PASS
[ST] section 6.2.10 describes the use of X.509 client certificates for TLS mutual authentication. The certificate selected for use is based on the certificate configured in the Security Profile associated with the TLS connection.

4.1.1.2 Guidance Documentation

FCS_TLSC_EXT.2.1

233 If the TSS indicates that mutual authentication using X.509v3 certificates is used, the evaluator shall verify that the AGD guidance includes instructions for configuring the client-side certificates for TLS mutual authentication.

Findings
PASS
[AGD] section 2.10.1 and [ADMIN] section ‘System logging over TLS’ describe how to configure a TLS trusted channel with an audit server. These descriptions include configuring mutual authentication for the channel.

4.1.1.3 Tests

234 For all tests in this chapter the TLS server used for testing of the TOE shall be configured to require mutual authentication.

FCS_TLSC_EXT.2.1

NIAP TD0670

235 **Removed:** ~~(covered by FCS_TLSC_EXT.1.1 Test 1 and testing for FIA_X.509_EXT.*).~~

NIAP TD0670

236 Test 1: The evaluator shall establish a connection to a peer server that is configured for mutual authentication (i.e. sends a server Certificate Request (type 13) message). The evaluator observes that the TOE TLS client sends both client Certificate (type 11) and client Certificate Verify (type 15) messages during its negotiation of a TLS channel and that Application Data is sent.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to establish a TLS connection and verify that the connection is successful and the packet sniffer shows the required messages related to mutual authentication.
Findings
PASS – The evaluator confirmed that in the packet capture, the server sent Certificate Request (type 13) message and the client sent both client Certificate (type 11) and client Certificate Verify (type 15) messages during its negotiation of a TLS channel and that Application Data was sent.

NIAP TD0670

237 In addition, all other testing in FCS_TLSC_EXT.1 and FIA_X509_EXT.* must be performed as per the requirements.

5 Evaluation Activities for Selection-Based Requirements

5.1 Cryptographic Support (FCS)

5.1.1 FCS_SSHS_EXT.1 SSH Server

5.1.1.1 TSS

FCS_SSHS_EXT.1.2

NIAP TD0631

238 The evaluator shall check to ensure that the TSS contains a list of supported public key algorithms that are accepted for client authentication and that this list is consistent with signature verification algorithms selected in FCS_COP.1/SigGen (e.g., accepting EC keys requires corresponding Elliptic Curve Digital Signature algorithm claims).

Findings
PASS
[ST] section 6.2.9 identifies ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521 as the algorithms supported for client authentication. The evaluator confirmed this is consistent with the algorithms and key sizes claimed for FCS_COP.1/SigGen.

NIAP TD0631

239 The evaluator shall confirm that the TSS includes the description of how the TOE establishes a user identity when an SSH client presents a public key or X.509v3 certificate. For example, the TOE could verify that the SSH client's presented public key matches one that is stored within the SSH server's authorized_keys file.

Findings
PASS
[ST] section 6.2.9 indicates the presented username is used to with public key authentication.

NIAP TD0631

240 If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, then the evaluator shall confirm its role in the authentication process is described in the TSS.

Findings
PASS
[ST] section 6.2.9 indicates the TOE supports password-based authentication. [ST] section 6.3.3 describes how passwords are used in the SSH authentication process.

FCS_SSHS_EXT.1.3

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

Findings
PASS
[ST] section 6.2.9 indicates that large packets are detected by examining the size. If the packet is “large,” the TOE drops the packet.

FCS_SSHS_EXT.1.4

242 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.

Findings
PASS
[ST] section 6.2.9 identifies AES-CBC-128, AES-CBC-256, AES-CTR-128, AES-CTR-256, aes128-gcm@openssh.com, and aes256-gcm@openssh.com as the SSH encryption algorithms. This list matches the algorithms selected in FCS_SSHS_EXT.1.4.

FCS_SSHS_EXT.1.5

NIAP TD0631

243 The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the SSH server’s host public key algorithms supported are specified and that they are identical to those listed for this component.

Findings
PASS
[ST] section 6.2.9 identifies ssh-rsa, rsa-sha2-256, and rsa-sha2-512 as the hostkey algorithms. This list matches the algorithms selected in FCS_SSHS_EXT.1.5.

FCS_SSHS_EXT.1.6

244 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.

Findings
PASS
[ST] section 6.2.9 identifies HMAC-SHA1, HMAC-SHA2-256, and HMAC-SHA2-512 as the data integrity algorithms. This list matches the algorithms selected in FCS_SSHS_EXT.1.6 except for “implicit.” “implicit” is not an algorithm, so omitting it from the TSS is reasonable.

FCS_SSHS_EXT.1.7

245 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.

Findings
PASS
[ST] section 6.2.9 identifies diffie-hellman-group14-sha1, ecdh-sha2-nistp256, diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp384, and ecdh-sha2-nistp521 as the key exchange algorithms. This list matches the algorithms selected in FCS_SSHS_EXT.1.7.

FCS_SSHS_EXT.1.8

246 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.

Findings
PASS
[ST] section 6.2.9 claims the TOE rekeys SSH connections rekey after 1 hour or an encryption key has been used to protect 1GB of data, whichever occurs first.

5.1.1.2 Guidance Documentation

FCS_SSHS_EXT.1.4

247 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).

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces the SSH configuration. This means that no configuration of SSH is needed.

FCS_SSHS_EXT.1.5

248 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).

Findings
PASS

[AGD] section 2.7 indicates FIPS Mode enforces the SSH configuration. This means that no configuration of SSH is needed. Note: Key size is address in FCS_CKM.1; however, key size is not part of the FCS_SSHS_EXT.1.5 selection.

FCS_SSHS_EXT.1.6

249 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).

Findings

PASS

[AGD] section 2.7 indicates FIPS Mode enforces the SSH configuration. This means that no configuration of SSH is needed.

FCS_SSHS_EXT.1.7

250 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.

Findings

PASS

[AGD] section 2.7 indicates FIPS Mode enforces the SSH configuration. This means that no configuration of SSH is needed.

FCS_SSHS_EXT.1.8

251 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.

Findings

PASS

[AGD] section 2.7 state the rekey thresholds, the thresholds are not configurable, and the TOE reacts to the first threshold reached.

5.1.1.3 Tests

FCS_SSHS_EXT.1.2

NIAP TD0631

252 Test objective: The purpose of these tests is to verify server supports each claimed client authentication method.

NIAP TD0631

253 Test 1: For each supported client public-key authentication algorithm, the evaluator shall configure a remote client to present a public key corresponding to that authentication method (e.g., 2048-bit RSA key when using ssh-rsa public key). The evaluator shall establish sufficient separate SSH connections with an appropriately configured remote non-TOE SSH client to demonstrate the use of all applicable public key algorithms. It is sufficient to observe the successful completion of the SSH Authentication Protocol to satisfy the intent of this test.

High-Level Test Description
Using SSH client, log into the TOE using each of the claimed public key algorithms with a valid key and show that the communication is successful.
Findings
PASS – The evaluator confirmed each claimed public key algorithm can be used to authenticate to the TOE.

NIAP TD0631

254 Test 2: The evaluator shall choose one client public key authentication algorithm supported by the TOE. The evaluator shall generate a new client key pair for that supported algorithm without configuring the TOE to recognize the associated 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.

High-Level Test Description
Generate a SSH keypair whose algorithm matches the keypair configured on the TOE. Attempt to authenticate using the newly created keypair and verify the connection is rejected.
Findings
PASS – The evaluator confirmed the TOE rejects the SSH connection when authentication using an unknown keypair is attempted.

NIAP TD0631

255 Test 3: [Conditional] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, 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 connecting SSH client.

Note	This is covered in FIA_UIA_EXT.1 Test 1. The TOE accepts password-based authentication. User authentication succeeds when the correct password is provided by the connecting SSH client.

NIAP TD0631

256 Test 4: [Conditional] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, the evaluator shall configure the TOE to accept password-based authentication and demonstrate that user authentication fails when the incorrect password is provided by the connecting SSH client.

Note	This is covered in FIA_UIA_EXT.1 Test 1. The TOE accepts password-based authentication. User authentication fails when the incorrect password is provided by the connecting SSH client.

FCS_SSHS_EXT.1.3

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

High-Level Test Description
Using a custom SSH client, log into the TOE using a valid username and password but ensure that a large packet is transmitted and verify the connection is terminated.
Findings
PASS – The evaluator confirmed that the TOE dropped the packet with larger size than the threshold.

FCS_SSHS_EXT.1.4

258 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.

High-Level Test Description
Establish an SSH connection to the TOE from a client and use a packet capture application to show that the communication is successful and the TOE encryption algorithms include only the ciphers claimed in the ST. Ensure that there are no additional ciphers claimed by the implementation that differ from the ST requirements.
Findings
PASS – The evaluator confirmed that the TOE advertises only the claimed ciphers in the ST, no other ciphers are claimed, and the encryption utilizes aes-128-gcm@openssh.com that was claimed in the ST.

FCS_SSHS_EXT.1.5

NIAP TD0631

259 Test objective: This test case is meant to validate that the TOE server will support host public keys of the claimed algorithm types.

NIAP TD0631

260 Test 1: The evaluator shall configure (only if required by the TOE) the TOE to use each of the claimed host public key algorithms. The evaluator will then use an SSH client to confirm that the client can authenticate the TOE server public key using the claimed algorithm. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

High-Level Test Description
Using an SSH client, connect to the TOE server and capture the TOE server's host key algorithms. Verify that the client successfully connects using each claimed host key algorithm.
Findings
PASS – The evaluator confirmed that the SSH authentication successfully used the claimed host key algorithms.

NIAP TD0631

261 Has effectively been moved to FCS_SSHS_EXT.1.2.

NIAP TD0631

262 Test objective: This negative test case is meant to validate that the TOE server does not support host public key algorithms that are not claimed.

NIAP TD0631

263 Test 2: The evaluator shall configure a non-TOE SSH client to only allow it to authenticate an SSH server host public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the non-TOE SSH client to the TOE SSH server and observe that the connection is rejected.

High-Level Test Description
Using an SSH client, forcibly attempt to negotiate an SSH host key using an unsupported host key algorithm and show it is unsuccessful.
Findings
PASS – The evaluator confirmed the connection failed and the TOE produced audit log message.

FCS_SSHS_EXT.1.6

264 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.

265 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.

High-Level Test Description

Using SSH client, log into the TOE using each of the claimed integrity algorithms in turn and show that the communication is successful. Review the negotiation line from the server to ensure that there are no additional integrity algorithms claimed by the implementation that differ from the ST.
Findings
PASS – The evaluator confirmed that for each integrity algorithm, the connection is successful, and the TOE does not advertise any unclaimed integrity algorithm.

266 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.

267 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.

High-Level Test Description
Using SSH client, log into the TOE using each the hmac-md5 integrity algorithm and show that the communication is unsuccessful.
Findings
PASS – The evaluator confirmed that the connection failed when using the MAC algorithm not claimed in the ST and audit log message was emitted.

FCS_SSHS_EXT.1.7

268 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.

High-Level Test Description
Using SSH client, log into the TOE using diffie-hellman-group-1-sha1 key exchange algorithm and show that the communication is unsuccessful.
Findings
PASS – The evaluator confirmed that the connection failed when establishing an SSH connection with a diffie-hellman-group1-sha1 KEX algorithm, and audit message was emitted.

269 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.

High-Level Test Description
Using SSH client, log into the TOE using each of the claimed key exchange algorithms and show that the communication is successful.
Findings
PASS – The evaluator confirmed that logging in to the TOE via SSH using the KEX algorithms claimed in the ST was successful.

FCS_SSHS_EXT.1.8

270 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.

271 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).

272 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.

High-Level Test Description
Using a custom SSH client, log into the TOE and keep the connection open for over 1 hour. Verify the TOE initiates a rekey of the SSH session when the time-based threshold is reached
Findings
PASS – The evaluator confirmed that the rekey had been initiated by the TOE shortly after 1 hour elapsed.

273 For testing of the traffic-based threshold the evaluator shall use the TOE to connect to an SSH client and shall transmit data to and/or receive data from the TOE within the active SSH session until the threshold for data protected by either encryption key is reached. It is acceptable if the rekey occurs before the threshold is reached (e.g. because the traffic is counted according to one of the alternatives given in the Application Note for FCS_SSHS_EXT.1.8).

274 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).

275 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.

High-Level Test Description
Using a custom SSH client, connect to the TOE and send large amounts of data over the channel. Ensure that the TOE rekeys before 1 GB in the aggregate has been transmitted. Ensure that the TOE is responsible for sending the rekey initiation.
Findings
PASS – The evaluator confirmed that the TOE rekeys when the 1 GB threshold is reached.

276 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).

Note	Neither threshold is configurable.

277 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.

Note	There are no hardware limitations that affect the Traffic-Based Threshold rekey test.

5.1.2 FCS_TLSC_EXT.1 Extended: TLS Client Protocol without mutual authentication

5.1.2.1 TSS

FCS_TLSC_EXT.1.1

278 The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified include those listed for this component.

Findings
PASS
[ST] section 6.2.10 claims 8 TLS ciphersuite. This list matches the selections in FCS_TLSC_EXT.1.1 and all algorithms required to support the ciphersuites are claimed in FCS_CKM.* and FCS_COP.1/*.

FCS_TLSC_EXT.1.2

279 The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the administrator/application configured reference identifier, including which types of reference identifiers are supported (e.g. application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported.

Findings
PASS
[ST] section 6.2.10 claims support for DNS or IP address reference identifiers. The reference identifiers are automatically configured by the TOE. Wildcards are not supported.

280 Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the "joining" component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected

attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

Findings
PASS
The TOE is not a distributed TOE.

281 If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC 5952 for IPv6, RFC 3986 for IPv4) is enforced.

Findings
PASS
[ST] section 6.2.10 says, "The TOE converts IPv4 address in the CN to binary format and stores them in an array in network byte order. The TOE enforces the RFC 3986 for IPv4 canonical format."

FCS_TLSC_EXT.1.4

282 The evaluator shall verify that TSS describes the Supported Elliptic Curves/Supported Groups Extension and whether the required behaviour is performed by default or may be configured.

Findings
PASS
[ST] section 6.2.10 claims the TOE does not present the Supported Elliptic Curves extension. This claim is consistent with the claimed ciphersuites.

5.1.2.2 Guidance Documentation

FCS_TLSC_EXT.1.1

283 The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Findings
PASS
[AGD] section 2.7 indicates FIPS Mode enforces the TLS configuration. This means that no configuration of TLS is needed.

FCS_TLSC_EXT.1.2

284 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings
PASS
[AGD] section 2.10.1 identifies the supported reference identifier types (IP or DNS), that reference identifiers are configured automatically, how to configure a logging server, and that the SAN extension is supported.

285 Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1, the SFR selects attributes from RFC 5280, and FCO_CPC_EXT.1.2 selects “no channel”; the evaluator shall verify the guidance provides instructions for establishing unique reference identifiers based on RFC5280 attributes.

Findings
PASS
The TOE is not a distributed TOE.

FCS_TLSC_EXT.1.4

286 If the TSS indicates that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Elliptic Curves/Supported Groups Extension.

Findings
PASS
[ST] section 6.2.10 indicates the Supported Elliptic Curves Extension is not presented, so guidance does not describe how to configure the extension.

5.1.2.3 Tests

NIAP TD0670

287 **Removed:** ~~For all tests in this chapter the TLS server used for testing of the TOE shall be configured not to require mutual authentication.~~

FCS_TLSC_EXT.1.1

288 Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the

ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to negotiate all specifically claimed ciphersuites. Verify the connection succeeds and the ciphersuite can be found in the packet capture.
Findings
PASS – The evaluator confirmed that for each claimed ciphersuite, the negotiation of the TLS connection succeeded and the ciphersuite can be found in the packet capture.

289 Test 2: The evaluator shall attempt to establish the connection using a server with a server certificate that contains the Server Authentication purpose in the extendedKeyUsage field and verify that a connection is established. The evaluator will then verify that the client rejects an otherwise valid server certificate that lacks the Server Authentication purpose in the extendedKeyUsage field, and a connection is not established. Ideally, the two certificates should be identical except for the extendedKeyUsage field.

High-Level Test Description
Construct two X.509 certificates: one with an extendedKeyUsage with ‘serverAuth’ and another without an extendedKeyUsage. Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server and show that the X.509 certificate without the EKU fails.
Findings
PASS – The evaluator confirmed for the similar certificate that established a successful connection, removing the Server Authentication purpose in the extendedKeyUsage field results in the TOE rejecting the certificate and a connection is not established.

290 Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send an ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite). The evaluator shall verify that the TOE disconnects after receiving the server’s Certificate handshake message.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server using any of the claimed ciphersuites. The Lightship TLS server will send back an otherwise validly constructed server certificate which does not match the requested the ciphersuite. Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server sends a validly constructed server certificate whose signature algorithm does not match the selected ciphersuite.

291 Test 4: The evaluator shall perform the following 'negative tests':

a) The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server using the TLS_NULL_WITH_NULL_NULL (cipher ID 0x0000). Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server uses ciphersuite TLS_NULL_WITH_NULL_NULL (cipher ID 0x0000).

b) Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite not presented in the Client Hello handshake message. The evaluator shall verify that the client rejects the connection after receiving the Server Hello.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a ciphersuite not proposed by the TOE. Verify TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server uses a ciphersuite not proposed by the TOE.

c) [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension the evaluator shall configure the server to perform an ECDHE or DHE key exchange in the TLS connection using a non-supported curve/group (for example P-192) and shall verify that the TOE disconnects after receiving the server's Key Exchange handshake message.

Test Not Applicable	The TOE does not support Elliptic Curves/Supported Groups Extension in the client hello.

292 Test 5: The evaluator performs the following modifications to the traffic:

a) Change the TLS version selected by the server in the Server Hello to a non-supported TLS version and verify that the client rejects the connection.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server advertising a non-supported TLS version. Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE rejects the connection when the Lightship TLS server advertises a non-supported TLS version.

b) [conditional]: If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the handshake does not finished successfully, and no application data flows. This test does not apply to cipher suites

using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a mangled key exchange signature. Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server sends a certificate with a mangled key exchange signature.

293 Test 6: The evaluator performs the following 'scrambled message tests':

a) Modify a byte in the Server Finished handshake message and verify that the handshake does not finish successfully and no application data flows.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a mangled finished message. Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server modified the finished message.

b) Send a garbled message from the server after the server has issued the ChangeCipherSpec message and verify that the handshake does not finish successfully and no application data flows.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a mangled finished message. Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that the TOE terminates the TLS handshake when the Lightship TLS server replaced the finished messages with garbled bytes.

c) Modify at least one byte in the server's nonce in the Server Hello handshake message and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE ciphersuite) or that the server denies the client's Finished handshake message.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a modified nonce value. Do this once for a non-DHE ciphersuite and once for a DHE or ECDHE key exchange ciphersuite.

Verify the TOE terminates the TLS handshake.
Findings
PASS – The evaluator confirmed that in both attempts with DHE ciphersuite and non-DHE ciphersuite, the TOE terminates the TLS handshake when the Lightship TLS server modified the last byte in Server Hello nonce.

FCS_TLSC_EXT.1.2

294 Note that the following tests are marked conditional and are applicable under the following conditions:

a) For TLS-based trusted channel communications according to FTP_ITC.1 where RFC 6125 is selected, tests 1-6 are applicable.

or

b) For TLS-based trusted path communications according to FTP_TRP where RFC 6125 is selected, tests 1-6 are applicable

or

c) For TLS-based trusted path communications according to FPT_ITT.1 where RFC 6125 is selected, tests 1-6 are applicable. Where RFC 5280 is selected, only test 7 is applicable.

Note that for some tests additional conditions apply.

295 IP addresses are binary values that must be converted to a textual representation when presented in the CN of a certificate. When testing IP addresses in the CN, the evaluator shall follow the following formatting rules:

- IPv4: The CN contains a single address that is represented a 32-bit numeric address (IPv4) is written in decimal as four numbers that range from 0-255 separated by periods as specified in RFC 3986.

- IPv6: The CN contains a single IPv6 address that is represented as eight colon separated groups of four lowercase hexadecimal digits, each group representing 16 bits as specified in RFC 4291. Note: Shortened addresses, suppressed zeros, and embedded IPv4 addresses are not tested.

296 The evaluator shall configure the reference identifier per the AGD guidance and perform the following tests during a TLS connection:

a) Test 1 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the CN.

Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead

of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

High-Level Test Description
Lightship developed TLS server sends X.509 certificates containing a CN that does not match the reference identifier and does not contain the SAN extension. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection fails and there will be no Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection fails when the server certificate that contains a CN that does not match the reference identifier and does not contain the SAN extension.

b) Test 2 [conditional]: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, URI). When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the SAN.

High-Level Test Description
Lightship developed TLS server sends X.509 certificates that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection fails and there will be no Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection fails when a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier.

c) Test 3 [conditional]: If the TOE does not mandate the presence of the SAN extension, the evaluator shall present a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each identifier type (e.g. IPv4, IPv6, FQDN) supported in the CN. If the TOE does mandate the presence of the SAN extension, this Test shall be omitted.

High-Level Test Description
Lightship developed TLS server sends X.509 certificates containing a CN that matches the reference identifier and does not contain the SAN extension. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection succeeds and there will be Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection succeeds when a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension.

d) Test 4 [conditional]: The evaluator shall present a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator shall verify that the connection succeeds. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, SRV).

High-Level Test Description
Lightship developed TLS server sends X.509 certificates containing a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection succeeds and there will be Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection succeeds when a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches.

e) Test 5 [conditional]: The evaluator shall perform the following wildcard tests with each supported type of reference identifier that includes a DNS name (i.e. CN-ID with DNS, DNS-ID, SRV-ID, URI-ID):

1) [conditional]: The evaluator shall present a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g. foo.*.example.com) and verify that the connection fails.

High-Level Test Description
Lightship developed TLS server sends X.509 certificates containing a wildcard that is not in the left-most label of the presented identifier (e.g., foo.*.example.com), one cert in the CN and one cert in the SAN. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection fails and there will be no Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection fails when a server certificate containing a wildcard that is not in the left-most label of the presented identifier (e.g., foo.*.example.com).

2) [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds, if wildcards are supported, or fails if wildcards are not supported. The evaluator shall configure the reference identifier without a left-most label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails. (Remark: Support for wildcards was always intended to be optional. It is sufficient to state that the TOE does not support wildcards and observe rejected connection attempts to satisfy corresponding assurance activities.)

High-Level Test Description
Have the TOE attempt connections to a TLS server with the following reference identifiers: • foo.example.com • example.com • bar.foo.example.com

Have the TLS server establish connections while presenting a certificate with *.example.com in the CN and, for a separate sect of connections, *.example.com in the SAN. Verify all connections fail and no Application Data flows.
Findings
PASS – The evaluator confirmed that all the connections failed when wildcards are not supported and the server certificates contains a wildcard in the left-most label (e.g., *.example.com).

NIAP TD0634

f) Test 6 [conditional]: [conditional] If IP address identifiers supported in the SAN or CN, the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) (e.g. CN=*.168.0.1 when connecting to 192.168.0.1, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 6.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending X.509 certificates that have the characteristics required by the test. The TOE client attempts a TLS connection with the correct reference identifier. Verify the connection fails and there will be no Application Data found in the packet capture.
Findings
PASS – The evaluator confirmed that the connection fails when a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) and not contain a SAN extension.

297 Test 7 [conditional]: If the secure channel is used for FPT_ITT, and RFC 5280 is selected, the evaluator shall perform the following tests. Note, when multiple attribute types are selected in the SFR (e.g. when multiple attribute types are combined to form the unique identifier), the evaluator modifies each attribute type in accordance with the matching criteria described in the TSS (e.g. creating a mismatch of one attribute type at a time while other attribute types contain values that will match a portion of the reference identifier):

1) The evaluator shall present a server certificate that does not contain an identifier in the Subject (DN) attribute type(s) that matches the reference identifier. The evaluator shall verify that the connection fails.

2) The evaluator shall present a server certificate that contains a valid identifier as an attribute type other than the expected attribute type (e.g. if the TOE is configured to expect id-at-serialNumber=correct_identifier, the certificate could instead include id-at-name=correct_identifier), and does not contain the SAN extension. The evaluator shall verify that the connection fails. Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the

reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass this test.

3) The evaluator shall present a server certificate that contains a Subject attribute type that matches the reference identifier and does not contain the SAN extension. The evaluator shall verify that the connection succeeds.

4) The evaluator shall confirm that all use of wildcards results in connection failure regardless of whether the wildcards are used in the left or right side of the presented identifier. (Remark: Use of wildcards is not addressed within RFC 5280.)

Test Not Applicable	The TOE does not claim FPT_ITT.1 with RFC 5280.

298 **FCS_TLSC_EXT.1.3**

299 The evaluator shall demonstrate that using an invalid certificate results in the function failing as follows:

300 Test 1: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds and a trusted channel can be established.

Note	This test case is performed as part of FIA_X509_EXT.1.1/Rev Test 1

301 Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Note	This test case is performed as part of FIA_X509_EXT.1.1/Rev Test 1. No override mechanisms are claimed.

302 Test 3 [conditional]: The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

Test Not Applicable	No override mechanisms are claimed.

FCS_TLSC_EXT.1.4

303 Test 1 [conditional]: If the TOE presents the Supported Elliptic Curves/Supported Groups Extension, the evaluator shall configure the server to perform ECDHE or DHE (as applicable) key exchange using each of the TOE's supported curves and/or groups. The evaluator shall verify that the TOE successfully connects to the server.

Test Not Applicable	Supported Elliptic Curves/Supported Groups Extension is not present in Client Hello - as claimed in the ST.

5.2 Identification and Authentication (FIA)

5.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

5.2.1.1 TSS

304 The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

Findings
PASS
[ST] section 6.3.6 indicates that certificate validity is checked when the TLS client validates the server certificate and when certificates are loaded onto the TOE.
[ST] section 6.3.6 indicates the TOE does not check for the code-signing or clientAuthentication EKUs because the TOE does not use X.509 certificates for trusted updates, firmware integrity, or client authentication.
Note: Revocation checking is covered in the finding below.

305 The TSS shall describe when revocation checking is performed and on what certificates. If the revocation checking during authentication is handled differently depending on whether a full certificate chain or only a leaf certificate is being presented, any differences must be summarized in the TSS section and explained in the Guidance.

Findings
PASS
[ST] section 6.3.6 indicates certificate revocation checking is performed for the "above scenarios" (i.e., when the TLS client validates the server certificate and when certificates are loaded onto the TOE). The TOE uses CRLs.

5.2.1.2 Guidance Documentation

306 The evaluator shall also ensure that the guidance documentation describes where the check of validity of the certificates takes place, describes any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e.

where the ST is therefore claiming that they are trivially satisfied) and describes how certificate revocation checking is performed and on which certificate.

Findings
PASS
[AGD] section 2.10.2 indicates the validity of certificates are checked when establishing a TLS connection to the syslog server.
[AGD] section 2.10.2 indicates the TOE verifies the serverAuthentication EKU and does not use certificates in a way that would require validation of other EKUs.

5.2.1.3 Tests

307 The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step or when performing trusted updates (if FPT_TUD_EXT.2 is selected). It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the TOE. It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected). The evaluator shall perform the following tests for FIA_X509_EXT.1.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function and shall use this chain to demonstrate that the function succeeds. Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store)

   Test 1b: The evaluator shall then 'break' the chain used in Test 1a by either removing the trust anchor in the TOE's trust store used to terminate the chain, or by removing one of the intermediate CA certificates (provided together with the leaf certificate in Test 1a) to complete the chain. The evaluator shall show that an attempt to validate this broken chain fails.

High-Level Test Description
With the Root CA in the TOE trust store: • Without Intermediate CA in the TOE trust store, force the TOE to connect to a TLS server that sends back the Intermediate and show that the connection is accepted. • Without Intermediate CA in the TOE trust store, force the TOE to connect to a TLS server that does not send back the Intermediate CA. Show that the connection is not accepted. • With the Intermediate CA in the TOE trust store, force the TOE to connect to a TLS server that does not send back the Intermediate CA. Show that the connection is accepted.
Findings
PASS – The evaluator confirmed that the TOE properly performed validity check on the certificate used in an authentication step and the connection succeeded.

b) Test 2: The evaluator shall demonstrate that validating an expired certificate results in the function failing.

High-Level Test Description
Create an X.509 certificate with a 'notAfter' date in the past. Force the TOE to connect to a TLS server that sends back this certificate and show it is not accepted. Show that CA certificates in the trust store that expire after being loaded result in an error.
Findings
PASS – The evaluator confirmed that the TOE properly validated an expired certificate and resulted in the connection failing.

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates--conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

Note	The TOE does not claim OCSP. Therefore, only CRL testing is performed.

High-Level Test Description
Load the CA into the TOE trust store. Ensure the CRLs are empty. Revoke the intermediate CA and place it into the CRL and load the CRL into the TOE. Verify the connection now fails due to the certificate being revoked. Unrevoke the intermediate CA. Revoke the server certificate and place it into the CRL and load it into the TOE. Verify the connection now fails due to the server certificate being revoked. Verify that a certificate now results in a successful connection.
Findings
PASS – The evaluator confirmed that the TOE can properly handle revoked certificates and the connection failed.

d) Test 4: If OCSP is selected, the evaluator shall configure the OCSP server or use a man-in-the-middle tool to present a certificate that does not have the OCSP signing purpose and verify that validation of the OCSP response fails. If CRL is selected, the evaluator shall configure the CA to sign a CRL with a certificate that does not have the cRLsign key usage bit set and verify that validation of the CRL fails.

Note	The TOE does not claim OCSP. Therefore, only CRL testing is performed.

High-Level Test Description
Ensure the CA is installed in the TOE trust store. Sign the empty Intermediate CRL and a new GLLeaf cert with Intermediate Cert that does not have a CRL Sign Bit.

The TLS connection will fail because the CRL cannot be validated.
Findings
PASS – The evaluator confirmed that a CRL signed with a CA certificate without CRLSign key usage bit resulted in validation failing.

e) Test 5: The evaluator shall modify any byte in the first eight bytes of the certificate and demonstrate that the certificate fails to validate. (The certificate will fail to parse correctly.)

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back a properly mangled X.509 certificate in which the ASN.1 header bytes in the first 8 bytes are modified. Verify the connection failed.
Findings
PASS – The evaluator confirmed that the modified certificate failed to validate and the connection terminated.

f) Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back an X.509 certificate in which the last byte of the certificate (the signature) is modified. Verify the connection failed.
Findings
PASS – The evaluator confirmed that the modified certificate failed to validate and the connection terminated.

g) Test 7: The evaluator shall modify any byte in the public key of the certificate and demonstrate that the certificate fails to validate. (The hash of the certificate will not validate.)

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back an X.509 certificate in which the public key of the certificate is modified.
Findings
PASS – The evaluator confirmed that the modified certificate failed to validate and the connection terminated.

NIAP TD0527 (REVISED 1 December 2020)

308 The following tests are run when a minimum certificate path length of three certificates is implemented.

NIAP TD0527 (REVISED 1 December 2020)

h) Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall conduct the following tests:

NIAP TD0527 (REVISED 1 December 2020)

Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test Not Applicable	The TOE does not claim EC certificates.

NIAP TD0527 (REVISED 1 December 2020)

Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

Test Not Applicable	The TOE does not claim EC certificates.

NIAP TD0527 (REVISED 1 December 2020)

Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

Test Not Applicable	The TOE does not claim EC certificates.

309 The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore

claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

310 The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation).

311 For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

a) Test 1: The evaluator shall ensure that at least one of the CAs in the chain does not contain the basicConstraints extension. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Load a known-good CA into the TOE trust store. Verify that connecting to our test server will yield a successful result. Clone the known good CA certificate and remove the basicConstraints extension. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings
PASS – The evaluator confirmed that the certificate with Basic Constraint removed failed to validate and the connection terminated.

b) Test 2: The evaluator shall ensure that at least one of the CA certificates in the chain has a basicConstraints extension in which the CA flag is set to FALSE. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate with the CA flag set to FALSE to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Clone the known good CA certificate and set the basicConstraints extension to have the CA flag set to FALSE. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings
PASS – The evaluator confirmed that the certificate with Basic Constraint value set to FALSE failed to validate and the connection terminated.

312 The evaluator shall repeat these tests for each distinct use of certificates. Thus, for example, use of certificates for TLS connection is distinct from use of certificates for trusted updates so both of these uses would be tested. But there is no need to repeat the tests for

each separate TLS channel in FTP_ITC.1 and FTP_TRP.1/Admin (unless the channels use separate implementations of TLS).

Note	The distinct uses of certificates are covered in the tests above.

5.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication

5.2.2.1 TSS

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

Findings

PASS

[ST] section 6.3.6 implicitly specifies how the CA certificates are selected via the path validation algorithm.

[ST] section 6.3.6 indicates the cert presented by the TLS server is chosen for TLS server connections.

Please see section 4.1.1.1 for details regarding the TOE's TLS client certificate.

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

Findings

PASS

[ST] section 6.3.7 indicates the last cached status is used when the revocation status cannot be determined and that the TOE accepts the certificate if no cached status is available.

5.2.2.2 Guidance Documentation

315 The evaluator shall also ensure that the guidance documentation describes the configuration required in the operating environment so the TOE can use the certificates. The guidance documentation shall also include any required configuration on the TOE to use the certificates. The guidance document shall also describe the steps for the Security Administrator to follow if the connection cannot be established during the validity check of a certificate used in establishing a trusted channel.

Findings

PASS

[AGD] section 2.10.2 describes how to configure CRL checking by configuring CDPs on the TOE.

5.2.2.3 Tests

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

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

High-Level Test Description
Show that if the CRL cannot be fetched, the TOE will validate the certificate based on the last cached information.
Findings
PASS – The evaluator confirmed that when the CRL cannot be fetched from a non-TOE entity, the TOE will validate the certificate based on the last cached information or accept the certificate when no cached information is available.

5.2.3 FIA_X509_EXT.3 Extended: X509 Certificate Requests

5.2.3.1 TSS

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

Findings
PASS
[ST] section 5.3.3, FIA_X509_EXT.3.1 does not select “device-specific information.”

5.2.3.2 Guidance Documentation

319 The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate Request. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the Certification Request.

Findings
PASS
[ADMIN] section ’Request and install host certificates’ describes how to generate CSRs. The guidance indicates the Common Name can be specified using the ‘cname’ parameter.

5.2.3.3 Tests

320 The evaluator shall perform the following tests:

a) Test 1: The evaluator shall use the guidance documentation to cause the TOE to generate a Certification Request. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the Certification Request provides the public key and other required information, including any necessary user-input information.

High-Level Test Description
Using the TOE CSR generator, create a new CSR and download it to an external CA entity for signing. Using OpenSSL, verify that the information in the CSR is as expected.
Findings
PASS – The evaluator confirmed the TOE can generate certificate request messages and includes the information claimed in the ST.

b) Test 2: The evaluator shall demonstrate that validating a response message to a Certification Request without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message and demonstrate that the function succeeds.

High-Level Test Description
The CSR from the previous test is signed by a CA which is not yet loaded in the TOE trust store. It is imported into the TOE but fails verification when the CAs are missing. Once the CAs are added, the verification step succeeds.
Findings
PASS – The evaluator confirmed the TOE rejects a signed CSR (certificate) when the trust chain is incomplete and accepts a signed CSR when the trust chain can be completed.

5.3 Security management (FMT)

5.3.1 FMT_MOF.1/Functions Management of security functions behaviour

5.3.1.1 TSS

321 For distributed TOEs see [ND-SD] chapter 2.4.1.1.

Findings
PASS
The TOE is not a distributed TOE.

322 For non-distributed TOEs, the evaluator shall ensure the TSS for each administrative function identified the TSS details how the Security Administrator determines or modifies the behaviour of (whichever is supported by the TOE) transmitting audit data to an external IT entity, handling of audit data, audit functionality when Local Audit Storage Space is full (whichever is supported by the TOE).

Findings
PASS

[ST] section 6.4.2 indicates the administrator can enable/disable transmission of external audit data to an external IT entity. This is consistent with the selection in FMT_MOF.1.1/Functions.

5.3.1.2 Guidance Documentation

323 For distributed TOEs see [ND-SD] chapter 2.4.1.2.

Findings

PASS

The TOE is not a distributed TOE.

324 For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the Security Administrator determines or modifies the behaviour of (whichever is supported by the TOE) transmitting audit data to an external IT entity, handling of audit data, audit functionality when Local Audit Storage Space is full (whichever is supported by the TOE) are performed to include required configuration settings.

Findings

PASS

[AGD] section 2.10.1 and [ADMIN] section ‘System logging over TLS’ describe how to configure a TLS trusted channel with an audit server. These descriptions include configuring mutual authentication for the channel.

5.3.1.3 Tests

325 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.

Note	The TOE only supports the Security Administrator role and does not allow any administrative actions prior to authentication as a Security Administrator, so this is tested as part of FIA_UIA_EXT.1 Tests 2 and 3.

326 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.

327 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.

Note	This is covered in FIA_X509_EXT.1.1/Rev Test 1 where an additional audit server is successfully configured.

328 Test 1 (if 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the handling of audit data without prior authentication as Security Administrator (by authentication as a user with no administrator privileges or without user authentication at all). Attempts to modify parameters without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator. The term ‘handling of audit data’ refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2/LocSpace.

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

329 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.

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

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

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

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

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

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

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

334 Test 3 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen from the second selection without prior authentication as Security Administrator (by authentication as a user with no administrator privileges or without user authentication at all). This can be done in one test or in separate tests. The attempt(s) to determine the behaviour of the selected functions without administrator authentication shall fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

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

Test Not Applicable	The ST does not claim this functionality and this test will not be conducted.

5.3.2 FMT_MTD.1/CryptoKeys Management of TSF Data

5.3.2.1 TSS

336 For distributed TOEs see [ND-SD] chapter 2.4.1.1.

Findings
PASS
The TOE is not a distributed TOE.

337 For non-distributed TOEs, the evaluator shall ensure the TSS lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Findings
PASS
[ST] sections 6.4.3 and 6.4.5 describe the Security Administrator’s ability to manage keys and the options available.

5.3.2.2 Guidance Documentation

338 For distributed TOEs see [ND-SD] chapter 2.4.1.2.

Findings
PASS
The TOE is not a distributed TOE.

339 For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation lists the keys the Security Administrator is able to manage to include the options available (e.g. generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Findings
PASS
[AGD] section 2.6 describes managing the SSH hostkeys. [AGD] section 2.10.2 and [ADMIN] section ‘X.509v3 certificates’ > ‘Manage CA certificates’ provide guidance for the secure management of the trust store. [ADMIN] section ’Request and install host certificates’ describes how to generate CSRs and manage X.509 private keys.

5.3.2.3 Tests

340 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.

Note	The TOE only supports the Security Administrator role and does not allow any administrative actions prior to authentication as a Security Administrator, so this is tested as part of FIA_UIA_EXT.1 Test 2 and 3.

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

Note	Generating a CSR also generates a private key which is covered in FIA_X509_EXT.3 Test 1.

High-Level Test Description
As the privileged user, attempt to generate an SSH private key and show it does succeed.
Findings

PASS – The evaluator confirmed that user with prior authentication as Security Administrator successfully generated an SSH private key and CSR private key.

6 Evaluation Activities for Security Assurance Requirements

6.1 ASE: Security Target

6.1.1 General ASE

342 When evaluating a Security Target, the evaluator performs the work units as presented in the CEM. In addition, the evaluator ensures the content of the TSS in the ST satisfies the EAs specified in Section 2 (Evaluation Activities for SFRs).

Findings
PASS
See above sections.

343 For distributed TOEs only the SFRs classified as ‘all’ have to be fulfilled by all TOE parts. The SFRs classified as ‘One’ or ‘Feature Dependent’ only have to be fulfilled by either one or some TOE parts, respectively. To make sure that the distributed TOE as a whole fulfills all the SFRs the following actions for ASE_TSS.1 have to be performed as part of ASE_TSS.1.1E.

ASE_TSS.1 element	**Evaluator Action**
ASE_TSS.1.1C	The evaluator shall examine the TSS to determine that it is clear which TOE components contribute to each SFR or how the components combine to meet each SFR. The evaluator shall verify the sufficiency to fulfil the related SFRs. This includes checking that the TOE as a whole fully covers all SFRs and that all functionality that is required to be audited is in fact audited regardless of the component that carries it out.

Findings
PASS – N/A
The TOE is not a distributed TOE.

6.2 ADV: Development

6.2.1 Basic Functional Specification (ADV_FSP.1)

344 The EAs for this assurance component focus on understanding the interfaces (e.g., application programming interfaces, command line interfaces, graphical user interfaces, network interfaces) described in the AGD documentation, and possibly identified in the

TOE Summary Specification (TSS) in response to the SFRs. Specific evaluator actions to be performed against this documentation are identified (where relevant) for each SFR in Section 2, and in EAs for AGD, ATE and AVA SARs in other parts of Section 5.

345 The EAs presented in this section address the CEM work units ADV_FSP.1-1, ADV_FSP.1-2, ADV_FSP.1-3, and ADV_FSP.1-5.

346 The EAs are reworded for clarity and interpret the CEM work units such that they will result in more objective and repeatable actions by the evaluator. The EAs in this SD are intended to ensure the evaluators are consistently performing equivalent actions.

347 The documents to be examined for this assurance component in an evaluation are therefore the Security Target, AGD documentation, and any required supplementary information required by the cPP: no additional "functional specification" documentation is necessary to satisfy the EAs. The interfaces that need to be evaluated are also identified by reference to the EAs listed for each SFR and are expected to be identified in the context of the Security Target, AGD documentation, and any required supplementary information defined in the cPP rather than as a separate list specifically for the purposes of CC evaluation. The direct identification of documentation requirements and their assessment as part of the EAs for each SFR also means that the tracing required in ADV_FSP.1.2D (work units ADV_FSP.1-4, ADV_FSP.1-6 and ADV_FSP.1-7) is treated as implicit and no separate mapping information is required for this element.

6.2.1.1 Evaluation Activity

348 *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.*

349 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.

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

Findings
PASS
From section 7.2.1 of the [NDcPP]: "For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation."
The [ST] and the guidance documentation comprise the functional specification. The evaluator was able to perform the Evaluation Activities specified in the [ND-SD], so the evaluator concluded that the functional specification sufficiently describes the parameters, purpose, and method of use for each TSFI that is identified as being security relevant.

6.2.1.2 Evaluation Activity

351 *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.*

Findings
PASS
Please see the previous work unit.

6.2.1.3 Evaluation Activity

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

353 The evaluator uses the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 2, including the EAs associated with testing of the interfaces.

354 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.

355 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'.

Findings
PASS
From section 7.2.1 of the [NDcPP]: "For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation."
The [ST] and the guidance documentation comprise the functional specification. The interfaces are implicitly mapped to SFRs if they are used to satisfy an Evaluation Activity for a specific SFR. The evaluator was able to perform the Evaluation Activities specified in the [ND-SD], the Findings for SFR related Evaluation Activities are the mapping of interfaces to SFRs.

6.3 AGD: Guidance Documents

356 It is not necessary for a TOE to provide separate documentation to meet the individual requirements of AGD_OPE and AGD_PRE. Although the EAs in this section are described under the traditionally separate AGD families, the mapping between the documentation provided by the developer and AGD_OPE and AGD_PRE requirements may be many-to-many, as long as all requirements are met in documentation that is delivered to Security Administrators and users (as appropriate) as part of the TOE.

357 Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in section A.9.1.1. (in the NDcPP-SD)

6.3.1 Operational User Guidance (AGD_OPE.1)

358 The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR.

359 In addition, the evaluator performs the EAs specified below.

6.3.1.1 Evaluation Activity

360 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.

Findings
PASS
The guidance documentation is posted to the NIAP website, ensuring administrators are aware of the documentation. The guidance documentation is also posted to the vendor's website at https://www.dell.com/support/home/en-ee/product-support/product/smartfabric-os10-emp-partner/docs.

6.3.1.2 Evaluation Activity

361 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.

Findings
PASS
There is only one operational environment claimed in [ST] section 2.2, Figure 1. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency.

6.3.1.3 Evaluation Activity

362 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.

Findings
PASS
[AGD] section 2.7 describes the configuration of the cryptographic operations (i.e., engines, protocols, algorithms, key sizes) to be consistent with the evaluated configuration. No configuration once the TOE is in the evaluated configuration is necessary. The [AGD] provides instructions for ensuring the evaluated functionality is used.

6.3.1.4 Evaluation Activity

363 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.

Findings
PASS
[AGD] section 1.3 clarifies the evaluated functionality. The evaluator confirmed [AGD] covers configuration of the in-scope functionality where additional configuration might be required.

6.3.1.5 Evaluation Activity

364 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.

NIAP TD0536

b) The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps:

5) 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).

6) 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.

Findings
PASS
See section 6.3.1.3 for configuration of the cryptographic engine.
[AGD] section 2.4 describes the update process.
See section 6.3.1.4 for details as to what was covered by the EAs.

6.3.2 Preparative Procedures (AGD_PRE.1)

365 The evaluator performs the CEM work units associated with the AGD_PRE.1 SAR. Specific requirements and EAs on the preparative documentation are identified (and where relevant are captured in the Guidance Documentation portions of the EAs) in the individual EAs for each SFR.

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

367 In addition, the evaluator performs the EAs specified below.

6.3.2.1 Evaluation Activity

368 *The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security 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).*

369 The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

Findings
PASS
[AGD] and [ADMIN] are written in a style that an average IT administrator (with general security knowledge, but not a CC/Dell expert) can understand the steps that need to be performed to configure the TOE.

6.3.2.2 Evaluation Activity

370 *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.*

Findings
PASS
There is only one operational environment claimed in [ST] section 2.2, Figure 1. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency.

6.3.2.3 Evaluation Activity

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

Findings
PASS
There is only one operational environment claimed in [ST] section 2.2, Figure 1. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency. While performing testing, the evaluator ensured the instructions are sufficient to successfully install the TOE in the operational environment.

6.3.2.4 Evaluation Activity

372 *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.*

Findings
PASS
[AGD] section 1.3.4 describes the TOE's dependencies on the larger operational environment necessary to maintain the security of the TSF. The guidance documentation provides extensive information on managing the security of the TOE as an individual product. Additional best practice guidance provided within those documents help instill a culture of secure manageability within a larger operational environment.

6.3.2.5 Evaluation Activity

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

374 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.

Findings
PASS
[AGD] section 2.6 describes the protected administrative capability over SSH. [AGD] section 2.8 describes changing default passwords.

6.4 ALC: Life-cycle Support

6.4.1 Labelling of the TOE (ALC_CMC.1)

375 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.

Findings
PASS
While performing the ALC_CMC.1 CEM work units, the evaluator verified the TOE is labeled with a unique reference and the reference is consistent with the ST.

6.4.2 TOE CM coverage (ALC_CMS.1)

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

Findings
PASS
While performing the ALC_CMC.1 CEM work units, the evaluator verified the configuration list contains the TOE and the evaluation evidence required by the SARs. Each configuration item was determined to contain a unique identifier.

6.5 ATE: Tests

6.5.1 Independent Testing – Conformance (ATE_IND.1)

377 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.

378 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 [ND-SD] Sections 2, 3 and 4.

379 The evaluator should consult Appendix 709 when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

380 Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in [ND-SD] section A.9.3.1.

Findings
PASS
The evaluator tested the SFRs by performing the required Test Evaluation Activities for each SFR. The evaluator confirmed the TOE functioned as described in the TSS and the operational guidance was accurate. The ETR covers the ATE_IND.1 CEM work units. The DTR documents the testing strategy and equivalency argument. The TOE is not a distributed TOE.

6.6 Vulnerability Assessment

6.6.1 Vulnerability Survey (AVA_VAN.1)

381 While vulnerability analysis is inherently a subjective activity, a minimum level of analysis can be defined and some measure of objectivity and repeatability (or at least comparability) can be imposed on the vulnerability analysis process. In order to achieve such objectivity and repeatability it is important that the evaluator follows a set of well-defined activities and documents their findings so others can follow their arguments and come to the same conclusions as the evaluator. While this does not guarantee that different evaluation facilities will identify exactly the same type of vulnerabilities or come to exactly the same conclusions, the approach defines the minimum level of analysis and the scope of that analysis and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities.

382 In order to meet these goals some refinement of the AVA_VAN.1 CEM work units is needed. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

383 Because of the level of detail required for the evaluation activities, the bulk of the instructions are contained in Appendix A, while an “outline” of the assurance activity is provided below.

6.6.1.1 Evaluation Activity (Documentation)

384 In addition to the activities specified by the CEM in accordance with [ND-SD] Table 2, the evaluator shall perform the following activities.

385 *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.*

NIAP TD0547

386 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 of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). 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 vulnerability hypotheses during their analysis.

Findings
PASS
The evaluator collected this information from the developer which was used to feed into the Type 1 Flaw Hypotheses search (below).

Distributed TOEs

387 If the TOE is a distributed TOE then the developer shall provide:

a) documentation describing the allocation of requirements between distributed TOE components as in [NDcPP, 3.4]

b) a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, 6.3.3]

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.4.1.2 and 3.5.1.2.

Findings
PASS

The TOE is not a distributed TOE.

6.6.1.2 Evaluation Activity

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

Findings

PASS

The following sources of public vulnerabilities were considered in formulating the specific list of flaws to be investigated by the evaluators, as well as to reference in directing the evaluators to perform key-word searches during the evaluation of the TOE. Hypothesis sources for public vulnerabilities were:

- Dell Security Advisories: https://www.dell.com/support/security/en-us/
- CVEs
  - NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search
  - Common Vulnerabilities and Exposures: http://cve.mitre.org/cve/
  - CVE Details: https://www.cvedetails.com/vulnerability-search.php
- US-CERT: http://www.kb.cert.org/vuls/html/search
- Tenable Network Security https://www.tenable.com/plugins
- Tipping Point Zero Day Initiative http://www.zerodayinitiative.com/advisories
- Offensive Security Exploit Database: https://www.exploit-db.com/
- Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities

Type 1 Hypothesis searches were conducted on August 10, 2023 and included the following search terms:

- Dell EMC Networking SmartFabric OS10 Build 10.5.4.3P1
- Each TOE hardware model
- Each processor model used by the TOE

Note: Additional proprietary search terms were also included.

The evaluation team determined that no residual vulnerabilities exist based on these searches that are exploitable by attackers with Basic Attack Potential.

No Type 2 flaw hypotheses applied to the TOE based on [ND-SD] sections A.1.2 and A.5.

The evaluation team developed Type 3 flaw hypotheses in accordance with [ND-SD] sections A.1.3 and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

The evaluation team developed Type 4 flaw hypotheses in accordance with [ND-SD] sections A.1.4 and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.