Dell EMC

Dell EMC Networking SmartFabric OS10

Assurance Activity Report

Version 1.4

September 15, 2020

Document prepared by

Lightship Security

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 Program (NIAP) reporting guidelines.

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	Canadian Common Criteria Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	Dell EMC
TOE	Dell EMC Networking SmartFabric OS10
Security Target	Dell EMC Networking SmartFabric OS10 Security Target, v1.6
Protection Profile	collaborative Protection Profile for Network Devices, v2.1, 24-September-2018

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, v2.1 (NDcPP-SD)
Interpretations	**NDcPP v2.1** TD0395: NIT Technical Decision for Different Handling of TLS1.1 and TLS1.2 *This TD does not apply to the TOE because it does not claim FCS_TLSS_EXT.2.* TD0396: NIT Technical Decision for FCS_TLSC_EXT.1.1, Test 2 *This TD applies to the TOE.* TD0397: NIT Technical Decision for Fixing AES-CTR Mode Tests *This TD applies to the TOE.*

	TD0398: NIT Technical Decision for FCS_SSH*EXT.1.1 RFCs for AES-CTR *This TD applies to the TOE.*
	TD0399: NIT Technical Decision for Manual installation of CRL (FIA_X509_EXT.2) *This TD applies to the TOE.*
	TD0400: NIT Technical Decision for FCS_CKM.2 and elliptic curve-based key establishment *This TD applies to the TOE.*
	TD0401: NIT Technical Decision for Reliance on external servers to meet SFRs *This TD applies to the TOE.*
	TD0402: NIT Technical Decision for RSA-based FCS_CKM.2 Selection *This TD applies to the TOE.*
	TD0407: NIT Technical Decision for handling Certification of Cloud Deployments *This TD does not apply to the TOE since is not a cloud deployment.*
	TD0408: NIT Technical Decision for local vs. remote administrator accounts *This TD applies to the TOE.*
	TD0409: NIT decision for Applicability of FIA_AFL.1 to key-based SSH authentication *This TD applies to the TOE.*
	TD0410: NIT technical decision for Redundant assurance activities associated with FAU_GEN.1 *This TD applies to the TOE.*
	TD0411: NIT Technical Decision for FCS_SSHC_EXT.1.5, Test 1 - Server and client side seem to be confused *This TD does not apply to the TOE as it does not claim SSHC.*
	TD0412: NIT Technical Decision for FCS_SSHS_EXT.1.5 SFR and AA discrepancy *This TD applies to the TOE.*
	TD0423: NIT Technical Decision for Clarification about application of Rfl#201726Rev2 *This TD applies to the TOE.*
	TD0424: NIT Technical Decision for NDcPP v2.1 Clarification - FCS_SSHC/S_EXT1.5

	This TD applies to the TOE.
	TD0425: NIT Technical Decision for Cut-and-paste Error for Guidance AA *This TD applies to the TOE.*
	TD0447: NIT Technical Decision for Using 'diffie-hellman-group-exchange-sha256' in FCS_SSHC/S_EXT.1.7 *This TD applies to the TOE.*
	TD0450: NIT Technical Decision for RSA-based ciphers and the Server Key Exchange message *This TD does not apply to the TOE as it does not claim TLSS.*
	TD0451: NIT Technical Decision for ITT Comm UUID Reference Identifier *This TD does not apply to the TOE as it does not claim TLSS.*
	TD0453: NIT Technical Decision for Clarify authentication methods SSH clients can use to authenticate SSH servers *This TD does not apply to the TOE as it does not claim SSHC.*
	TD0475: NIT Technical Decision for Separate traffic consideration for SSH rekey *This TD applies to the TOE.*
	TD0477: NIT Technical Decision for Clarifying FPT_TUD_EXT.1 Trusted Update *This TD applies to the TOE.*
	TD0478: NIT Technical Decision for Application Notes for FIA_X509_EXT.1 iterations *This TD applies to the TOE.*
	TD0480: NIT Technical Decision for Granularity of audit events *This TD applies to the TOE.*
	TD0481: NIT Technical Decision for FCS_(D)TLSC_EXT.X.2 IP addresses in reference identifiers *This TD applies to the TOE.*
	TD0482: NIT Technical Decision for Identification of usage of cryptographic schemes *This TD applies to the TOE.*
	TD0483: NIT Technical Decision for Applicability of FPT_APW_EXT.1

	This TD applies to the TOE.
	TD0484: NIT Technical Decision for Interactive sessions in FTA_SSL_EXT.1 & FTA_SSL.3
	This TD applies to the TOE.
	TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4
	While the TOE does not claim NTP, the TD is acknowledged and will be applied to the SD.
	TD0529: NIT Technical Decision for and Authority Information Access extension
	This TD does not apply to the TOE as it does not claim OCSP.
	TD0530: NIT Technical Decision for FCS_TLSC_EXT.1.1 5e test clarification
	This TD applies to the TOE.
	TD0531: NIT Technical Decision for Challenge-Response for Authentication
	This TD applies to the TOE.
	TD0532: NIT Technical Decision for Use of seeds with higher entropy
	This TD applies to the TOE.
	TD0533: NIT Technical Decision for FTP_ITC.1 with signed downloads
	While the TOE does not claim signed downloads, the TD content is acknowledged.
	TD0535: NIT Technical Decision for Clarification about digital signature algorithms for FTP_TUD.1
	While the TOE does not claim signed downloads, the TD content is acknowledged.
	TD0536: NIT Technical Decision for Update Verification Inconsistency
	This TD applies to the TOE.
	TD0538: NIT Technical Decision for Outdated link to allowed-with list
	This TD applies to the TOE.

2 TOE Details

2.1 Overview

1 The Dell EMC Networking SmartFabric OS10 – a network switch – as the Target of Evaluation (TOE) for the purposes of Common Criteria (CC) evaluation. The TOE is deployed within a network to provide layer 2 and layer 3 network management and interconnectivity functionality.

2.2 TOE models

Dell EMC Networking Switch	CPU	Notes on differences
S3048-ON	Intel Atom C2338	Supported speeds (1Gbps – 100Gbps), number and type of ports and hardware formfactor.
S4048-ON		
S4048T-ON		
S4112F-ON		
S4112T-ON		
S4128F-ON		
S4128T-ON		
S4148F-ON		
S4148T-ON		
S4148U-ON		
MX5108n		
S4248FB-ON	Intel Atom C2538	
S4248FBL-ON		
S6010-ON		
Z9100-ON		
MX9116n		
S5212F-ON	Intel Atom C3538	
S5224F-ON		
S5232F-ON		

S5248F-ON	
S5296F-ON	
Z9264F-ON	
Z9332F-ON	Intel Pentium D1508

2.3   Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	Dell EMC Networking SmartFabric OS10 Security Target, v1.6, September 2020
[SUPP]	Dell EMC Networking SmartFabric OS10 Release 10.5.1 Common Criteria Guide, v1.5, September 2020.
[ADMIN]	Dell EMC SmartFabric OS10 User Guide, Release 10.5.1, 2020-03, rev. A01

2.4   Summary of SFRs

Table 4: List of SFRs

Requirement	Title
FAU_GEN.1	Audit Data Generation
FAU_GEN.2	User Identity Association
FAU_STG_EXT.1	Protected Audit Event Storage
FCS_CKM.1	Cryptographic Key Generation
FCS_CKM.2	Cryptographic Key Establishment
FCS_CKM.4	Cryptographic Key Destruction
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)
FCS_COP.1/SigGen	Cryptographic Operation (Signature Generation and Verification)
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)
FCS_RBG_EXT.1	Random Bit Generation
FCS_SSHS_EXT.1	SSH Server Protocol
FCS_TLSC_EXT.2	TLS Client Protocol with authentication

Requirement	Title
FIA_AFL.1	Authentication Failure Management
FIA_PMG_EXT.1	Password Management
FIA_UIA_EXT.1	User Identification and Authentication
FIA_UAU_EXT.2	Password-based Authentication Mechanism
FIA_UAU.7	Protected Authentication Feedback
FIA_X509_EXT.1/Rev	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FIA_X509_EXT.3	X.509 Certificate Requests
FMT_MOF.1/ManualUpdate	Management of security functions behaviour
FMT_MOF.1/Functions	Management of security functions behaviour
FMT_MTD.1/CoreData	Management of TSF Data
FMT_MTD.1/CryptoKeys	Management of TSF Data
FMT_SMF.1	Specification of Management Functions
FMT_SMR.2	Restrictions on Security Roles
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
FPT_APW_EXT.1	Protection of Administrator Passwords
FPT_TST_EXT.1	TSF testing
FPT_TUD_EXT.1	Extended: Trusted update
FPT_STM_EXT.1	Reliable Time Stamps
FTA_SSL_EXT.1	TSF-initiated Session Locking
FTA_SSL.3	TSF-initiated Termination
FTA_SSL.4	User-initiated Termination
FTA_TAB.1	Default TOE Access Banners
FTP_ITC.1	Inter-TSF trusted channel
FTP_TRP.1/Admin	Trusted Path

3 Evaluation Activities for SFRs

3.1 Security Audit (FAU)

3.1.1 FAU_GEN.1 Audit data generation

3.1.1.1 TSS

3   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:	Section [ST] 6.1.1 specifies that the TOE logs the following actions Generate SSH key-pair, Generate CSR, Import Certificate, Import CA Certificate.

4   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:	Not a distributed TOE.

3.1.1.2 Guidance Documentation

5   **TD410 -** 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:	[SUPP] Annex A lists an example of each auditable event as taken from Table 2, 4, and 5 in the NDcPP for which an SFR has been claimed.

6   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:	The evaluator performed this activity as part of those AAs associated with ensuring the corresponding guidance documentation satisfied their independent requirements. However, overall, the evaluator considered the administrator guides published by the vendor [SUPP] and [ADMIN]. The evaluator reviewed the contents of the documentation and looked specifically for functionality related to the scope of the

3.1.1.3   Tests

7   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.

8   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.

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

Note	These tests are conducted throughout the test plan.

3.1.2   FAU_GEN.2 User identity association

3.1.2.1   Tests

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

11   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

12 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:	[ST] 6.1.3. TOE can send audit logs to external syslog serve in real-time via TLS.

13 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:	[ST] 6.1.3. Log entries may be stored locally depending on the available disk space. When the local audit data store is full, the TOE will overwrite the oldest logs. Only authorized administrators can view audit records, with no ability to modify records.

14 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:	Not a distributed TOE.

15 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:	[ST] 6.1.3. When the local audit data store is full, the TOE will overwrite the oldest logs.

16 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:	[ST] 6.1.3. This is performed in real-time.

17 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:	Not a distributed TOE.

18   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:	Not a distributed TOE.

3.1.3.2   Guidance Documentation

19   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:	[SUPP] section 2.10.1 provides an appropriate pointer to the end-user into the [ADMIN] reference. Reviewing the [ADMIN] reference 'System Logging over TLS', we can see that this section describes the concepts and CLI commands needed to achieve the trusted channel. The 'crypto' commands introduce the reader to the X.509 requirements and CSR generation – all of which are used in testing FIA_X509_EXT.1/Rev. The higher-order commands responsible for configuring the logging endpoint are used heavily . They are described in [ADMIN] under 'System logging' and 'Audit log'. The documentation appears to the evaluator to be complete, based on their extensive testing efforts. No requirements are levied on the syslog server. The term 'syslog' implies an RFC compliant implementation; the fact the trusted channel is executed over mutually authenticated TLS implies a syslog server capable of such a configuration.

20   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:	[ADMIN] under 'Audit Log' indicates that the log records are saved both locally and sent to any configured syslog servers. No delay is implied.

21   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:	The TOE only claims overwrite in the [ST]. There is no management function to modify this behaviour and none was found in the administrative guidance.

3.1.3.3   Tests

22   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.

Findings:	Verification that the data is encrypted is satisfied by FTP_ITC.1 for the logging channel. The logging server used for testing was a syslog-ng v3.8.1. Due to the log-forwarding mechanism used on logging server, the audit records are therefore confirmed to have been successfully received by the audit server whenever the test cases are run.

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
Examine the last log entry and then perform a series of actions which will push the oldest record off the list and show that it is successful.
Findings: PASS

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

Findings:	This functionality is not claimed.

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.

Findings:	This functionality is not claimed.

3.2 Cryptographic Support (FCS)

3.2.1 FCS_CKM.1 Cryptographic Key Generation

3.2.1.1 TSS

23   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:	[ST] 6.2.1. RSA 2048-bit is used in TLS and SSH. ECC P-256 is used in SSH. DH group 14 is used in TLS and SSH. DH group 16 is used in SSH.

3.2.1.2 Guidance Documentation

24   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:	The set of key generation schemes and key sizes is not configurable. No additional configuration is needed beyond placing the device into FIPS mode as described in [SUPP] section 2.7.

3.2.1.3 Test

25   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

26   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.

27   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

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

29   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

30   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.

Key Generation for Finite-Field Cryptography (FFC)

31   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.

32   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

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

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

34   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 $1 <= x <= q-1$.

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

36   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.

37 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

- $g \ne 0,1$
- q divides p-1
- $g^q \bmod p = 1$
- $g^x \bmod p = y$

38 for each FFC parameter set and key pair.

Diffie-Hellman Group 14

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

Findings:	RSA keys can be constructed for the purpose of either SSH host-key identification or for building client-side TLS private keys. CAVP certificate C616 (https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/details?product=10975) shows support for RSA KeyGen (186-4).
	ECC keys can be constructed for the purpose of SSH host key identification. CAVP certificate C616 shows support for ECDSA KeyGen (186-4).

3.2.2   FCS_CKM.2 Cryptographic Key Establishment

3.2.2.1   TSS

40 **TD0482 -** 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.

41 If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall claim the TOE meets RFC 3526 Section 3.

42 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: (sample table from TD).

43 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:	[ST] key establishment schemes in 6.2.2 correspond to key generation in 6.2.1. Diffie-Hellman group 14 and 16 meets RFC 3526 section 3 by implementing the 2048-bit and 4096-bit Modular Exponential (MODP) groups.
	The TSS in [ST] section 6.2.2 does not have a well-formatted table as per the TD. However the TD does not mandate such presentation. Rather, the TD mandates that the intent is to be able to identify the scheme to the service. Since there is only one instance of FCS_TLSC_EXT.2 and one instance of FCS_SSHS_EXT.1, there is no ambiguity in the content and presentation of the material presented in section 6.2.2 as reviewed by the evaluator.

3.2.2.2 Guidance Documentation

44   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:	When operating in FIPS mode, all cryptographic parameters are restricted to those defined in the SFRs by default. No additional configuration is needed beyond placing the device into FIPS mode as described in [SUPP] section 2.7.

3.2.2.3 Tests

Key Establishment Schemes

45   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

46   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

47   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.

48   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.

49   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.

50   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.

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

Validity Test

52   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.

53   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).

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

SP800-56B Key Establishment Schemes

55   **TD0402 -** 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.

Findings:	Refer to FTP_ITC.1 which tests RSA key establishment for TLS.

Diffie-Hellman Group 14

56   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.

Findings:	RSA key exchange is performed as part of the claimed ciphersuites for TLS (FCS_TLSC_EXT.2). CAVP certificate C616 (https://csrc.nist.gov/projects/cryptographic-algorithm-validation-program/details?product=10975) shows support for RSA KeyGen (186-4), SHS, and DRBG as per the required mapping. ECC key establishment is performed for the purposes of FCS_SSHS_EXT.1. CAVP certificate C616 shows support for KAS-ECC. DH group 14 is tested as part of FTP_TRP.1/Admin and FTP_ITC.1 during TLS.

3.2.3   FCS_CKM.4 Cryptographic Key Destruction

3.2.3.1   TSS

57   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:** | [ST] Table 14 lists this information. |

58   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:** | [ST] Table 14 lists this information. |

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

| **Findings:** | [ST] Table 14 lists this information. |

60   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:** | As per [ST] table 14, all keys are stored plaintext in the specified medium. |

61   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:** | The [ST] does not identify a configuration or circumstances which does not conform to the key destruction requirement. |

62   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:** | No specification of "a value that does not contain any CSP". |

---

[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.

3.2.3.2   Guidance Documentation

63   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.

64   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:	Neither the ST nor the AGD describes circumstances where key destruction may be delayed.

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

3.2.4.1   Tests

AES-CBC Known Answer Tests

65   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.

66   **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.

67   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.

68   **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.

---

[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).

69   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.

70   **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].

71   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.

72   **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].

73   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

74   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.

75   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

76   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]
```

77   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.

78   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

79   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.

80   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.

81   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.

82   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.

TD0397]

AES-CTR Known Answer Tests

83   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. Due to the fact that 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):

84   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.

85   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.

86   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.

87   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].

88   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

89   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

90   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]
```

91   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.

Findings:	AES-CTR for 128-bit and 256-bit keys is claimed for CAVP C616.

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

3.2.5.1 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

92   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

93   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

94   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.

95   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

96   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.

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

Findings:	CAVP C616 shows support for both RSA and ECDSA SigGen (186-4) and SigVer (186-4).

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

3.2.6.1 TSS

98   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:	[ST] 6.2.6. Hashing used in TLS, SSH and for hashing passwords in non-volatile storage.

3.2.6.2 Guidance Documentation

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

Findings:	The cryptographic module's hash algorithms are not configurable. No additional configuration is needed beyond placing the device into FIPS mode as described in [SUPP] section 2.7.

3.2.6.3 Tests

100   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.

101   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

102   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

103   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

104   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 $1 \leq i \leq m$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Byte-oriented Mode

105   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 $1 \leq i \leq m/8$. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

| 106 | 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. |

| **Findings:** | CAVP C616 shows support for SHS with SHA1, SHA2-256 and SHA2-512. |

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

3.2.7.1 TSS

| 107 | 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:** | [ST] 6.2.7 Table 13 lists the correct values for the claimed HMAC functions. |

3.2.7.2 Tests

| 108 | 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. |

| **Findings:** | CAVP C616 shows support for HMAC-SHA1, HMAC-SHA2-256 and HMAC-SHA2-512. |

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

| 109 | Documentation shall be produced—and the evaluator shall perform the activities—in accordance with Appendix D of [NDcPP]. |

3.2.8.1 TSS

| 110 | 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:** | [ST] 6.2.8. The TOE employs CTR_DRBG, seeded with 256 of full entropy from Linux Kernel RNG. |

3.2.8.2 Guidance Documentation

| 111 | The evaluator shall confirm that the guidance documentation contains appropriate instructions for configuring the RNG functionality. |

| **Findings:** | The RNG is not configurable. No additional configuration is needed beyond placing the device into FIPS mode as described in [SUPP] section 2.7. |

3.2.8.3   Tests

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

113   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).

114   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.

115   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.

Findings:	CAVP C616 shows support for Counter mode DRBG using a keysize of 256 bits. Assuming the seed is full entropy, this is sufficient to generate keys of the maximum claimed size (256-bit AES keys).

3.3   Identification and Authentication (FIA)

3.3.1   FIA_AFL.1 Authentication Failure Management

3.3.1.1   TSS

116   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:	[ST] 6.3.5 specifies the following: "The TOE is capable of tracking authentication failures of remote administrators (those using SSH) by using a counter for each remote user. 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."

117   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:	[ST] 6.3.5. Local console access does not implement lockout when configured as per the guidance documentation.

3.3.1.2   Guidance Documentation

118   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:	The [SUPP] in section 2.12, points the user to the 'Security' section in the [ADMIN] guide. This section of the guide provides concepts and CLI commands suitable to configure the number of successive login attempts and the lock-out time under 'Enable user lockout'. (The [ST] only claims reactivation after a period of time.)

119   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:	[SUPP] Section 2.6(a)(ii) provides information necessary to provide a user account which is free from the influences of a remote attacker attempting to deny service to an administrator.

3.3.1.3   Tests

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

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

b.   Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows.

If the administrator action selection in FIA_AFL.1.2 is included in the ST then the evaluator shall confirm by testing that following the operational guidance

and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

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

High-Level Test Description
Set the lockout thresholds to 3 attempts and 1 minute.
Using the SSH interface, log into the TOE twice using an incorrect password. On the third attempt, log in correctly and verify that the threshold has not been reached.
Using the SSH interface, log into the TOE three times using an incorrect password. On the fourth attempt, log in correctly and verify that the threshold has been reached and that the user cannot log in.
Using a secondary workstation with a distinct IP, log into the TOE using the Web interface with the correct password. The attempt should fail.
Attempt to log into the local console using the unlockable account. The attempt should succeed.
Wait 45 seconds.
Attempt to login using the locked out username and correct password. The attempt should fail.
Wait another 1 minute for the timer to expire.
Attempt to login using the locked out username and correct password. The attempt should succeed.

Findings: PASS

3.3.2 FIA_PMG_EXT.1 Password Management

3.3.2.1 Guidance Documentation

121   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:	The [SUPP] in section 2.12, points the user to the 'Security' section in the [ADMIN] guide. From here, there is a subsection called 'User Configuration' which identifies the characters which can be used to compose a password (under 'Password Strength'), the minimum length and maximum length (under 'Role-based access control' it states "A maximum of 32 alphanumeric characters; 9 characters minimum.").
	General guidance is provided along with appropriate CLI commands to help the end-user compose appropriately strong password policies. This information is found under 'Password Strength' in the [ADMIN] guide.

3.3.2.2 Tests

122 The evaluator shall perform the following tests.

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

High-Level Test Description
Change the password length to be 15 characters. Change the password for the built-in 'admin' user using the identified CLI. Show that the password can be used to login to the SSH and local console. Change the password for the built-in 'admin' back to a known good password. Change the password length to be 9 characters. Change the password for the non-built-in 'low-priv-admin' user to be only 8 characters and show it is rejected. Show that only the new password can be used to login.
Findings: PASS

3.3.3 FIA_UIA_EXT.1 User Identification and Authentication

3.3.3.1 TSS

123 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:	[ST] 6.3.2 – 6.3.4. Either local logon over serial connection or remote administrative via SSHv2. Section 6.3.3 in the [ST] describes credential, protocol, and successful login.

124 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:	[ST] 6.3.3. No TOE administrative access is permitted until the administrator has been successfully identified and authenticated. The TOE does permit an unauthenticated user to view the login banner prior to successful identification and authentication which is consistent with the SFR in section 5. The TOE also presents the TOE major version number in the CLI login prompt prior to I&A (e.g. OS10) but does not include the specific minor/revision/build version number indicators.

125 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:	Not a distributed TOE.

126 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:	Not a distributed TOE.

3.3.3.2 Guidance Documentation

127 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:	[SUPP] section 2.6 points the user to section 'CLI Basics' in the [ADMIN] guide. This document provides information that for CLI access, usernames and passwords can be used. Additional information can be found in [ADMIN] under section 'Log In'.
	In [SUPP] section 2.6(a), for local console access, the signalling properties of the RS-232 line are described. When SSH public keys are to be used, the appropriate CLI commands are provided to provision those per-user ([SUPP] section 2.6 (b).

3.3.3.3 Tests

128 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 of the identified interfaces, do:
Log into the identified management interface using a known-good credential and logout.
Login into the identified management interface using a known-bad credential and logout.
Ensure the appropriate audit messages appear.
Findings: PASS

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
The device does not have any services configured prior to I&A other than a TOE banner.
All claimed services available to remote entities are identified as part of AVA_VAN.1 test scanning.
Findings: PASS

    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
The device does not have any services configured prior to I&A aside from a TOE banner.
All claimed services available to local entities are identified as part of AVA_VAN.1 test scanning.
Findings: PASS

    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.

Findings:	Not a distributed TOE.

3.3.4   FIA_UAU_EXT.2 Password-based Authentication Mechanism

129   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   Tests

130   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
Log into the local management interface.
Ensure the password field does not echo characters as claimed by the ST.
Findings: PASS

3.4 Security management (FMT)

3.4.1 General requirements for distributed TOEs

3.4.1.1 TSS

131 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:	Not a distributed TOE.

3.4.1.2 Guidance Documentation

132 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:	Not a distributed TOE.

3.4.1.3 Tests

133 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).

Findings:	Not a distributed TOE.

3.4.2 FMT_MOF.1/ManualUpdate

3.4.2.1 TSS

134 For distributed TOEs see chapter 3.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings:	Not a distributed TOE.

3.4.2.2 Guidance Documentation

135 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:	[SUPP] section 2.4 provides information pertaining to performing a TOE firmware update. Under 'OS10 Upgrade' in the [ADMIN] guide, the guidance indicates that the TOE will reboot which will cause all functions to cease to operate during the update procedure.

136 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:	Not a distributed TOE.

3.4.2.3 Tests

137   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.

138   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.

High-Level Test Description

Log into the CLI using an account with privileges which should not permit upgrades. Attempt to upgrade the device. The action should fail.

Findings: PASS

3.4.3   FMT_MTD.1/CoreData   Management of TSF Data

3.4.3.1 TSS

139   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:	The [ST] in section 6.4.3 specifies that users must login prior to any administrative functions being available.

140   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:	[ST] 6.4.3. Users (admin) must be logged in before being able to manage the TOE's trust store. In [ST] section 6.4.4, FMT_SMR.1 defines various roles (all of whom are flavours of 'Security Administrators') with varying levels of administrative capability. Only the secadmin (via "cryptographic key operations for secure access paths") and the sysadmin (access to everything) roles have access to the trust store.

3.4.3.2 Guidance Documentation

141   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:	The guidance documentation covers all TSF-data manipulating functions implemented in response to the SFRs.
	The TOE claims the following functions:

a) Ability to administer the TOE locally and remotely: This is covered under FIA_UIA_EXT.1;

b) Ability to configure the access banner: This is covered under FTA_TAB.1;

c) Ability to configure the session inactivity time before session termination or locking: This is covered under FTA_SSL_EXT.1 and FTA_SSL.3;

d) Ability to update the TOE, and to verify the updates using hash comparison capability prior to installing those updates: This is described under FMT_MOF.1/ManualUpdate;

e) Ability to configure the authentication failure parameters for FIA_AFL.1: This is described in FIA_AFL.1;

f) Ability to configure audit behaviour: This is described in FAU_STG_EXT.1;

g) Ability to manage the cryptographic keys: This is described in FMT_MOF.1/CryptoKeys;

h) Ability to set the time which is used for time-stamps: This is described in FPT_STM_EXT.1;

i) Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors: This is described in [SUPP] 2.11 as well as in [ADMIN] section 'X.509v3 Certificates'; and

j) Ability to import X.509v3 certificates to the TOE's trust store: This is described in [SUPP] 2.11 as well as in [ADMIN] section 'X.509v3 Certificates'.

142   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: | The TOE supports handling of X.509v3 certificates and provides a trust store. Information on loading CA certificates into the trust store are provided in [ADMIN] under 'X.509v3 Certificates'. Additional instructions to the end-user are provided in section 2.11 pertaining to verifying signature chains when loading certificates associated with Certificate Signing Requests originating from the TOE.

A trust anchor is anything in the ca-certs trust store. |

3.4.4   FMT_SMF.1 Specification of Management Functions

143   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)

144 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:	The [ST] specifies in section 6.4.6 that security functions are available over the CLI which is available locally via console and remotely via SSH.

145 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:	Not a distributed TOE.

146 **TD0408 -** 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:	The TSS in [ST] section 6.3.2 claims the local console is operated over a serial connection over RJ-45. The same section indicates the remote console is operated over SSH. The [SUPP] in section 2.6 provides guidance for both the local serial console and remote SSH CLI. No warnings are necessary since the serial console is signalled completely independent of the L2/L3 network for the remote interface.

3.4.4.2 Guidance Documentation

147 See section 3.4.4.1.

3.4.4.3 Tests

148 The evaluator tests management functions as part of testing the SFRs identified in section 4.4.4. 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 Guidance Documentation

149 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:	[SUPP] Section 2.6 'Administrative Interfaces' provides instructions for administering the TOE both locally and remotely via SSH.

3.4.5.2   Tests

150   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.

Findings:	There are no explicit test activities and therefore none are recorded here. All interfaces are tested throughout the 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

151   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:	[ST] 6.5.1. While all keys are stored in plaintext, there are no interfaces designed for the purpose of viewing these keys.

3.5.2   FPT_APW_EXT.1 Protection of Administrator Passwords

3.5.2.1   TSS

152   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:	[ST] 6.5.2. Passwords are hashed in NVRAM using SHA-256. No interface available for viewing passwords.

3.5.3   FPT_TST_EXT.1   TSF testing

3.5.3.1   TSS

153   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:	In [ST] section 6.5.3, the TSS describes low-level tests which are performed on boot. It describes the cryptographic self-tests performed at start time. It also describes

| | kernel and TSF binary file integrity checks which are performed at boot as long as secure-boot mode is enabled during initial configuration (which is required in the evaluated configuration). |

154   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:** | Not a distributed TOE. |

3.5.3.2   Guidance Documentation

155   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:** | [SUPP] Section 2.3 – Self-Tests provides an indication of the self-tests that are run and the possible errors that might result as well as the action an administrator should take in response. |

156   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:** | Not a distributed TOE. |

3.5.3.3   Tests

157   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.

158   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.

159   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.

160   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
Force a reboot of the TOE using the CLI interface. Show that there is a record that cryptographic self-tests and integrity tests run on restart.

High-Level Test Description
Findings: PASS

3.5.4   FPT_TUD_EXT.1   Trusted Update

3.5.4.1   TSS

161   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:	[ST] 6.5.4. CLI command "show version" to display information about TOE firmware. Delayed activation not specified.

162   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:	[ST] 6.5.4. The TOE uses a manual hash verification mechanism to verify update integrity, prior to initiating an upgrade.

163   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:	Automatic checking not specified.

164   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:	Not a distributed TOE.

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

Findings:	Only a hash mechanism is specified.

| 166 | 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:** | [ST] 6.5.4. The hash process is manual and must be actively performed by the administrator. In addition, the administrator is required to include the published hash in the command line when committing the installation. |

3.5.4.2   Guidance Documentation

| 167 | 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:** | In [ADMIN] section 'OS10 Upgrade', the document provides the 'show version' command to query the currently active version. The TOE does not claim delayed activation, but it is possible to query the OS loaded into the standby partition using 'show boot'. |

| 168 | 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:** | In [SUPP] section 2.2, there are instructions to verify the downloaded firmware image. The 'verify' command corresponds to the same command provided in the [ST] in section 6.5.4. |

| 169 | 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:** | A published hash is used to protect the trusted update mechanism. This trusted hash is obtained from Dell Networking according to section 2.2 of [SUPP]. |

| 170 | 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:** | Not a distributed TOE. |

| 171 | 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:	Not a distributed TOE.

172   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:	Certificate-based mechanisms are not claimed for this TOE.

3.5.4.3   Tests

173   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
Get the current version of the TOE.
Attempt to install a legitimate version of the TOE.
After the install, get the current version of the TOE and ensure it is consistent with the newly installed version.
Findings: PASS

b. **TD0477 -** 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.

Not Applicable	The TOE only uses published hashes.

c.   **TD0477 -** 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)   **TD0477 -** 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. 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

High-Level Test Description
Get the current version of the TOE.
Attempt to install a legitimate version of the TOE without a sha256 hash value provided.
Attempt to install a legitimate version of the TOE with an invalid sha256 hash value.
Ensure both attempts fail.
Findings: PASS

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.

Test Not Applicable	The TOE does not claim delayed activation.

174   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.

175   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).

Note:	The TOE only supports manual updates.

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

Test Not Applicable	The TOE does not claim delayed activation.

3.5.5 FPT_STM_EXT.1 Reliable Time Stamps

3.5.5.1 TSS

177   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:	[ST] 6.5.5. Internal time clock used, configured by an administrator upon initial configuration, and can be manually changed by an administrator thereafter.

3.5.5.2 Guidance Documentation

178   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:	[SUPP] section 2.9 informs the administrator to refer to [ADMIN] under section 'System Clock'. That document section provides the 'clock set' function to set the clock and 'clock' to view the current clock. Time zones can be set using 'clock timezone' where necessary.
	The TOE does not claim support for NTP and NTP is disabled in the evaluated configuration.

3.5.5.3 Tests

179   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
Get the current date and time. Change the date/time in the past by 1 day, 1 hour and 42 minutes. Verify the date/time was set properly.
Change the date/time in the future by 7 days, 1 hour and 42 minutes. Verify the date/time was set properly.
Findings: PASS

     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 TOE does not claim NTP.

180	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.

Findings:	The TOE does not have independent time information.

3.6 TOE Access (FTA)

3.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.6.1.1 Guidance Documentation

181	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:	[SUPP] Section 2.6 – session termination on idle timeout is supported on the local interface.

3.6.1.2 Tests

182	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
For each of 1, 3, 5 minutes:
Change the idle timeout to this value;
Log into the device;
Wait for the full duration of the timeout. The session should terminate.
Findings: PASS

3.6.2 FTA_SSL.3 TSF-initiated Termination

3.6.2.1 Guidance Documentation

183	**TD0425 -** The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Findings:	[SUPP] Section 2.6 – session termination on idle timeout is supported on the remote interface.

3.6.2.2   Tests

184   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
For each of 1, 3, 5 minutes:
Change the idle timeout to this value;
Log into the device;
Wait for the full duration of the timeout. The session should terminate.
Findings: PASS

3.6.3   FTA_SSL.4 User-initiated Termination

3.6.3.1   Guidance Documentation

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

Findings:	[SUPP] section 2.14 describes that the 'exit' command terminates the CLI session. Note that 'end' and 'exit' can be used to navigate the user through the command hierarchy as well, as described under their CLI command headings (under 'Common OS10 commands' in the [ADMIN] guide). However, when the user is in EXEC mode (the highest command level), typing 'exit' will terminate the session.

3.6.3.2   Tests

186   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
Log out using the TSFI previous discussed.
Verify that the session has been terminated.
Findings: PASS

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.
Log out using the TSFI previous discussed.
Findings: PASS

3.6.4 FTA_TAB.1 Default TOE Access Banners

3.6.4.1 TSS

187   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:	[ST] 6.3 details a serial connection for local administration, and remote administration over SSHv2. This is consistent with [ST] section 2. Section 6.6.4 in the [ST] describes TOE access banner presentation at the CLI – the only administrative interface available.

3.6.4.2 Guidance Documentation

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

Findings:	[SUPP] section 2.6(a)(i) points the administrator to the [ADMIN] guide under 'Login banner' which provides the necessary concepts and CLI commands to set the login banner for both the local and remote CLI interfaces.

3.6.4.3 Tests

189   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
Log into the SSH CLI interface.
Change the banner to a random string.
Log into fresh sessions for all interactive interfaces and show that the banner was modified and is presented prior to I&A.
Findings: PASS

3.7 Trusted path/channels (FTP)

3.7.1 FTP_ITC.1 Inter-TSF trusted channel

3.7.1.1 TSS

190 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:	[ST] 6.7.1. The only authorized IT entity identified is an external (syslog) audit server. As such, the TOE acts a server or client (per SFR section 5), communicating over TLS.

3.7.1.2 Guidance Documentation

191 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:	Section 2.10 of [SUPP] provides the pointers to the instructions to configure the TOE to use TLS for securely transmitting audit records to a remote endpoint. No instructions are provided to recover an unintentionally broken connection since the connection will retry automatically once the connection is re-established.

3.7.1.3 Tests

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

193 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 only trusted channel is the remote audit log, 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.

High-Level Test Description
Engage wireshark over the appropriate interface.

High-Level Test Description
Log into the CLI and disable and re-enable the logging interface.
Examine wireshark and verify that the log interface sends a Client Hello TLS message.
Findings: PASS

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
Engage wireshark over the appropriate interface to capture log message traffic.
Log into the serial device and logout.
Examine wireshark and verify that the log interface sends encrypted traffic to the remote logging server IP endpoint.
Findings: PASS

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 MAC 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
Engage wireshark over the logging interface.
Perform a looping login activity once per second to ensure logging messages are being tested.
Physically disconnect the remote logging server (disconnect from the remote end rather that from the TOE end to ensure that the TOE is unable to invoke any layer 2 carrier-sensing mechanism).
Wait 5 seconds.
Physically reconnect the remote logging server.
Examine wireshark and verify that the log interface continues to send encrypted Application Data packets.

High-Level Test Description
Repeat the above with a 15 minute disconnection and show that the TOE logs the fact that the logging interface is down. Once it is reconnected, show that the system automatically reconnects and traffic is protected using Application Data packets.
Findings: PASS

Further assurance activities are associated with the specific protocols.

194   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.

Findings:	Not a distributed TOE.

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

Findings:	Information was obtained to be able to complete the test cases above.

3.7.2   FTP_TRP.1/Admin Trusted Path

3.7.2.1   TSS

196   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:	[ST] 6.7.2. CLI over SSH for remote administration.

3.7.2.2   Guidance Documentation

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

Findings:	[SUPP] Section 2.6 – remote administration using SSH.

3.7.2.3   Tests

198   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 only trusted path is the SSH, which is set up as per the evaluated configuration. It is constantly tested throughout the evaluation.

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

High-Level Test Description
Engage wireshark over the appropriate interface.
Log into the trusted path.
Examine wireshark and verify that the trusted path sends encrypted traffic after any initial plaintext protocol negotiation occurs.
Findings: PASS

199   Further assurance activities are associated with the specific protocols.

200   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.

Findings:	Not a distributed TOE.

4   Evaluation Activities for Optional Requirements

201   No optional requirements have been claimed.

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

202   The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication and that this list conforms to FCS_SSHS_EXT.1.5. and ensure that if password-based authentication methods have been selected in the ST then these are also described.

Findings:	[ST] 6.2.9 describes SSH-RSA authentication, as specified in FCS_SSHS_EXT.1.5, and confirms that password or public key SSH-RSA authentication is supported.

FCS_SSHS_EXT.1.3

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

Findings:	[ST] 6.2.9. Packets over 256 KB are automatically dropped.

FCS_SSHS_EXT.1.4

204   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:	[ST] 6.2.9. Optional characteristics are specified and consistent with SFR selections, as are encryption algorithms. The TOE utilises AES-CTR-128 and AES-CTR-256 for SSH encryption.

FCS_SSHS_EXT.1.5

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

Findings:	[ST] 6.2.9. Optional characteristics are specified and consistent with SFR selections, as are encryption algorithms. The TOE supports ssh-rsa, rsa-sha2-256, rsa-sha2-512 public key mechanisms and signature algorithms.

FCS_SSHS_EXT.1.6

206   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:	[ST] 6.2.9. Data integrity algorithms listed and consistent with SFR selections. The TOE provides data integrity for SSH connections via HMAC-SHA1, HMAC-SHA2-256 and HMAC-SHA2-512

FCS_SSHS_EXT.1.7

207   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:	[ST] 6.2.9. The TOE supports diffie-hellman-group14-sha1 and diffie-hellman-group16-sha512 for SSH key exchanges.

FCS_SSHS_EXT.1.8

208   The evaluator shall check that the TSS specifies the following:

      1. Both thresholds are checked by the TOE.

      2. Rekeying is performed upon reaching the threshold that is hit first.

Findings:	[ST] 6.2.9. The TOE will re-key SSH connections after 1 hour of after an aggregate of 1 gig of data has been exchanged (whichever occurs first).

5.1.1.2   Guidance Documentation

FCS_SSHS_EXT.1.4

209   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:	SSH ciphers can be configured as per [SUPP] section 2.6(b)(i).

FCS_SSHS_EXT.1.5

210   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:	The TOE can be enabled to permit public key authentication as per [SUPP] section 2.6(b)(ii). However, beyond enabling the functionality, the specified public key authentication algorithms are not configurable.

FCS_SSHS_EXT.1.6

211   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:	SSH integrity algorithms can be configured as per [SUPP] section 2.6(b)(i).

FCS_SSHS_EXT.1.7

212   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:	SSH key exchange algorithms are not configurable.

FCS_SSHS_EXT.1.8

213   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:	SSH rekey limits are not configurable.

5.1.1.3   Tests

FCS_SSHS_EXT.1.2

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

Note	This test was conducted as part of FIA_UIA_EXT.1/FIA_UAU_EXT.2.

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

Note	This test was conducted as part of FIA_UIA_EXT.1/FIA_UAU_EXT.2.

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

FCS_SSHS_EXT.1.3

217   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

FCS_SSHS_EXT.1.4

218   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
Using SSH client, log into the TOE using each of the claimed ciphers in turn and show that the communication is successful. Review the negotiation line from the server to ensure that there are no additional ciphers claimed by the implementation that differ from the ST or the PP requirements.
Findings: PASS

FCS_SSHS_EXT.1.5

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

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

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

     TD0412 - Test objective: The purpose of this negative test is to verify that the server rejects authentication attempts of clients that present a public key that does not match public key(s) associated by the TOE with the identity of the client (i.e. the public keys are unknown to the server). To demonstrate correct functionality it is sufficient to determine that an SSH connection was not established after using a valid username and an unknown key of supported type.

High-Level Test Description
Create two public/private key pairs. Load the public key portion from pair A into the TOE. Using SSH client, log into the TOE using private key from pair B. The attempt should fail.
Findings: PASS

221   Test 3: The evaluator shall configure an SSH client to only allow a public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the SSH client to the TOE and observe that the connection is rejected.

High-Level Test Description
Create a public/private key pair for DSA unsupported algorithms. Load the public key portion from the newly generated key into the TOE for the admin user. The attempt to load may fail. Using SSH client, log into the TOE using newly generated private key portion. The attempt should fail.
Findings: PASS

FCS_SSHS_EXT.1.6

222   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.

223   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 or the PP requirements.
Findings: PASS

224   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.

225   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

FCS_SSHS_EXT.1.7

226   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

227   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 algorithm and show that the communication is successful.
Findings: PASS

FCS_SSHS_EXT.1.8

228   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.

229   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).

230   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 push at least 1GB of data in less than 1 hour to force rekeying.
Findings: PASS

231   **TD0475 -** 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).

232   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).

233   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, log into the TOE and push less than the rekey limit of data in at least 1 hour to force rekeying by time-based mechanisms.
Findings: PASS

234   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	These limits are not configurable for this TOE.

235   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.

Findings:	The TOE does not have hardware limitations.

5.1.2   FCS_TLSC_EXT.2 Extended: TLS Client Protocol with authentication

5.1.2.1   TSS

FCS_TLSC_EXT.2.1

236   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:	[ST] 6.2.10. The ciphersuites are specified and are the same as those listed in the component.

FCS_TLSC_EXT.2.2

237   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. The evaluator shall ensure that this description identifies if certificate pinning is supported or used by the TOE and how it is implemented.

Findings:	[ST] 6.2.10. The reference identifier for the Audit Server is configured by the administrator using the web GUI. The reference identifiers must be an IP address. Wildcards are not supported in IPv4 addresses and therefore they are, de facto, not supported (with confirmation by testing). The TLS client does not support certificate pinning.

238   **TD0481**— 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 (RFC5952 for IPv6, RFC 3986 for IPv4) is enforced.

Findings:	[ST] 6.2.10. The TOE converts IPv4 addresses in the CN to binary format and stores them an array in network byte order. The TOE enforces the RFC 3986 IPv4 canonical format.

FCS_TLSC_EXT.2.4

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

Findings:	Per SFR selection, the TSF does not present the Supported Elliptic Curves Extension.

FCS_TLSC_EXT.2.5

240   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:	In section 6.2.9 of the [ST], the TLS client supports mutual authentication and is capable of presenting a certificate to the server for authentication.

5.1.2.2   Guidance Documentation

FCS_TLSC_EXT.2.1

241   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:	While the [SUPP] in section 2.11 states that audit logging must be configured to operate over a TLS connection, the actual ciphersuites are not configurable.

FCS_TLSC_EXT.2.2

242   **TD0481 -** 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:	[SUPP] section 2.11 indicates that the only permitted identifier type in the evaluated configuration is an IPv4 address. The same section informs the user that this IPv4 address can be in the CN or the SAN. [ADMIN] 'System logging over TLS' provides information as to how to configure the reference identifier (logging server …). The TOE includes supports for IP addresses, but no warning or CA policy recommendations are provided in the guidance documents as the TOE can securely make use of IP addresses. Network organization, such as IP address assignment of trusted IT entities is outside of the scope of the NDcPP objectives for the TOE environment. As per the [ST] section 6.2.10: the TOE converts IPv4 addresses in the CN to binary format and stores them [in] an array in network byte order. Wildcards are not permitted in IPv4 addresses in the Common Name field.

FCS_TLSC_EXT.2.4

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

Findings:	N/A - The TSF does not present the supported elliptic curves extension.

FCS_TLSC_EXT.2.5

244   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:	In [SUPP] section 2.10.1, the document points to [ADMIN] section 'System Logging over TLS' which describes the requirement that the TOE have a client-side certificate ready for use. The [ADMIN] provides the necessary concepts and CLI commands to perform these operations. These commands were tested.

5.1.2.3   Tests

FCS_TLSC_EXT.2.1

245   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 a specifically claimed ciphersuite.
Findings: PASS

246   *The goal of the following test is to verify that the TOE accepts only certificates with appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation.*

247   **TD0396 -** 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 extension and verify that a connection is established. The evaluator shall repeat this test using a different, but otherwise valid and trusted, certificate that lacks the Server Authentication purpose in the extendedKeyUsage extension and ensure that a connection is not established. Ideally, the two certificates should be similar in structure, the types of identifiers used, and the chain of trust.

High-Level Test Description
Construct two X.509 certificates: one with an extendedKeyUsage with 'serverAuth' and another without. 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

248 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.
Findings: PASS

249 Test 4: The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the client denies the connection. Test 2 in FCS_TLSS_EXT.1.1 or FCS_TLSS_EXT.2.1 can be used as a substitute for this test.

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).
Findings: PASS

250 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 (for example 1.5 represented by the two bytes 03 06) 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 an incorrect TLS version.
Findings: PASS

  b. 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.
Findings: PASS

  c. 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 non-negotiated ciphersuite.
Findings: PASS

d. If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the client rejects the connection after receiving the Server Key Exchange message. 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.
Findings: PASS

e. **TD0530 -** Modify a byte in the Server Finished handshake message, and verify that the handshake is not finished 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.
Findings: PASS

f. Send a garbled message from the Server after the Server has issued the ChangeCipherSpec message 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 sending a mangled finished message.
Findings: PASS

FCS_TLSC_EXT.2.2

251 **TD0481 -** Note that where a (D)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.

252 **TD0481 -** 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.

253 The evaluator shall configure the reference identifier according to the AGD guidance and perform the following tests during a (D)TLS connection:

254 **TD0481 -** Test 1: 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.

255 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
Force the TOE client to attempt a handshake with an OpenSSL s_server sub-application sending X.509 certificates that have the characteristics required by the test.
Findings: PASS

256 **TD0481 -** Test 2: 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
Force the TOE client to attempt a handshake with an OpenSSL s_server sub-application sending X.509 certificates that have the characteristics required by the test.
Findings: PASS

257 **TD0481 -** 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
Force the TOE client to attempt a handshake with an OpenSSL s_server sub-application sending X.509 certificates that have the characteristics required by the test.
Findings: PASS

258   **TD0481 -** Test 4: 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
Force the TOE client to attempt a handshake with an OpenSSL s_server sub-application sending X.509 certificates that have the characteristics required by the test.
Findings: PASS

259   **TD0481 -** Test 5: 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. 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.

2. 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.)

Note	The TOE does not claim DNS type reference identifiers hence this test is trivially successful.

260   **TD0481 -** Test 6: [conditional] If URI or Service name reference identifiers are supported, the evaluator shall configure the DNS name and the service identifier. The evaluator shall present a server certificate containing the correct DNS name and service identifier in the URIName or SRVName fields of the SAN and verify that the connection succeeds. The evaluator shall repeat this test with the wrong service identifier (but correct DNS name) and verify that the connection fails.

Note	The TOE does not support URL or SrvName reference identifiers.

261   **TD0481 -** Test 7: [conditional] If pinned certificates are supported the evaluator shall present a certificate that does not match the pinned certificate and verify that the connection fails.

Note	The TOE does not support pinned certificates.

262   **TD0481 -** Test 8: [conditional] If IP addresses are supported, 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 an asterisk (*) (e.g. CN=192.168.1.* when connecting to 192.168.1.20, 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 8.

High-Level Test Description
Force the TOE client to attempt a handshake with an OpenSSL s_server sub-application sending X.509 certificates that have the characteristics required by the test.
Findings: PASS

FCS_TLSC_EXT.2.3

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

264   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.

265   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.

266   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.

Note	This test is not applicable for this TOE.

FCS_TLSC_EXT.2.4

267   Test 1: If using DHE or ECDH, the evaluator shall configure the server to perform an ECDHE key exchange in the TLS connection using a non-supported curve (for example P-192) and shall verify that the TOE disconnects after receiving the server's Key Exchange handshake message.

Note	The TOE does not support Elliptic Curves Extension in the client hello.

FCS_TLSC_EXT.2.5

268   The purpose of these tests is to confirm that the TOE appropriately handles connection to peer servers that support and do not support mutual authentication.

269   Test 1: The evaluator shall establish a connection to a peer server that is not configured for mutual authentication (i.e. does not send Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE did not send Client's Certificate message (type 11) during handshake.

High-Level Test Description
Configure a test TLS server to operate without mutual authentication and show that the TOE does not send back a certificate.
Findings: PASS

270   Test 2: The evaluator shall establish a connection to a peer server with a shared trusted root that is configured for mutual authentication (i.e. it sends Server's Certificate Request (type 13) message). The evaluator observes negotiation of a TLS channel and confirms that the TOE responds with a non-empty Client's Certificate message (type 11) and Certificate Verify (type 15) messages.

High-Level Test Description
Configure a test TLS server to operate with mutual authentication and show that the TOE sends back a certificate.
Findings: PASS

5.2   Identification and Authentication (FIA)

5.2.1   FIA_X509_EXT.1/Rev   X.509 Certificate Validation

5.2.1.1   TSS

271   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:	[ST] 6.3.6 describes certificate validity checks at TOE TLS client validation of server X.509 certificates, and when certificates are loaded into the TOE. As X.509 certificates are not used for trusted updates, firmware integrity self-tests or client authentication, the code-signing and clientAuthentication purpose is not checked in the extendedKeyUsage for related certificates.

272   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:	[ST] 6.3.6. Several points where certificates are checked for validation are detailed in 6.3.6, and revocation performed using CRLs.

273   It is expected that revocation checking is performed when a certificate is used in an authentication step. It is expected that revocation checking is performed on both leaf and intermediate CA certificates when a leaf certificate is presented to the TOE as part of the certificate chain during authentication. Revocation checking of any CA certificate designated a trust anchor is not required. It is not sufficient to perform a revocation check of a CA certificate only when it is loaded onto the device.

Findings:	In section 6.3.6 of the [ST], the TOE checks certificates when they are consumed by the TLS client as well as when they are loaded into the TOE.

5.2.1.2   Tests

274   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
Create a sequence of three X.509 certificates: a root CA, an intermediate CA signed by the root CA and a leaf node certificate signed by the intermediate CA. Load the root CA into the TOE trust store. Force the TOE to connect to a TLS server such that the Intermediate CA is not delivered. Show that the connection is not accepted. Force the TOE to connect to a TLS server that sends back the Intermediate CA as well as the leaf and show that the connection is accepted.
Findings: PASS

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

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.

High-Level Test Description
Load the CA into the TOE trust store. Ensure the CRLs are empty. Revoke the intermediate CA and place into the CRL and load the CRL into the TOE. Verify the connection now fails due to the certificate being revoked. Unrevoke the intermediate. Revoke the server certificate and place into the CRL and load into the TOE. Verify the connection now fails due to the server certificate being revoked.
Findings: PASS

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.

High-Level Test Description
Load a CA into the trust store that is missing the CRLSigning purpose.
Load the CRL for the corresponding CA. Show that the TOE prevents loading this CRL.
Findings: PASS

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.
Findings: PASS

f. Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

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.
Findings: PASS

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

275   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.

276   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).

277   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
Attempt to load a CA certificate and missing the basicConstraints extension. Verify it fails to load.
Findings: PASS

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
Attempt to load a CA certificate having a basicConstraints extension with a FALSE CA value. Verify it fails to load.
Findings: PASS

278   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).

Findings:	The only use of X.509 certificates is for the syslog channel.

5.2.2   FIA_X509_EXT.2 X.509 Certificate Authentication

5.2.2.1   TSS

279   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:	In section 6.3.7 of the [ST], the TSS indicates there is a trust store. The trust store is used to store root CA and intermediate CA certificates. The algorithm for how the

trust store is searched in order to validate the certificate chain is provided in section 6.3.6 of the [ST]. Since there is only one trust store for the TOE, there are no explicit instructions needed to determine which trusted set of certificates are to be used. The Guidance documentation in [SUPP] section 2.11 provides introductory material for the X.509v3 trust store in the TOE. This information is (heavily) expanded upon in [ADMIN] section 'X.509v3 Certificates'.

280   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:	In section 6.3.7 of the [ST], it states that if the CRL was fetched, but then cannot be fetched again, the last known state of the certificate is used until the CRL can be fetched again.

5.2.2.2   Tests

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

282   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

5.2.3   FIA_X509_EXT.3 Extended: X509 Certificate Requests

5.2.3.1   TSS

283   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:	Device-specific information not selected.

5.2.3.2   Guidance Documentation

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:	In [SUPP] section 2.11, the document points to [ADMIN] section 'X.509v3 Certificates' which provides the necessary concepts and instructions to build a CSR using the 'crypto cert generate …' function as described under subsection 'Request and install host certificates'.
	The ST author claimed public key, Common Name, Organization, Organizational Unit and Country as the RDNs which the TOE can provide. In reality, the TOE can provide additional fields as described in the above noted sections in the [ADMIN] guide.

5.2.3.3   Tests

284   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 to an external CA entity for signing. Using OpenSSL, verify that the information in the CSR is as expected.
Findings: PASS

    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

5.3   Security management (FMT)

5.3.1   FMT_MOF.1/Functions   Management of security functions behaviour

5.3.1.1   TSS

285   For distributed TOEs see chapter 3.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings:	Not a distributed TOE.

5.3.1.2 Tests

286   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.

High-Level Test Description
For each of the defined TSFI functions found in the TOE, attempt to change them one at a time to one of their legal values and show that the change is not permitted.
Findings: PASS

287   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.

288   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.

High-Level Test Description
For each of the defined TSFI functions found in the TOE, attempt to change them one at a time using the privileged 'admin' user to one of their legal values and show that the change is permitted. Verify the effect of the change.
Findings: PASS

289   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.

Note:	The TOE does not claim this functionality and this test will not be conducted.

290	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.
291	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.

Note:	The TOE does not claim this functionality and this test will not be conducted.

292	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.

Note:	The TOE does not claim this functionality and this test will not be conducted.

293	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.
294	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.

Note:	The TOE does not claim this functionality and this test will not be conducted.

295	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.

Note:	The TOE does not claim this functionality and this test will not be conducted.

296   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.

Note:	The TOE 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

297   For distributed TOEs see chapter 3.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings:	Not a distributed TOE.

5.3.2.2   Tests

298   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.

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

High-Level Test Description
As the unprivileged user, attempt to generate an SSH private key and show it cannot succeed.
As the privileged user, attempt to generate an SSH private key and show it does succeed.
As an unprivileged user, attempt to generate a private key for use in TLS X.509 and show it cannot succeed.
Creation of private keys for use in X.509 certificates is shown in FIA_X509_EXT.3.
Findings: PASS

6   Vulnerability Assessment

300   5.6.1.1 Evaluation Activity: 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.

301   The developer shall provide documentation identifying the list of software and hardware components[3] that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside the TOE) such as a web server and protocol or cryptographic libraries. This additional documentation is merely a list of the name and version number of the components, and will be used by the evaluators in formulating hypotheses during their analysis.

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

302   5.6.1.2 Evaluation Activity: 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:	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 and Notices: https://www.dell.com/support/security/en-ca/ 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 OpenSSL Vulnerabilities: https://www.openssl.org/news/vulnerabilities.html Google

---

[3] In this sub-section the term "components" refers to parts that make up the TOE. It is therefore distinguished from the term "distributed TOE components", which refers to the parts of a TOE that are present in one physical part of a distributed TOE. Each distributed TOE component will therefore generally include a number of the hardware and software components that are referred to in this sub-section: for example, each distributed TOE component will generally include hardware components such as processors and software components such as an operating system and libraries.

Type 1 Hypothesis searches were conducted on July 22, 2020 and included the following search terms (version details have been removed in this public document):

- Dell OS10

- Debian operating system

- OpenSSL

- Pkix-ssh

- Netconf

- Syslog-ng

- Auditd

- PAM

- YumaPro

- Dell OpenSSL Cryptographic Library v2.5

- TCP

- SSH v2

The evaluation team determined that no residual vulnerabilities exploitable by attackers exist with Basic Attack Potential.

There are no type-2 hypotheses identified for the NDcPP.

The evaluation team developed Type 3 flaw hypotheses in accordance with Sections A.1.3, A.1.4, 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 Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.