Keysight Technologies

Vision Series Network Packet Broker v5.7.1

Assurance Activity Report

Version 1.3

March 2022

Document prepared by

www.lightshipsec.com

Table of Contents

1 Introduction

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

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	Canadian Common Criteria Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	Keysight Technologies
TOE	Vision Series Network Packet Broker v5.7.1 Build 5.7.1.18
Security Target	Vision Series Network Packet Broker v5.7.1 Security Target, v1.5
Protection Profile	collaborative Protection Profile for Network Devices, v2.2E (NDcPP), 23-March-2020

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, v2.2 (NDcPP-SD)
Interpretations	**NDcPP v2.2e**
	TD0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1) *This TD applies to the TOE.*
	TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4 *This TD applies to the TOE.*
	TD0536: NIT Technical Decision for Update Verification Inconsistency *This TD applies to the TOE.*

TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3 *This TD applies to the TOE.*
TD0538: NIT Technical Decision for Outdated link to allowed-with list *This TD does not apply to the TOE as this is for Protection Profile.*
TD0546: NIT Technical Decision for DTLS - clarification of Application Note 63 *This TD does not apply to the TOE as FCS_DTLSC_EXT.1.1 is not claimed.*
TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN *This TD applies to the TOE.*
TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test *This TD applies to the TOE.*
TD0556: NIT Technical Decision for RFC 5077 question *This TD applies to the TOE.*
TD0563:NiT Technical Decision for Clarification of audit date information *This TD applies to the TOE.*
TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria *This TD applies to the TOE.*
TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7 *This TD applies to the TOE.*
TD0570: NiT Technical Decision for Clarification about FIA_AFL.1 *This TD applies to the TOE.*
TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1 *This TD applies to the TOE.*
TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers *This TD applies to the TOE.*
TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e *This TD applies to the TOE.*
TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3 *This TD applies to the TOE.*

	TD0591: NIT Technical Decision for Virtual TOEs and hypervisors *This TD does not apply to the TOE as it is not a Virtual TOE.* TD0592: NIT Technical Decision for Local Storage of Audit Records *This TD applies to the TOE.*
Tools	Refer to the Test Plan.

1.3 Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	Vision Series Network Packet Broker v5.7.1 Security Target, v1.5
[SUPP]	Vision Series Network Packet Broker v5.7.1 Common Criteria Guide, v1.2
[USER]	TradeVision Network Packet Broker Release 5.7.1 User Guide 202103050910-06:00 913-2817-01 Rev A Vision 7300/7303 Network Packet Broker Release 5.7.1 User Guide 202103051339-06:00 913-2811-01 Rev A Vision Edge 10S Network Packet Broker Release 5.7.1 User Guide 202103050913-06:00 913-2816-01 Rev A Vision Edge 40/100 Network Packet Broker Release 5.7.1 User Guide 202103051339-06:00 913-2813-01 Rev A Vision ONE Network Packet Broker Release 5.7.1 User Guide 202103051342-06:00 913-2812-01 Rev A Vision X Network Packet Broker Release 5.7.1 User Guide 202103051355-06:00 913-2810-01 Rev A
[PP]	collaborative Protection Profile for Network Devices, v2.2E (NDcPP), 23-March-2020
[SD]	Evaluation Activities for Network Device cPP, v2.2 (NDcPP-SD)
[ED]	Vision Series Network Packet Broker Entropy Description, v1.3, September 2021
[INSTALL]	Vision ONE Network Packet Broker Installation Guide 913-2419-01 Rev F TradeVision Network Packet Broker Installation Guide 913-2421-01 Rev C Vision Edge 40/100 Network Packet Broker Installation Guide 913-2450-01 Rev D Vision Edge 10S Network Packet Broker Installation Guide 913-2529-01 Rev D

Ref	Document
	Vision 7300/7303 Network Packet Broker Installation Guide 913-2530-01 Rev D Vision X Network Packet Broker Installation Guide 913-2542-01 Rev D Ixia Vision 7300/7303 Startup Guide 913-2413-01 Rev B Vision Edge 10S Startup Guide 913-2414-01 Rev C Ixia Vision Edge E40/E100 Startup Guide 913-2415-01 Rev C Vision ONE Startup Guide 913-2416-01 Rev D Vision X Quick Start Guide Digital 913-2499-01 Rev E TradeVision Release 5.7.1 Quick Start Guide, 913-2818-01 Rev A
[WEB_API]	Keysight TradeVision Series Web API User Guide Release 5.7.1 913-2804-01 Rev A Keysight 7300 Series Web API User Guide Release 5.7.1 913-2799-01 Rev A Keysight Vision E10S Web API User Guide Release 5.7.1 913-2805-01 Rev A Keysight E40 Series Web API User Guide Release 5.7.1 913-2801-01 Rev A Keysight E100 Series Web API User Guide Release 5.7.1 913-2802-01 Rev A Keysight Vision ONE Series Web API User Guide Release 5.7.1 913-2800-01 Rev A Keysight Vision X Series Web API User Guide Release 5.7.1 913-2803-01 Rev A
[IND]	Keysight Technologies Vision Series Network Packet Broker v5.7.1 NDcPP 2.2E Test Plan, v1.1
[AVA]	Keysight Technologies Vision Series Network Packet Broker v5.7.1 NDcPP 2.2E Vulnerability Assessment, v1.7
[CBG4]	Canadian Common Criteria Scheme Guidance for Evaluators Guide #4, version 3.5, September 2018

2 Evaluation Activities for SFRs

2.1 Security Audit (FAU)

2.1.1 FAU_GEN.1 Audit data generation

2.1.1.1 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:	[ST] / TSS section 6.1.1 indicates that the following information is logged (both the actions and key references) as a result of the Security Administrator generating/importing or deleting cryptographic keys: - Importing Certificate - Importing CA Certificate - Generating a CSR (which implicitly includes a private key).

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:	The TOE is not a distributed TOE.

2.1.1.2 Guidance Documentation

5 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:	Annex A of [SUPP] shows an example of each auditable event required by FAU_GEN.1 (including each mandatory, optional or selection-based SFR in the ST, if there is an auditable event associated with the SFR).

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 made his determination of the administrative actions, by using FMT SFRs, FIA_PMG_EXT.1, FPT SFRs, and FTA SFRs, as a guide. The evaluator navigated through the [USER] guides for administrative actions related to those functional requirements, specifically, configuration changes related to TSF data and TSF. This includes the following sections in the user guide: - Connecting using the Craft Port Interface - Government Security Configuration Guide - Viewing and Changing System Settings - Managing Users - Authentication, Authorization, and Accounting (AAA) - About Local Syslog Viewer - Syslog Support - Software Upgrade/Downgrade and Cold Spare Upgrade Procedures - NPB Syslog Messages The evaluator also performed the administrative operations, involving changes to TSF data and TSF behaviour, as part of NDcPP v2.2 Test assurance activities, and confirmed that the documents provide sufficient information and instructions to enable evaluator to conduct NDcPP test requirements successfully.

2.1.1.3 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.

Findings:	These tests are conducted throughout the test plan. The TOE is not a distributed TOE.

2.1.2 FAU_GEN.2 User identity association

2.1.2.1 TSS & Guidance Documentation

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

2.1.2.2 Tests

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

12 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.

Findings:	The TOE is not a distributed TOE.

2.1.3 FAU_STG_EXT.1 Protected audit event storage

2.1.3.1 TSS

13 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:	This information was found in [ST] / TSS section 6.1.3 and states that audit data is transferred to a Syslog server and log events are sent in real-time over TLS.

14 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] / TSS section 6.1.3 indicates that the TOE stores local audit data in 5 1MB rotating log files. When local audit data store becomes full, the oldest audit record is deleted. Only authorized administrators may view audit records and no capability to modify the audit records is provided.

15 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:	The TOE is not a distributed TOE. [ST] / TSS 6.1.3 indicates that audit records can be stored locally in the audit store or audit data is transferred to a Syslog server via TLS.

16 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] / TSS section 6.1.3 states that when the local audit data store is full, the TOE will delete the oldest audit record.

17 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 acceptable frequency for the transfer of audit data.

Findings:	[ST] / TSS section 6.1.3 states that the log events are sent in real-time.

18 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:	The TOE is not a distributed TOE.

19 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:	The TOE is not a distributed TOE.

2.1.3.2 Guidance Documentation

20 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:	The evaluator checked [USER] and determined that the section “Syslog Support” describes configuration needed on the TOE to communicate with the audit server, with steps including: - Enabling TLS Encryption - Adding or Modifying External Syslog Servers - Generating and Uploading a Client Certificate and Uploading a Server Trusted Root Certificate (To use the encryption and mutual authentication capability that is provided by TLS) - Confirming connections to external syslog servers As stated in the same section of the [USER] as a requirement, TLS encryption can only be enabled on a syslog server configured with a DNS name.

21 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:	The evaluator checked [USER] and determined that the section “How local syslog files work - appending and overwriting files” describes information on the local audit data. The TOE keeps up to five syslog files. Each of the five syslog files contains up to 1MB of data. The TOE overwrites the oldest syslog file with the newest syslog message when full. Syslog messages are created and sent to each external syslog server configured whenever a new configuration or state changes occurs on the system.

22 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 behaviour stated under FAU_STG_EXT.1.3 is not a configurable option. The behaviour described in the TSS is consistent with the information in the [USER] section “How local syslog files work - appending and overwriting files”.

2.1.3.3 Tests

23 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 is a **syslog-ng v3.8.1** as described in the Test Setup. 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
Export audit logs to .csv file as base line. Run a script to generate voluminous logs. Export audit logs again to a second .csv file. Compare the two .csv files to demonstrate that new audit logs were added while oldest logs were purged.
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:	The ST does not claim this functionality.

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:	The TOE is not a distributed TOE.

2.2 Cryptographic Support (FCS)

2.2.1 FCS_CKM.1 Cryptographic Key Generation

2.2.1.1 TSS

24 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] / TSS section 6.2.1 shows the key generation sizes for all claimed key generation schemes. Same section also identifies the usage for each scheme. a) RSA 2048-bit. Used when generating CSRs. b) ECC P-256/P-384/P-521. Used in TLS.

2.2.1.2 Guidance Documentation

25 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:	As per FCS_CKM.1; the TOE does not allow administrators to configure RSA key size as the only choice is 2048-bit. The TOE also does not provide administrator option to configure elliptic curves for ECDHE. [USER] sections “Configure Server Certificate for Web API Communication” and “Adding or Modifying External Syslog Servers” include information on RSA public/private key pair generation and generating CSR for web server and syslog client.

2.2.1.3 Tests

26 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

27 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*.

28 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

29 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

30 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

31 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)

32 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.

33 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

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

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

35 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.

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

37 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.

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

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

39 for each FFC parameter set and key pair.

[Modified by TD0580] ***FFC Schemes using "safe-prime" groups***

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

Findings:	RSA key generation and ECC key generation are covered by the following CAVP certificates all of which claim RSA KeyGen and KAS ECC Component for NIST curves P-256, P-384 and P-521: C1551, RSA 3118, KAS 210. These claims are consistent with FCS_CKM.1 and FCS_CKM.2 in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

2.2.2 FCS_CKM.2 Cryptographic Key Establishment

2.2.2.1 TSS

41 [Modified by TD0580] 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.

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:

Scheme	SFR	Service
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
ECDH	FCS_IPSEC_EXT.1	Authentication Server

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] / TSS section 6.2.2 illustrates the various key establishment schemes and the usage for each scheme. This section matches the schemes in section 6.2.1 and the FCS_CKM.1 selection. *"a) ECC schemes. Used in TLS. TOE is sender and receiver. The TOE operates as a TLS server for the web GUI/WebAPI providing administration and as a TLS client for the trusted channel with a syslog server.* *Note: RSA is not included in the list of key establishment schemes as it is only used for authentication in TLS, and not key exchange, per the TLS ciphersuites."* The information provided in Section 6.2.2 of the [ST] / TSS is sufficient and includes the scheme, SFR, and service.

Scheme	SFR	Service
ECC	FCS_TLSS_EXT.1	GUI / Administration
	FCS_TLSC_EXT.1/2	LDAP Server
	FCS_TLSC_EXT.1/2	Syslog Server

2.2.2.2 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:	The evaluator checked [USER] and verified that the user guide does not provide the administrator option of configuration of key establishment schemes. However, since there is only one key-establishment method (ECC) supported by the TOE, no configuration is available or needed.

2.2.2.3 Tests

[Modified by TD0580]

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.

Findings:	Key establishment schemes are covered by the following CAVP certificates for KAS-ECC: C1551, KAS 210. These claims are consistent with FCS_CKM.2 in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

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.

RSA-based key establishment

55 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.

Note	The ST does not claim RSA-based key establishment schemes.

FFC Schemes using "safe-prime" groups

56 The evaluator shall verify the correctness of the TSF's implementation of safe-prime groups by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses safe-prime groups. This test must be performed for each safe-prime group that each protocol uses.

Findings:	The ST does not claim FFC schemes using “safe-prime” groups.

2.2.3 FCS_CKM.4 Cryptographic Key Destruction

2.2.3.1 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:	All relevant keys are described in table 15 in section 6.5 of the [ST] / TSS, which includes their origin, their storage location and zeroization (destruction) methods. Keys live in both persistent Flash as well as in RAM and are in plaintext. The zeroization methods used are consistent with the selected destruction method in the SFR. The TOE claims cryptographic channels covering TLS for trusted channels. The TOE would be required to persistently store private keys and X.509 public key certificates when acting as a server/peer in TLS capacities which is consistent with the given table. The various session keys are consistent with the protocols.

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:	The mechanism by which the TOE destroys plaintext keys in non-volatile memory is described in [ST] / TSS section 6.5.1. The method used is consistent with the selected destruction method in the SFR.

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] / TSS section 6.5.1 provides sufficient information on zeroization methods for volatile and non-volatile memory to ensure that the relevant interfaces support the selections and description in the TSS.

[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.

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:	All relevant keys are described in [ST] / TSS section 6.5.1. The ST does not claim any keys that are stored in non-plaintext format.

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:	No such information is conveyed in the [ST] / TSS. There are no obvious circumstances that would prevent conformance to the described mechanism.

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:	The TOE does not claim this selection.

2.2.3.2 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:	There are no obvious circumstances that would prevent conformance to the key destruction requirement and therefore, [USER] does not state any situations where key destruction may be delayed or prevented.

2.2.3.3 Tests

65 None

[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).

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

2.2.4.1 TSS

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

Findings:	TSS identifies the key sizes and modes supported by the TOE for data encryption/decryption. [ST] / TSS section 6.2.4 states that the TOE provides symmetric encryption and decryption capabilities using 128 and 256 bit AES in CBC and GCM mode. AES is implemented in TLS and is used in BouncyCastle key encryption.

2.2.4.2 Guidance Documentation

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

Findings:	There is no configuration available to enable a specific mode or key size of the supported algorithm (AES) as the TOE is restricted to allow only AES-128 and AES-256 in CBC and GCM modes during TLS handshake.

2.2.4.3 Tests

AES-CBC Known Answer Tests

68 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.

69 **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.

70 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.

71 **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.

72 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.

73 **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].

74 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.

75 **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].

76 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

77 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.

78 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

79 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]
```

80 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.

81 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

82 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.

83 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.

84 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.

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

AES-CTR Known Answer Tests

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

selection in FCS_COP.1/DataEncryption, the AES-CBC Known Answer Test, AES-GCM Known Answer Test, or the following test shall be performed (all of these tests demonstrate the correctness of the AES algorithm):

87 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.

88 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.

89 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.

90 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].

91 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

92 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

93 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]
```

94 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 encryption and decryption is covered by the following CAVP certificates both of which claim 128 and 256-bit AES in CBC and GCM mode: C1551, AES 5940. These claims are consistent with FCS_COP.1/DataEncryption in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

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

2.2.5.1 TSS

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

Findings:	[ST] / TSS section 6.2.5 specifies that the TOE provides signature generation and verification services using RSA Signature Algorithm with key size of 2048.

2.2.5.2 Guidance Documentation

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

Findings:	There are no configuration options available for TOE signature services. It is enabled by default to support RSA 2048. Section “Upgrade/Downgrade Guidelines to/from Release 4.5 or Higher/Lower” in [USER] also supports this as “the system is able to read and install (upgrade to) releases from 4.5 and higher that use an RSA 2048 bit key.”

2.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

97 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

98 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

99 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.

100 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

101 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.

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

Findings:	RSA signature verification is covered by the following CAVP certificates both of which claim RSA Signature Algorithm with key size of 2048: C1551, RSA 3118. These claims are consistent with FCS_COP.1/SigGen in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

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

2.2.6.1 TSS

103 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] / TSS section 6.2.6 states that the TOE provides services using SHA-1 and SHA-256 and they are implemented in the following parts of the TSF: a) TLS; b) Digital signature verification as part of trusted update validation; c) Hashing of passwords in non-volatile storage; and d) Authentication of NTP server.

2.2.6.2 Guidance Documentation

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

Findings:	Cryptographic hash configuration is enabled by default and matches the ST requirements. [USER] sections "Software Upgrade" and "Enabling and Configuring NTP Servers" provide matching information about determined hash sizes of the TSF parts such as: 1) "Digital signature verification as part of trusted update validation: "Software Upgrade" specifies SHA-256 as the hash function that is used. 2) Authentication of NTP server: "Enabling and Configuring NTP Servers" section specifies SHA-1 as the only message digest type.

2.2.6.3 Tests

105 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.

106 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

107 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

108 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

109 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

110 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

111 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:	Hashing is covered by the following CAVP certificates which claim SHA-1 and/or SHA-256: C1551, SHS 4693, C1550, SHS 4692. These claims are consistent with

	FCS_COP.1/Hash in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

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

2.2.7.1 TSS

112 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: The key length, block size, hash function and output MAC length are found in table 14 in section 6.2.7 of [ST] / TSS.

Table 14: HMAC Characteristics

Algorithm	Block Size	Key Size	Digest Size
HMAC-SHA-1	512 bits	160 bits	160 bits
HMAC-SHA-256	512 bits	256 bits	256 bits
HMAC-SHA-512	1024 bits	512 bits	512 bits

2.2.7.2 Guidance Documentation

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

Findings: Cryptographic keyed hash configuration is enabled by default. In addition, [SUPP] section 2.3 indicates that during FIPS-Approved power-up self-tests, integrity check is done for ST claimed HMAC functions.

2.2.7.3 Tests

114 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: HMAC is covered by the following CAVP certificates both of which claim HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512: C1551, HMAC 3915. These claims are consistent with FCS_COP.1/KeyedHash in the [ST] section 5.3.2 and Table 14: HMAC Characteristics. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

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

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

2.2.8.1 TSS

116 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] / TSS section 6.2.10 states that the TOE contains a Hash_DRBG (any) that is seeded from a hardware entropy source that provides a minimum of 256 bits of entropy.

2.2.8.2 Guidance Documentation

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

Findings:	[USER] only provides instructions on enabling FIPS mode, in section of “Enable Server FIPS Encryption”, implicitly configuring use of FIPS approved RBG function. “Government Security Configuration Guide” section in [USER] has sufficient information that the TOE implements FIPS 140-2 Level 1 validated cryptographic modules, which include the claimed DRBG type.

2.2.8.3 Tests

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

119 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).

120 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.

121 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:	Hash DRBG is covered by the following CAVP certificates both of which claim use of a 256-bit key: C1551, DRBG 2493. These claims are consistent with FCS_RBG_EXT.1 in the [ST] section 5.3.2. The evaluator verified that the crypto modules used in the TOE correspond to the crypto modules which have CAVP-validated cryptographic algorithms.

2.3 Identification and Authentication (FIA)

2.3.1 FIA_AFL.1 Authentication Failure Management

2.3.1.1 TSS

122 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] / TSS section 6.3.5 describes that the TOE is capable of tracking authentication failures of administrators. When the administrator-configured number of unsuccessful failed attempts are reached using Web GUI/Web API or serial console, the TOE successfully locks the user account and no longer authenticates valid credentials until the System Administrator unlocks the account. The default administrative account cannot be locked and is limited to the serial console in the evaluated configuration.

123 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] / TSS section 6.3.5 indicates that the default administrative account cannot be locked and has access only to serial console during evaluated configuration. This ensures that authentication failures by remote administrators cannot lead to a situation where no administrator access is available.

2.3.1.2 Guidance Documentation

124 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 requirement is covered in section of “Locking Regular Users” in [USER]. *In DoD Security Policy password mode, both administrator and regular user accounts will be disabled after three (3) consecutive failed local authentication attempts through Web API or Web Console. In either case, the configurable number of days of inactivity or the configured number of consecutive failed local authentication attempts, the user account is disabled and the user is prevented from logging in to an NPB. The next time the user attempts to login, they are prompted to contact an administrator to enable their account. Only administrators can enable user accounts. Enabling and disabling user accounts is syslogged.”* Same section specifies that the default "admin" user account will not be locked implicitly implying that it can be used to recover accounts. Section “Recovering when no user can authenticate” describes the necessary steps for a recovery in a scenario where no one is able to log in to the Web Console or the Web API which includes remote administrators. In this specific case, serial console can be used.

125 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:	The requirement is covered in section of “Recovering when no user can authenticate” in [USER]. It is ensured that in a scenario where no user is able to log in to the Web Console or the Web API, the serial console can be used to revert this situation as long as administrator can access the TOE with valid credentials. “Locking Regular Users” in [USER] specifies that the default "admin" account will not be locked. Therefore, administration cannot be made permanently or temporarily unavailable due to blocking.

2.3.1.3 Tests

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

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

High-Level Test Description
Using the Web interface, attempt to log into the TOE twice using an incorrect password. On the third attempt, log in with the correct password and verify that the threshold has not been reached. Using the Web interface, attempt to log into the TOE three times using an incorrect password. On the fourth attempt, log in with the correct password and verify that the threshold has been reached and that the user cannot log in. As another security admin, unlock the affected account, and log in with correct password. The user should log into the TOE successfully. Repeat the failed authentication (three times) via Web API. Then, unlock the locked remote administrator via Web API using ‘admin’ account.

High-Level Test Description
Repeat the failed authentication (three times) via Web API. Then, unlock the locked remote administrator via serial console using ‘admin’ account.
Findings: PASS

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.

Findings:	See previous test case. The ST includes administrative action selection in FIA_AFL.1.2. Previous test case includes performing each action specified in the ST to re-enable the remote administrator’s access.

2.3.2 FIA_PMG_EXT.1 Password Management

2.3.2.1 TSS

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

Findings:	[ST] / TSS section 6.3.1 contains the list of the supported special characters and minimum and maximum number of characters supported for administrator passwords, as follows: “The passwords can be composed of any combination of upper and lower case letters, numbers, and special characters “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, “`”, ”~”, ”_”, ”+”, ”-“, ”=”, ”{“, ”}”, “|”, “\”, “:”, “”, ”;”, ””, “<”, “>”, “?”, “,”, “.”, “/”, “[”, ”]”. “The minimum password length is settable by the Administrator and can range from 15 to 100 characters.”

2.3.2.2 Guidance Documentation

128 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 evaluator checked the user guide and determined that this requirement is covered in the Section of "Details of DoD Password Security Policies": *The minimum DoD Security Policy password length is configurable, defaults to 15 characters (the minimum required by NDcPP), and must use at least one (1) character from each for the first four (4) following character sets:* *- Uppercase letters A – Z* *- Lowercase letters a – z* *- Numeric digits 0 – 9* *- Special characters ` ~ ! @ # $ % ^ & * ( ) _ + - = { } | [ ] \ : " ; ' < > ? , . /* The same section also has additional information on strong password policies and on the password similarity and password reuse prevention mechanisms.

2.3.2.3 Tests

129 The evaluator shall perform the following tests.

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

High-Level Test Description
Via Web GUI:
Change password to be less than 15 characters in length, and it shall fail.
Change password to be of no upper case character, and it shall fail.
Change password to be of no lower case character, and it shall fail.
Change password to be of no numeric character, and it shall fail.
Change password to be of no special character, and it shall fail.
Via Web API:
Change the minimum password length to 101 (maximum allowed is 100) and it shall fail.
Change the minimum password length to 10 and it shall fail.
Change the password with a compliant one and it shall succeed.
Findings: PASS

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

Findings:	See previous test case. The evaluator tested passwords that do not meet the requirements in some way and verified that the TOE does not support those passwords.

2.3.3 FIA_UIA_EXT.1 User Identification and Authentication

2.3.3.1 TSS

130 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] / TSS section 6.3.2 specifies the logon methods as follows: a) Local. Directly connecting to the TOE via the Serial Console b) Web GUI. Remotely connecting to the TOE web GUI/WebAPI via HTTPS [ST] / TSS section 6.3.3 specifies the logon process for each logon method provided.

131 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] / TSS section 6.3.2 contains a list of actions that may occur or allowed prior to authentication. a) Display the warning banner b) Access to help files (PDF and html format) c) View alarm notifications (i.e. system health alarms) d) View software version number

132 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:	The TOE is not a distributed TOE.

133 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:	The TOE is not a distributed TOE.

2.3.3.2 Guidance Documentation

134 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:	The evaluator checked the [USER] and [SUPP], and verified that the TOE only supports password based authentication for administrators to access the web UI and local console (see “Administration Interfaces” section in [SUPP]); the user guide provides instructions on how to access the TOE for initial installation and configuration using default credentials (section of “Configure Government Security Settings” in [USER]) and how to create a user and provision the user account with password (section of “Add Users” in [USER]). There is no configuration to limit services provided before login.

2.3.3.3 Tests

135 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: Log into the identified management interface using a known-good credential and logout. Attempt to login into the identified management interface using known-bad credentials. Verify the attempt fails. 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 set of services are consistent with those stated in the ST. Prior to identification and authentication process, a user can only access the following: - Display the warning banner in accordance with FTA_TAB.1; - Access to help files (PDF and html format) - View alarm notifications - View software version number.
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 set of services are consistent with those stated in the ST. Prior to identification and authentication process, a user can only access the following: - Display the warning banner in accordance with FTA_TAB.1; - View alarm notifications - View software version number
Findings: PASS

136 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:	The TOE is not a distributed TOE.

2.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

137 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.

2.3.5 FIA_UAU.7 Protected Authentication Feedback

2.3.5.1 TSS

138 None

2.3.5.2 Guidance Documentation

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

Findings:	The evaluator examined [USER] to determine that no necessary preparatory steps are needed to ensure authentication data is not revealed while entering for each local login.

2.3.5.3 Tests

140 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.

High-Level Test Description
Ensure the password field does not echo characters – even a masking character -- as claimed by the ST.
Findings: PASS

2.4 Security management (FMT)

2.4.1 FMT_MOF.1/ManualUpdate

2.4.1.1 TSS

141 For distributed TOEs see chapter 2.4.1.1 of the [SD]. There are no specific requirements for non-distributed TOEs.

Findings:	The TOE is not a distributed TOE.

2.4.1.2 Guidance Documentation

142 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:	The evaluator examined the [USER] and determined that any necessary steps to perform manual update are described under “Software Upgrade/Downgrade” in the Appendix section. Same section provides warnings on the approximate duration of inaccessibility to the TOE during a manual software update and duration to reach a fully operational state after the manual update is completed and the system restarts.

143 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:	The TOE does not have distributed components.

2.4.1.3 Tests

144 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.

145 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 Web GUI using an account with privileges which should not permit upgrades. Attempt to upgrade the device. The action should fail.

High-Level Test Description
Attempt to push firmware upgrades directly via Web API with a user account that does not have privileges. The action should fail.
Findings: PASS

2.4.2 FMT_MTD.1/CoreData Management of TSF Data

2.4.2.1 TSS

146 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:	[ST] / TSS section 6.4.3 states that users are required to login and no administrative function is available before identification and authentication occurs.

147 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] / TSS section 6.4.4 describes that access to TSF data and functions such as "generation, importation, or deletion of cryptographic keys" is restricted to the Security Administrators. No administrative function is available before identification and authentication occurs.

2.4.2.2 Guidance Documentation

148 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:	[USER] and section 1.4.3 "Evaluated Functions" of [SUPP] provide information on the secure administration of security functions which include TSF-data-manipulating functions, such as: a) Administrator authentication with passwords b) Configurable password policies c) Role Based Access Control d) Access banners e) Protection of cryptographic keys and passwords f) Management of critical security functions and data, which also includes: - Secure web access - Local audit logs - Managing users - Session management - Certificate management (including trust store / CA certs configuration) - Software upgrade (trusted update) The "secure administration" refers to ensuring that only administrators have access to these functions.

149 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:	[USER] provides sufficient information for the administrator to maintain the trust store, and securely load CA certificates, under the following sections: - "Uploading a Custom Server Certificate" - "Syslog Support"

2.4.2.3 Tests

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

2.4.3 FMT_SMF.1 Specification of Management Functions

151 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.

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

152 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:	[ST] / TSS section 6.4.5 and [USER] indicates that the TOE may be managed via the CLI (console), GUI (HTTPS) or WebAPI (HTTPS) and describes the specific management capabilities including which functions can be performed on specific interfaces.

153 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:	[ST] / TSS section 6.4.5 specifies the Serial Console as the local administrative interface: *"a) Ability to administer the TOE locally via Serial Console and remotely via web GUI/WebAPI"* [SUPP] also refers to the Serial console as the local administrative interface in the section 3.2 "Administration Interfaces".

154 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:	The TOE is not a distributed TOE.

2.4.3.2 Guidance Documentation

155 See section 2.4.4.1.

2.4.3.3 Tests

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

Findings:	The evaluator tested management functions using all the supported interfaces throughout this test plan. In addition to standalone configuration, cluster mode is independently tested for management functions. Guidance and evaluator conducted tests provided sufficient proof that clustering does not change how TSF data is managed for each TOE included in a cluster.

2.4.4 FMT_SMR.2 Restrictions on security roles

2.4.4.1 TSS

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

Findings:	[ST] / TSS section 6.4.6 details the TOE supported roles as pre-defined profiles that are assigned when creating a user and the restrictions of the roles involving administration of the TOE, as the management of TSF data is restricted to the Security Administrators.

2.4.4.2 Guidance Documentation

158 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:	The evaluator verified that [SUPP] contains instructions for administering the TOE both locally and remotely in section of “Administration Interfaces”.

2.4.4.3 Tests

159 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 this test plan.

2.5 Protection of the TSF (FPT)

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

2.5.1.1 TSS

160 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: Table 15 in section 6.5.1 of [ST] / TSS includes all the relevant information. In all cases, plaintext keys cannot be viewed through an interface designed specifically for that purpose.

Table 15: Keys

Key	Algorithm	Storage	Zeroization
TLS Server Private Key	BouncyCastle RSA (2048 bits)	Flash – plaintext	Overwritten with zeroes via a shell script called by Java. The shell script uses the Linux 'dd' command-line utility to zeroize the non-volatile memory via a single direct overwrite (consisting of zeroes) followed by a read-verify.
		RAM - plaintext	JVM garbage collection on deallocation.
TLS Client Private Key	BouncyCastle RSA (2048 bits)	Flash – plaintext	Overwritten with zeroes via a shell script called by Java. The shell script uses the Linux 'dd' command-line utility to zeroize the non-volatile memory via a single direct overwrite (consisting of zeroes) followed by a read-verify.
		RAM - plaintext	JVM garbage collection on deallocation.
TLS Session Keys	BouncyCastle AES (128)	RAM - plaintext	JVM garbage collection on deallocation.
NTP Key	User generated	Flash – plaintext	Overwritten with new value of key via Linux file overwritten.

2.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

2.5.2.1 TSS

161 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: Table 16 in section 6.5.2 of [ST] / TSS describes the passwords that are protected and TSS states that in all cases, plaintext passwords cannot be viewed through an interface designed specifically for that purpose.

Table 16: Passwords

Key/Password	Generation/ Algorithm	Storage
Locally stored administrator passwords	User generated	Flash - SHA-1

2.5.3 FPT_TST_EXT.1 TSF testing

2.5.3.1 TSS

162 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: [ST] / TSS section 6.5.3 lists the self-tests and provides sufficient information on how they are performed. The TOE performs this suite of FIPS power-up and conditional self-tests to verify its correct operation. If any of the self-tests fail, the TOE enters into a critical error state and the appliance must be rebooted by an administrator to run the tests again.

163 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: The TOE is not a distributed TOE.

2.5.3.2 Guidance Documentation

164 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: The evaluator checked [SUPP] and verified that, in section “Power-on Self-Tests”, it describes possible errors that may result from self-tests, and actions the administrator should take in response. The description further supports the self-testing description in the TSS.

165 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:	The TOE is not a distributed TOE.

2.5.3.3 Tests

166 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.

167 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.

168 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.

169 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
Reset the TOE and witness that the start-up includes an indicator that self-tests were executed and passed permitting the device to operate. Via local console, logged in as a security administrator, issued on-demand self-tests, and observed that the TOE restarted. The evaluator logged in again via local console, choosing to show the results of self-tests. There was also an audit record for self-tests.
Findings: PASS

2.5.4 FPT_TUD_EXT.1 Trusted Update

2.5.4.1 TSS

170 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] / TSS section 6.5.4 states that the current firmware version may be queried using GUI/WebAPI. After the TOE verifies the digital signature of the upgrade files and the software is installed, the TOE automatically restarts and requires the administrator to log in again for the upgrade process to be complete.

171 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] / TSS section 6.5.4 specifies that the TOE verifies the digital signature of the upgrade files using a hardcoded RSA 2048-bit public key. Verification occurs before installation and installation fails and a log is generated if the signature verification fails for any reason. Upon successful verification, the TOE software extracts the upgrade components from the tar file and performs the software upgrade. After the software is installed, the TOE automatically restarts and requires the administrator to log in again for the upgrade process to be complete. If the newly installed software fails for any reason, the TOE software reverts to the previous version without loss of functionality.

172 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:	FPT_TUD_EXT.1.2 does not claim those selections.

173 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:	The TOE is not a distributed TOE.

174 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:	The TOE does not rely on hash-based integrity mechanisms.

2.5.4.2 Guidance Documentation

175 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:	The evaluator checked the section “Updating the TOE” in [SUPP] which indicates that the authenticity of the update is verified by digital signature. The evaluator also

	verified that section of “Software Upgrade/Downgrade and Cold Spare Upgrade Procedures” in [USER] describes how to query the currently active version, how to upgrade the TOE and how to determine if upgrade is successful or unsuccessful. The description is consistent with TSS in the [ST]. The TOE does not claim delayed activation.

176 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:	The evaluator checked the section “Updating the TOE” in [SUPP] which indicates that the authenticity of the update is verified by digital signature. The evaluator also verified that section of “Software Upgrade/Downgrade and Cold Spare Upgrade Procedures” in [USER] describes how to query the currently active version, how to upgrade the TOE and how to determine if upgrade is successful or unsuccessful. The description is consistent with TSS in the [ST].

177 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:	The TOE does not rely on hash-based integrity mechanisms.

178 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 Security Administrator; it does not need to give information about the internal communication that takes place when applying updates.

Findings:	The TOE is not a distributed TOE.

179 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:	The TOE is not a distributed TOE.

180 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 used for this TOE.

2.5.4.3 Tests

181 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
Verify the current version of the TOE. Attempt to install a legitimate version of the TOE by upgrading it to the evaluated version. After the install, get the current version of the TOE and ensure it is consistent with the newly installed version.
Findings: PASS

b. Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

   1) A modified version (e.g. using a hex editor) of a legitimately signed update

   2) An image that has not been signed

   3) An image signed with an invalid signature (e.g. by using a different key as expected for creating the signature or by manual modification of a legitimate signature)

   4) If the TOE allows a delayed activation of updates the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs depending on the point in time when an attempted update is rejected. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

High-Level Test Description
Attempt to install a bad image, an unsigned image, and a badly signed image for an upgrade attempt. After each attempt, make sure the TOE rejects the illegitimate update. The TOE does not support delayed activation.
Findings: PASS

c. Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted. If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

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

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

3) If the TOE allows delayed activation of updates, the TOE must be able to display both the currently executing version and most recently installed version. The handling of version information of the most recently installed version might differ between different TOEs. Depending on the point in

time when the attempted update is rejected, the most recently installed version might or might not be updated. The evaluator shall verify that the TOE handles the most recently installed version information for that case as described in the guidance documentation. After the TOE has rejected the update the evaluator shall verify, that both, current version and most recently installed version, reflect the same version information as prior to the update attempt.

Findings:	The TOE does not support published hashes.

182 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.

183 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).

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

2.5.5 FPT_STM_EXT.1 Reliable Time Stamps

2.5.5.1 TSS

185 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] / TSS section 6.5.5 has a list of security functions that make use of time and states that the TOE uses external NTP servers to synchronize the clock and provide reliable timestamps for syslog messages.

2.5.5.2 Guidance Documentation

186 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:	The evaluator checked [USER] and confirmed that it provides instructions to the administrator how to set the time by use of NTP server in “Enabling and Configuring NTP Servers” subsection under “Viewing and Modifying System Settings”. It further describes how to establish communication between the TOE and NTP server and configuration of the NTP client on the TOE to support the communication.

2.5.5.3 Tests

187 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.

Findings:	The TOE only supports the use of an NTP server and does not support direct setting of the time.

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.

High-Level Test Description
The evaluator changed the time on NTP server. There should be an audit record which shows that the TOE's NTP client was in sync with updated time.
Findings: PASS

188 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 support independent time information.

2.6 TOE Access (FTA)

2.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

2.6.1.1 TSS

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

Findings:	[ST] / TSS section 6.6.1 states that the TOE is configured to terminate local inactive sessions (Serial Console) based on a specified time period of inactivity configured by a Security Administrator.

2.6.1.2 Guidance Documentation

190 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:	The evaluator confirmed that the section of "Serial (CRAFT) Port Console Access and Authentication" in [USER] contains instructions for configuring local admin session locking or termination, where the administrator can set the "Session Timeout" after

	inactivity. *"By default, once you are logged in, the current session times out after 60 seconds of inactivity, requiring you to log back in. This setting is configurable to the right of the Session timeout field."*

2.6.1.3 Tests

191 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 60, 120 seconds: Change the idle timeout to this value; Log into the device; Before the set threshold is reached, verify the session is still alive by sending a keep alive. This should reset the timeout clock. The purpose is to ensure the timeout is not premature. Wait for the full duration of the timeout without sending any keep alives. The session should terminate.
Findings: PASS

2.6.2 FTA_SSL.3 TSF-initiated Termination

2.6.2.1 TSS

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

Findings:	[ST] / TSS section 6.6.2 states that the Security Administrator may configure the TOE to terminate an inactive remote interactive session following a specified period of time.

2.6.2.2 Guidance Documentation

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

Findings:	The evaluator confirmed that the section of "Viewing and Modifying System Settings" in [USER] contains instructions for configuring inactivity time period for remote administrator session termination. *"Session timeout: Select the hyperlink to configure the idle login session timeout. If a timeout is specified, a user is automatically logged out if there is no Web Console activity from that user in the specified time. The logout can be configured for minutes, hours, or never. You should set the session timeout to at least 10 minutes to allow potential software upgrades to complete."*

2.6.2.3 Tests

194 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, 2 minutes: Change the idle timeout to this value; Log into the device; Before the set threshold is reached, verify the session is still alive by sending a keep alive. This should reset the timeout clock. The purpose is to ensure the timeout is not premature. Wait for the full duration of the timeout without sending any keep alives. The session should terminate. Note that because the system uses a single command to control all idle timers, we will set in one interface and check in another.
Findings: PASS

2.6.3 FTA_SSL.4 User-initiated Termination

2.6.3.1 TSS

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

Findings:	[ST] / TSS section 6.6.3 states that Administrative users are allowed to terminate their own interactive sessions by logging out.

2.6.3.2 Guidance Documentation

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

Findings:	The evaluator verified that section of “Administration Interfaces” in [SUPP] describes how to terminate local or remote interactive session. *“User may terminate the local session by selecting Logout from Main Menu.”* *“User may use the Logout button to terminate the current Web Console session.”*

2.6.3.3 Tests

197 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 command. 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 Web GUI interface. Copy the URL presented. Log out using the TSFI command / action. Paste the URL back into the web browser and attempt to navigate directly to it.
Findings: PASS

2.6.4 FTA_TAB.1 Default TOE Access Banners

2.6.4.1 TSS

198 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:	Section 6.3.2 FIA_UIA_EXT.1 in [ST] / TSS indicates the methods of access available for the Security Administrator. [ST] / TSS section 6.6.4 states that the TOE displays an administrator configurable message to users prior to login at the Serial Console and Web GUI.

2.6.4.2 Guidance Documentation

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

Findings:	The evaluator verified that [USER] describes how to configure the banner message in section “Configure the Login Banner” (for remote) and “Serial (CRAFT) Port Console Access and Authentication” (for local).

2.6.4.3 Tests

200 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
From the Web GUI: Change the banner for Web GUI to a random string. Logout and verify in the login page that the banner is shown prior to I&A. Change the banner for Serial Console to a random string. Logout and verify in the login prompt for serial console that the banner is shown.
Findings: PASS

2.7 Trusted path/channels (FTP)

2.7.1 FTP_ITC.1 Inter-TSF trusted channel

2.7.1.1 TSS

201 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] / TSS section 6.7.1 describes the supported secure communications. The TOE provides secure communication for the syslog server and LDAP Authentication server with TLS v1.2.

2.7.1.2 Guidance Documentation

202 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:	The evaluator confirmed that the instructions for establishing trusted channels are provided in [USER] section of “Syslog Support” for syslog and “Configuring LDAP” for LDAP. Other information may be found in the section of “TLS Communication” for syslog and “Administrator Authentication” for LDAP in [SUPP].

2.7.1.3 Tests

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

204 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.

Findings:	The TOE maintains trusted channels to the remote audit log and LDAP server, which are 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. Log into the syslog-ng machine and restart the syslog-ng service to clear out TLS session information (which will force a new handshake). Log into the TOE using a predefined LDAP user. Examine wireshark and verify that the TOE initiates TLS communications with the syslog and LDAP endpoints that are encrypted.
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.

Findings:	Refer to previous test. The packet captures show that the TOE initiates TLS communications with the syslog and LDAP endpoints that are encrypted. In addition, the channel data for each communication channel is constantly tested throughout the evaluation.

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

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

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

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

High-Level Test Description
Engage wireshark over the logging interface. Log into the TOE over the serial console using a predefined LDAP user. Logout. Physically disconnect both the remote logging server and the LDAP server by disconnecting the physical cable that connects the server (that resides in the services VM) into the engineering switch. Wait 1 minute. Log into the TOE over the serial console using a predefined LDAP user. LDAP server will be unable to confirm the login. Wait 1 minute. Physically reconnect the server device back into the switch. Log into the TOE over the serial console using a predefined LDAP user. Examine wireshark and verify that both the syslog and LDAP continue to send encrypted Application Data packets without any user intervention. Repeat the above test but wait 7m30s between attempts for a total of 15 minutes of interruption. Note: Above tests would trigger a log transfer event every time the user initiated a login attempt which can be used to verify for syslog secure channel as well that when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.
Findings: PASS

Further assurance activities are associated with the specific protocols.

205 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:	This is not a distributed TOE.

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

2.7.2 FTP_TRP.1/Admin Trusted Path

2.7.2.1 TSS

207 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] / TSS section 6.7.2 indicates that WebGUI/WebAPI are accessible over HTTPS per FCS_HTTPS_EXT.1.1. The protocol listed in the TSS is consistent with the requirement and the requirements in the ST.

2.7.2.2 Guidance Documentation

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

Findings:	The evaluator confirmed that the instructions are provided in the section of “Administration Interfaces” of [SUPP] which lists all the sections in the [USER] that contain instructions to securely establish the remote administrative sessions.

2.7.2.3 Tests

209 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.

Findings:	The only trusted paths are the web GUI and Web API, which are both set up as per the evaluated configuration. They are 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

210 Further assurance activities are associated with the specific protocols.

211 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:	This is not a distributed TOE.

3 Evaluation Activities for Optional Requirements

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

3.1.1.1 TSS

FCS_TLSC_EXT.2.1

212 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:	[ST] / TSS section 6.2.11 states that: "The TOE supports 2-way certificate authentication with X.509v3 certificates" which implies that the TOE uses client-side certificates for TLS mutual authentication.

3.1.1.2 Guidance Documentation

FCS_TLSC_EXT.2.1

213 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:	Section "Adding or Modifying External Syslog Servers" of [USER] states: *"If you want to use the encryption and mutual authentication capability that is provided by TLS, you need to upload both a client and a server trusted root certificate."* [USER] includes instructions for configuring the client-side certificates under the following sections for each TOE model: ***"**Generate and Upload a Client Certificate to a Vision ONE System"* *"Generate and Upload a Client Certificate to a Vision X System"* *"Generating and Uploading a Client Certificate to a TradeVision System"* *"Generate and Upload a Client Certificate to a 7300/7303 System"* *"Generate and Upload a Client Certificate to a Vision E10S System"* *"Generate and Upload a Client Certificate to a Vision E40 or a Vision E100 System"* [USER] includes instructions for configuring the server trusted root certificate under the following section: *"Upload a Server Trusted Root Certificate"*

3.1.1.3 Tests

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

FCS_TLSC_EXT.2.1

215 The tests are covered under FCS_TLSC_EXT.1.1 Test 1 and testing for FIA_X.509_EXT.*).

4 Evaluation Activities for Selection-Based Requirements

4.1 Cryptographic Support (FCS)

4.1.1 FCS_HTTPS_EXT.1 HTTPS Protocol

4.1.1.1 TSS

216 The evaluator shall examine the TSS and determine that enough detail is provided to explain how the implementation complies with RFC 2818.

Findings:	[ST] / TSS section 6.2.8 provides a description as to how the TOE conforms with RFC 2818. RFC 2818 specifies HTTP over TLS. The majority of RFC 2818 is spent on discussing practices for validating endpoint identities and how connections must be setup and torn down. The TOE web GUI/WebAPI operates on an explicit port designed to natively speak TLS: it does not attempt STARTTLS or similar multi-protocol negotiation which is described in section 2.3 of RFC 2818. The web server attempts to send closure Alerts prior to closing a connection in accordance with section 2.2.2 of RFC 2818.

4.1.1.2 Guidance Documentation

217 The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to configure TOE for use as an HTTPS client or HTTPS server.

Findings:	[SUPP] section “TLS Communication” provides instruction on enabling TLS over HTTP communications to use the TOE as an HTTPS server. [USER] section “Configure Server Certificate for Web API Communication” states: *“The system is supplied with an Keysight signed certification for TLS and a Thawte certification for code signing. You can upload a custom certificate signed by a CA for better TLS security. All the TLS1.2-encrypted communication interfaces use the self-signed SSL certificate by default, or the custom certificate if one is uploaded.”* [USER] provides information on uploading a custom certificate in “Uploading a Custom Server Certificate” The TOE does not use HTTPS in a client capacity as stated in the [ST].

4.1.1.3 Tests

218 This test is now performed as part of FIA_X509_EXT.1/Rev testing.

219 Tests are performed in conjunction with the TLS evaluation activities.

220 If the TOE is an HTTPS client or an HTTPS server utilizing X.509 client authentication, then the certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1.

4.1.2 FCS_NTP_EXT.1 NTP Protocol

4.1.2.1 TSS

FCS_NTP_EXT.1.1

221 The evaluator shall examine the TSS to ensure identifies the version of NTP supported, how it is implemented and what approach the TOE uses to ensure the timestamp it receives from an NTP timeserver (or NTP peer) is from an authenticated source and the integrity of the time has been maintained.

Findings:	[ST] / TSS section 6.2.9 states that the TOE uses NTP v4 that implements the symmetric key authentication scheme defined in RFC5905, performing authentication using SHA-1. The TOE does not update NTP time from broadcast or multicast addresses and supports configuration of at least 3 NTP time sources to ensure integrity of the time has been maintained.

222 The TOE must support at least one of the methods or may use multiple methods, as specified in the SFR element 1.2. The evaluator shall ensure that each method selected in the ST is described in the TSS, including the version of NTP supported in element 1.1, the message digest algorithms used to verify the authenticity of the timestamp and/or the protocols used to ensure integrity of the timestamp.

Findings:	[ST] / TSS section 6.2.9 matches the information provided in the SFR as NTP v4 is supported with the message digest algorithm of SHA1.

4.1.2.2 Guidance Documentation

FCS_NTP_EXT.1.1

223 The evaluator shall examine the guidance documentation to ensure it provides the administrator instructions as how to configure the version of NTP supported, how to configure multiple NTP servers for the TOE's time source and how to configure the TOE to use the method(s) that are selected in the ST.

Findings:	As stated in the [ST], and confirmed in [USER] in "Remote Services Settings" section, TSF supports configuration of at least three (3) NTP time sources to provide redundancy. NTP converges to an accurate time more quickly when multiple NTP servers are configured. Same section has the following statement on the version of NTP supported: *"The system supports NTP version 4, but also retains compatibility with versions 1-3"* The evaluator examined [USER] and confirmed that configuration of NTP is described in sections of "Enable authentication on any NTP servers" and "Enabling and Configuring NTP Servers", including how to configure multiple NTP servers.

FCS_NTP_EXT.1.2

224 For each of the secondary selections made in the ST, the evaluator shall examine the guidance document to ensure it instructs the Security Administrator how to configure the TOE to use the algorithms that support the authenticity of the timestamp and/or how to configure the TOE to use the protocols that ensure the integrity of the timestamp.

Findings:	The evaluator examined [USER] and confirmed that section: "Enabling and Configuring NTP Servers" describes how to configure the TOE to use the algorithms that support the authenticity of the timestamp. The supported algorithm can be enabled/disabled and verified under System > Settings > Remote Services > NTP > Selecting the Configured NTP Server > NTP Server Settings window as shown in the [USER].

225 Each primary selection in the SFR contains selections that specify a cryptographic algorithm or cryptographic protocol. For each of these secondary selections made in the ST, the evaluator shall examine the guidance documentation to ensure that the documentation instructs the Security Administrator how to configure the TOE to use the chosen option(s).

Findings:	The TOE only supports one primary cryptographic algorithm and therefore has no instructions on how to configure other options.

FCS_NTP_EXT.1.3

226 The evaluator shall examine the guidance documentation to ensure it provides the Security Administrator instructions as how to configure the TOE to not accept broadcast and multicast NTP packets that would result in the timestamp being updated.

Findings:	The TOE does not accept broadcast and multicast NTP packets as stated in the [ST], so it is not configurable.

4.1.2.3 Tests

FCS_NTP_EXT.1.1

227 The version of NTP selected in element 1.1 and specified in the ST shall be verified by observing establishment of a connection to an external NTP server known to be using the specified version(s) of NTP. This may be combined with tests of other aspects of FCS_NTP_EXT.1 as described below.

High-Level Test Description
Engage wireshark over the appropriate interface.
Log into the trusted path.
Set up NTP settings on the TOE.
Start up NTP server.
Analyse the traffic between the NTP Server and the TOE.
Findings: PASS

FCS_NTP_EXT.1.2

228 The cryptographic algorithms selected in element 1.2 and specified in the ST will have been specified in an FCS_COP SFR and tested in the accompanying Evaluation Activity for that SFR. Likewise, the cryptographic protocol selected in in element 1.2 and specified in the ST will have been specified in an FCS SFR and tested in the accompanying Evaluation Activity for that SFR.

229 [Conditional] If the message digest algorithm is claimed in element 1.2, the evaluator will change the message digest algorithm used by the NTP server in such a way that new value does not match the configuration on the TOE and confirms that the TOE does not synchronize to this time source.

230 The evaluator shall use a packet sniffer to capture the network traffic between the TOE and the NTP server. The evaluator uses the captured network traffic, to verify the NTP version, to observe time change of the TOE and uses the TOE's audit log to determine that the TOE accepted the NTP server's timestamp update.

231 The captured traffic is also used to verify that the appropriate message digest algorithm was used to authenticate the time source and/or the appropriate protocol was used to ensure integrity of the timestamp that was transmitted in the NTP packets.

High-Level Test Description
Engage wireshark over the appropriate interface. Log into the trusted path. Make sure the NTP settings on the TOE are in evaluated configuration and time is synchronized. Change time settings on the NTP server and restart ntpd service. Change message digest algorithm on the NTP server and restart ntpd service. Analyse the traffic between the NTP Server and the TOE.
Findings: PASS

FCS_NTP_EXT.1.3

232 The evaluator shall configure NTP server(s) to support periodic time updates to broadcast and multicast addresses. The evaluator shall confirm the TOE is configured to not accept broadcast and multicast NTP packets that would result in the timestamp being updated. The evaluator shall check that the time stamp is not updated after receipt of the broadcast and multicast packets.

High-Level Test Description
Engage wireshark over the appropriate interface. Log into the trusted path. Make sure the NTP settings on the TOE are in evaluated configuration and time is synchronized. Change /etc/ntp.conf settings on the NTP server to support periodic time updates to broadcast/multicast addresses. Change time/date settings on the NTP server. Check to see if the TOE accepts these broadcast/multicast NTP packets or not. Analyse the traffic between the NTP Server and the TOE.
Findings: PASS

FCS_NTP_EXT.1.4

233 [Modified by TD0528] Test 1:The evaluator shall confirm the TOE supports configuration of at least three (3) NTP time sources. The evaluator shall configure at least three NTP servers to support periodic time updates to the TOE. The evaluator shall confirm the TOE is configured to accept NTP packets that would result in the timestamp being updated from each of the NTP servers. The evaluator shall check that the time stamp is updated after receipt of the NTP packets. The purpose of this test to verify that the TOE can be configured to synchronize with multiple NTP servers. It is up to the evaluator to determine that the multi-source update of the time information is appropriate and consistent with the behaviour prescribed by the RFC 1305 for NTPv3 and RFC 5905 for NTPv4.

High-Level Test Description
Launch wireshark to listen to packets inbound/outbound of the TOE interface.

High-Level Test Description
Configure the TOE with three valid NTP servers. In the first phase of the test, the first server will be the active NTP server. The second will be an inactive IP and the third will be an inactive IP as well. The TOE should send network traffic to each of the three defined IP addresses. Only the active NTP server should respond. The TOE should synchronize to the clock. In the second phase of the test, set the first NTP server to be an inactive server IP. Set the second to be the active server and set the third to be another inactive IP. Run the test again. The TOE will still send out to all three IPs but it will still only synchronize with the active server (second). In the final phase of the test, the first two NTP servers are the inactive IPs and the third is the active NTP server. Run the test again and show that the TOE transmits to all three and synchronizes to the active server (third). Verify the findings with the audit logs and packet traffic between the NTP servers and the TOE.
Findings: PASS

234 [Modified by TD0528] Test 2: (The intent of this test is to ensure that the TOE would only accept NTP updates from configured NTP Servers).

235 The evaluator shall confirm that the TOE would not synchronize to other, not explicitly configured time sources by sending an otherwise valid but unsolicited NTP Server responses indicating different time from the TOE's current system time. This rogue time source needs to be configured in a way (e.g. degrade or disable valid and configured NTP servers) that could plausibly result in unsolicited updates becoming a preferred time source if they are not discarded by the TOE. The TOE is not mandated to respond in a detectable way or audit the occurrence of such unsolicited updates. The intent of this test is to ensure that the TOE would only accept NTP updates from configured NTP Servers. It is up to the evaluator to craft and transmit unsolicited updates in a way that would be consistent with the behaviour of a correctly-functioning NTP server.

High-Level Test Description
Note: This test relies on the built-in spoofing capabilities that have been added to the Lightship custom ntpd. The TOE is configured with 3 legitimate NTP servers. A rogue time source is set up with a network adapter that has the same IP address of the peer NTP server's IP address. This time source also has identical ntp.conf and ntp.keys available as the NTP servers to craft a response packet that would be consistent with the behaviour of a correctly-functioning NTP server. During the test, the network adapter of the peer NTP server will be brought down, to prevent it from responding to the TOE's NTP timestamp request packets. The TOE will send out NTP timestamp requests to all servers once again and while the candidate NTP servers will respond with legitimate packets, peer NTP server will not, due to its interface being brough down earlier. Instead, rogue time source will receive the packet, craft an exact response packet that the original server would and spoof the source address of the response. The TOE should not accept the unsolicited but properly crafted NTP updates from a spoofed IP address. After some time, the TOE should switch the server it syncs from the peer server to another (candidate) NTP server available.
Findings: PASS

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

4.1.3.1 TSS

FCS_TLSC_EXT.1.1

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] / TSS section 6.2.11 specifies the ciphersuites that are supported. The list is consistent with those listed in the component. a) TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 b) TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492

FCS_TLSC_EXT.1.2

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] / TSS section 6.2.11 states that, the TOE supports 2-way certificate authentication with X.509v3 certificates for Syslog communication. Only DNS names are supported as acceptable reference identifiers. For server certificate verification, the TOE compares the reference identifiers to the identifier in the presented server's TLS certificate. The TLS client does not support certificate pinning. Only Syslog Communication supports wildcards.

238 Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS should supplied to the "joining" component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The combination of attribute types, uniquely identify the remote TOE component; types, is sufficient to support unique identification of the maximum supported number of TOE components.

Findings:	The TOE is not a distributed TOE.

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

Findings:	The ST does not claim support for IP addresses in the CN as reference identifiers.

FCS_TLSC_EXT.1.4

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

Findings:	Section 6.2.11 of [ST] / TSS describes the Supported Elliptic Curves Extension as follows: The TLS client will transmit the Supported Elliptic Curves extension in the Client Hello message by default with support for the following NIST curves: P256, P384, P521. The non-TOE server can choose to negotiate the elliptic curve from this set for any of the mutually negotiable elliptic curve ciphersuites.

4.1.3.2 Guidance Documentation

FCS_TLSC_EXT.1.1

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:	The evaluator checked [USER] and confirmed that the TOE does not allow administrator to configure cipher suites, which is consistent with TSS in the [ST].

FCS_TLSC_EXT.1.2

242 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 “TLS Communication” describes all identifiers. [USER] section “General Settings” provides information on SAN extension support as the administrator is allowed to edit the Subject Alternative Name (or SAN) and allowed to generate a CSR that has multiple SAN entries as stated under “Uploading a Custom Server Certificate” settings. Note that for Syslog communication, only DNS names are supported as acceptable reference identifiers. IP addresses are not allowed for reference identity as stated in the [SUPP].

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

Findings:	The TOE is not a distributed TOE.

FCS_TLSC_EXT.1.4

244 If the TSS indicates that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD

guidance includes configuration of the Supported Elliptic Curves/Supported Groups Extension.

Findings:	The evaluator examined the [USER] and verified that supported elliptic curves extension is not configurable.

4.1.3.3 Tests

245 For all tests in this chapter the TLS server used for testing of the TOE shall be configured not to require mutual authentication.

FCS_TLSC_EXT.1.1

246 Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to negotiate all specifically claimed ciphersuites.
Findings: PASS

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

High-Level Test Description
Construct two X.509 certificates: one with an extendedKeyUsage with ‘serverAuth’ and another without. 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 ciphersuite.
Findings: PASS

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

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server using the TLS_NULL_WITH_NULL_NULL (cipher ID 0x0000).
Findings: PASS

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a non-negotiated ciphersuite.
Findings: PASS

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

High-Level Test Description
Force the TOE client to connect to a Lightship TLS server which will use an unsupported EC curve.
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 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. [conditional]: If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the handshake does not finished successfully, and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a mangled key exchange signature.
Findings: PASS

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

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a changed first payload byte in the finished message.
Findings: PASS

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a mangled finished message.
Findings: PASS

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to attempt a handshake with a test server sending a modified nonce value.
Findings: PASS

FCS_TLSC_EXT.1.2

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

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

or

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

or

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

253 Note that for some tests additional conditions apply.

254 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.

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

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

   Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 1.

High-Level Test Description
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 (a CN that does not match the reference identifier and does not contain the SAN extension).
Findings: PASS

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

High-Level Test Description
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 (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).
Findings: PASS

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

Findings:	This test is done under FCS_TLSC_EXT.1.1 Test 2. The server certificate used in that test contains a CN that matches the reference identifier and does not contain the SAN extension. Refer to FCS_TLSC_EXT.1.1 Test 2 for results.

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

High-Level Test Description
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 (a CN that does not match the reference identifier but does contain an identifier in the SAN that matches).
Findings: PASS

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

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

High-Level Test Description
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 (containing a wildcard that is not in the left-most label of the presented identifier).
Findings: PASS

2. [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall

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

High-Level Test Description
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 (containing a wildcard in the left-most label).
Findings: PASS

f. Test 6 [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 6.

Findings:	This test is not applicable as only DNS names are supported as acceptable reference identifiers.

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

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

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

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

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

Findings:	The TOE does not claim FPT_ITT.1 with RFC 5280.

FCS_TLSC_EXT.1.3

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

257 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.

Findings:	This test case is performed as part of FIA_X509_EXT.1 - Test 1.

258 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.

High-Level Test Description
Create a sequence of three new X.509 certificates: a root CA, an intermediate CA signed by the root CA and a leaf node certificate signed by the intermediate CA. Without uploading the new root CA certificate in the TOE's trust store (TOE already has a root CA in its trust store from the evaluated configuration setup), initiate a connection using the new certificate and verify that the validation fails and show that the certificate is not automatically accepted.
Findings: PASS

259 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.

Findings:	N/A. The ST does not claim any additional override mechanisms.

FCS_TLSC_EXT.1.4

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

High-Level Test Description
Force the TOE client to connect to a Lightship TLS server, which will use a supported EC curve.
Findings: PASS

4.1.4 FCS_TLSS_EXT.1 Extended: TLS Server Protocol without mutual authentication

4.1.4.1 TSS

FCS_TLSS_EXT.1.1

261 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 are identical to those listed for this component.

Findings:	[ST] / TSS section 6.2.12 lists the supported ciphersuites. The ciphersuites are identical to those listed for this component. a) TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 b) TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492.

FCS_TLSS_EXT.1.2

262 The evaluator shall verify that the TSS contains a description of how the TOE technically prevents the use of old SSL and TLS versions.

Findings:	[ST] / TSS section 6.2.12 states that the server only allows TLS protocol version 1.2 (rejecting any other protocol version, including SSL 2.0, SSL 3.0, TLS 1.0 and TLS 1.1 and any other unknown TLS version string supplied)

FCS_TLSS_EXT.1.3

263 If using ECDHE or DHE ciphers, the evaluator shall verify that the TSS describes the key agreement parameters of the server Key Exchange message.

Findings:	[ST] / TSS section 6.2.12 states that the TLS server is capable of negotiating ciphersuites that include ECDHE key agreement schemes and the TLS server supports NIST curves P256, P384, P521.

FCS_TLSS_EXT.1.4

264 The evaluator shall verify that the TSS describes if session resumption based on session IDs is supported (RFC 4346 and/or RFC 5246) and/or if session resumption based on session tickets is supported (RFC 5077).

Findings:	[ST] / TSS section 6.2.12 states the TOE supports session resumption based on session IDs according to RFC 5246.

265 If session tickets are supported, the evaluator shall verify that the TSS describes that the session tickets are encrypted using symmetric algorithms consistent with

FCS_COP.1/DataEncryption. The evaluator shall verify that the TSS identifies the key lengths and algorithms used to protect session tickets.

Findings:	The TOE does not support session tickets.

266 If session tickets are supported, the evaluator shall verify that the TSS describes that session tickets adhere to the structural format provided in section 4 of RFC 5077 and if not, a justification shall be given of the actual session ticket format.

Findings:	The TOE does not support session tickets.

267 [Modified by TD0569] If the TOE claims a (D)TLS server capable of session resumption (as a single context, or across multiple contexts), the evaluator verifies that the TSS describes how session resumption operates (i.e. what would trigger a full handshake, e.g. checking session status, checking Session ID, etc.). If multiple contexts are used the TSS describes how session resumption is coordinated across those contexts. In case session establishment and session resumption are always using a separate context, the TSS shall describe how the contexts interact with respect to session resumption (in particular regarding the session ID). It is acceptable for sessions established in one context to be resumable in another context.

Findings:	[ST] / TSS section 6.2.12 indicates that the TOE supports session resumption based on session IDs according to RFC 5246. When a connection is established by resuming a session, new ClientHello.random and ServerHello.random values are hashed with the session's master_secret. Sessions cannot be resumed unless both the client and server agree.

4.1.4.2 Guidance Documentation

FCS_TLSS_EXT.1.1

268 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 (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

Findings:	The evaluator checked [USER] and confirmed that the TOE does not allow administrator to configure cipher suites, which is consistent with TSS in the [ST].

FCS_TLSS_EXT.1.2

269 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	[USER] has no configuration requirement for TLS version, as it only supports TLS 1.2, which is consistent with the [ST]. [USER] section “Viewing and Changing System Settings” also confirms this as it states: *“Vision NPBs only support TLS 1.2 for all HTTPS TLS communication in Web Console, Web API, and AppStack GUI.”*

FCS_TLSS_EXT.1.3

270 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	The evaluator examined the [USER] and verified that supported elliptic curves extension is not configurable.

[Modified by TD0569] FCS_TLSS_EXT.1.4

271 The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Findings:	The evaluator examined the [USER] and verified that no configuration is necessary for the TOE to meet the requirement for session resumption based on session IDs.

4.1.4.3 Tests

FCS_TLSS_EXT.1.1

272 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 client, connect to the TOE using the claimed ciphersuites.
Findings: PASS

273 Test 2: The evaluator shall send a Client Hello to the server with a list of ciphersuites that does not contain any of the ciphersuites in the server's ST and verify that the server denies the connection. Additionally, the evaluator shall send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the server denies the connection.

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using an unsupported ciphersuite. Then connect to the TOE using TLS_NULL_WITH_NULL_NULL.
Findings: PASS

274 Test 3: The evaluator shall perform the following modifications to the traffic:

a. Modify a byte in the Client Finished handshake message, and verify that the server rejects the connection and does not send any application data.

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE and modify the first payload byte in the Client Finished message.
Findings: PASS

b. (Test Intent: The intent of this test is to ensure that the server's TLS implementation immediately makes use of the key exchange and authentication algorithms to: a) Correctly encrypt (D)TLS Finished message and b) Encrypt every (D)TLS message after session keys are negotiated.)

The evaluator shall use one of the claimed ciphersuites to complete a successful handshake and observe transmission of properly encrypted application data. The evaluator shall verify that no Alert with alert level Fatal (2) messages were sent.

The evaluator shall verify that the Finished message (Content type hexadecimal 16 and handshake message type hexadecimal 14) is sent immediately after the server's ChangeCipherSpec (Content type hexadecimal 14) message. The evaluator shall examine the Finished message (encrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 11 22 33 44 55...) and confirm that it does not contain unencrypted data (unencrypted example in hexadecimal of a TLS record containing a Finished message, 16 03 03 00 40 14 00 00 0c...), by verifying that the first byte of the encrypted Finished message does not equal hexadecimal 14 for at least one of three test messages. There is a chance that an encrypted Finished message contains a hexadecimal value of '14' at the position where a plaintext Finished message would contain the message type code '14'. If the observed Finished message contains a hexadecimal value of '14' at the position where the plaintext Finished message would contain the message type code, the test shall be repeated three times in total. In case the value of '14' can be observed in all three tests it can be assumed that the Finished message has indeed been sent in plaintext and the test has to be regarded as 'failed'. Otherwise it has to be assumed that the observation of the value '14' has been due to chance and that the Finished message has indeed been sent encrypted. In that latter case the test shall be regarded as 'passed'.

High-Level Test Description
Perform a successful handshake using one of the accepted ciphersuites and verify that the Server Finished message is encrypted.
Findings: PASS

FCS_TLSS_EXT.1.2

275 The evaluator shall send a Client Hello requesting a connection for all mandatory and selected protocol versions in the SFR (e.g. by enumeration of protocol versions in a test client) and verify that the server denies the connection for each attempt.

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE and attempt to negotiate SSL 1.0, SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1 and TLS 1.2. The TOE shall deny the connection for the unsupported protocols claimed in the SFR.
Findings: PASS

FCS_TLSS_EXT.1.3

276 Test 1: [conditional] If ECDHE ciphersuites are supported:

a) The evaluator shall repeat this test for each supported elliptic curve. The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single supported elliptic curve specified in the Elliptic Curves Extension. The Evaluator shall verify (though a packet capture or instrumented client) that the TOE selects the same curve in the Server Key Exchange message and successfully establishes the connection.

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using a valid ECDHE ciphersuite and curve combination and verify that the public key size that comes back in the Server Key Exchange message matches the expected bit size for the chosen curve. Note that the TOE does not support DHE / RSA for key exchange.
Findings: PASS

b) The evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single unsupported elliptic curve (e.g. secp192r1 (0x13)) specified in RFC4492, chap. 5.1.1. The evaluator shall verify that the TOE does not send a Server Hello message and the connection is not successfully established.

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using a valid ECDHE ciphersuite and an unsupported curve and verify that the TOE fails to send back a Server Hello message and terminates the connection.
Findings: PASS

277 Test 2: [conditional] If DHE ciphersuites are supported, the evaluator shall repeat the following test for each supported parameter size. If any configuration is necessary, the evaluator shall configure the TOE to use a supported Diffie-Hellman parameter size. The evaluator shall attempt a connection using a supported DHE ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a Server Key Exchange Message where p Length is consistent with the message are the ones configured Diffie-Hellman parameter size(s).

Findings:	Test not applicable. The TOE does not support DHE ciphersuites.

278 Test 3: [conditional] If RSA key establishment ciphersuites are supported, the evaluator shall repeat this test for each RSA key establishment key size. If any configuration is necessary, the evaluator shall configure the TOE to perform RSA key establishment using a supported key size (e.g. by loading a certificate with the appropriate key size). The evaluator shall attempt a connection using a supported RSA key establishment ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a certificate whose modulus is consistent with the configured RSA key size.

Findings:	Test not applicable. The TOE does not support RSA ciphersuites.

FCS_TLSS_EXT.1.4

279 Test Objective: To demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).

280 Test 1 [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.
b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).
c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps:
Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.
d) The client completes the TLS handshake and captures the SessionID from the ServerHello.
e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).
f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

281 [Modified by TD0569] Remark: If multiple contexts are supported for session resumption, the session ID or session ticket may be obtained in one context for resumption in another context. It is possible that one or more contexts may only permit the construction of sessions to be reused in other contexts but not actually permit resumption themselves. For contexts which do not permit resumption, the evaluator is required to verify this behaviour subject to the description provided in the TSS. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Findings:	Test not applicable. The TOE supports session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2).

282 Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).
b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

283 [Modified by TD0569] Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ID may be obtained in one

context for resumption in another context. There is no requirement that the session ID be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ID constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

High-Level Test Description
Show that the TOE will handle session resumption via Session IDs as per the provided test steps. On Lightship developed TLS client: a) Initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages. b) Initiate a handshake and capture the TOE-generated session ID in the Server Hello message. Then disrupt the handshake by generating an unencrypted fatal Alert message. Finally, initiate a new Client Hello using the previously captured session ID, and verify that the server implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake.
Findings: PASS

284 Test 3 [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) [Modified by TD0556] The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

285 [Modified by TD0569] Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ticket may be obtained in one context for resumption in another context. There is no requirement that the session ticket be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ticket constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

Findings:	Test not applicable. The TOE does not support session tickets.

4.2 Identification and Authentication (FIA)

4.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

4.2.1.1 TSS

286 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] / TSS section 6.3.6 lists where the check of validity of the certificates takes place. The TOE performs X.509 certificate validation at the following points: a) TOE TLS client validation of server X.509 certificates; b) When certificates are loaded into the TOE, such as when importing CAs, certificate responses and other device-level certificates TSS also identifies the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE. The TOE does not provide X.509 Certificate Validation on trusted updates, firmware integrity self-tests or client authentication, the code-signing and clientAuthentication purpose is not checked in the extendedKeyUsage for related certificates.

287 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] / TSS section 6.3.6 provides a list of validation characteristics that the certificates are checked for and claims that certificate revocation checking for those scenarios is performed using a CRL.

288 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:	[ST] / TSS section 6.3.6 states that X.509 certificates are checked during TLS client validation for the purpose of server authentication. If, during the entire trust chain verification activity, any certificate under review fails a verification check, then the entire trust chain is deemed untrusted.

4.2.1.2 Guidance Documentation

289 The evaluator shall also ensure that the guidance documentation describes where the check of validity of the certificates takes place, describes any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) and describes how certificate revocation checking is performed and on which certificate.

Findings:	[USER] section “Valid Certificate Requirements” provides information on the requirements for the TOE to accept certificates uploaded to it, that matches the claims in the [ST] / TSS. [USER] section “Viewing and Changing System Settings” states how the TOE verifies certificates and how to allow the TOE to check CRLs for certificate revocation checking. *“For every certificate on an NPB, you need to get the Certificate Distribution Points (CDP) where the Certificate Revocation List (CRL) is located, and include them in the allowlist”*. [USER] does not contain any information on extendedKeyUsage field rules.

4.2.1.3 Tests

290 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 that sends back a certificate chain in the Server Certificate message and show that the connection is accepted. Remove the root CA from the TOE trust store. Force the TOE to connect to a TLS server to show that the connection is no longer 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 expired root CA certificate. Show that loading expired root CA certificate in the trust store fails. Create an expired server certificate and expired intermediate CA certificate. Force the TOE to connect to a TLS server with either expired server certificate or expired intermediate CA certificate and show both cases are not accepted.
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. Verify that a certificate results in a successful connection. Revoke the server certificate and place into the CRL. Verify the connection now fails due to the certificate being revoked. Revoke the intermediate CA and place into the CRL. Verify the connection now fails due to the 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
Create root ca and intermediate CA cert without cRLsign key usage. Upload a root ca cert without cRLsign key usage to the TOE. It should fail. Listen for a connection with an intermediate CA cert without cRLsign key usage from the server side. Try to connect TOE to it, but TLS connection shall not be established.
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 certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back an X.509 certificate in which the last byte of the certificate (the signature) is modified.
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

h. [Modified by TD0527] Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall conduct the following tests:

[Modified by TD0527] Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

[Modified by TD0527] Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key

information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

[Modified by TD0527] Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

Findings:	Tests not applicable. The TOE provides no support for EC certificates as indicated in FCS_COP.1/SigGen.

291 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.

292 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).

293 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
Clone the known good CA certificate and remove the basicConstraints extension. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings: PASS

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

which will be retrieved from the TOE itself when validating future certificate chains).

High-Level Test Description
Clone the known good CA certificate and set the basicConstraints extension to have the CA flag set to FALSE. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings: PASS

294 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:	This is done as required.

4.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication

4.2.2.1 TSS

295 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:	[ST] / TSS section 6.3.7 indicates that there is a central certificate store for certificates to be stored and examined as part of the validation process. Instructions for configuring the trusted IT entities to supply appropriate X.509 certificates are captured in the guidance documents.

296 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:	[ST] / TSS section 6.3.7 states that if a CRL cannot be obtained, the validation will fail.

4.2.2.2 Guidance Documentation

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

Findings:	[USER] section “TLS Security Details” states that: *“The system is supplied with an Keysight signed certification for TLS and a Thawte certification for code signing. You can upload a custom certificate signed by a CA for better TLS security.”* [USER] chapters “Viewing and Changing System Settings”, “Authentication, Authorization, and Accounting (AAA)” and “Syslog Support” then refer to in detail subsections on generating and uploading a client certificate, uploading a server trusted root certificate, and confirming connections after the certificates are in place.

4.2.2.3 Tests

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

299 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
Attempt to initiate a connection after terminating CRL distribution server. Verify that when the TOE cannot establish a connection to determine the validity of a certificate, the certificate is not accepted.
Findings: PASS

4.2.3 FIA_X509_EXT.3 Extended: X509 Certificate Requests

4.2.3.1 TSS

300 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:	The developer did not select “device specific information”.

4.2.3.2 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:	The evaluator checked [USER] and confirmed that description of X509 certificate requests is in section of “Viewing and Changing System Settings” (for web server certificate) and section of “Syslog Support” (for syslog client certificate), including information on how to generate a CSR.

4.2.3.3 Tests

301 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. Try to import newly generated certificate into the TOE without supplying full CA certificate chain. The certificate cannot be imported because the CA is missing. Then concatenate the newly created leaf certificate with all CA certificates in the CA certificate chain and attempt to reimport. The import is successful.
Findings: PASS

4.3 Security management (FMT)

4.3.1 FMT_MOF.1/Functions Management of security functions behaviour

4.3.1.1 TSS

302 For distributed TOEs see chapter 2.4.1.1 of the [SD]**Error! Reference source not found.**.

Findings:	The TOE is not a distributed TOE.

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

Findings:	[ST] / TSS section 6.4.2 states that TOE restricts the ability to modify (enable/disable) transmission of audit records to an external audit server to Security Administrators.

4.3.1.2 Guidance Documentation

304 For distributed TOEs see chapter 2.4.1.2 of the [SD].

Findings:	The TOE is not a distributed TOE.

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

Findings:	[USER] section “Syslog Support” describes the necessary steps to transmit audit data to an external IT entity, which would be an external syslog server. [USER] section “How local syslog files work - appending and overwriting files” specifies the audit functionality when Local Audit Storage Space is full and no modification of behaviour is supported. [SUPP] also provides information on confirmed log events, syslog configuration and locally stored logs in section “Audit Logging”.

4.3.1.3 Tests

306 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
Sign into the TOE as a non-admin user via Web UI, attempt to change syslog settings, but fail – Modification of syslog settings is disabled.
Findings: PASS

307 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.

308 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
Sign into the TOE as an admin via Web GUI, attempt to change syslog settings and show that the change is permitted. Verify the effect of the change.
Findings: PASS

309 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.

Findings:	This test is not applicable as the ST does not claim the selection of “'handling of audit data” for FMT_MOF.1/Functions SFR.

310 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.

311 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.

Findings:	This test is not applicable as the ST does not claim the selection of “'handling of audit data” for FMT_MOF.1/Functions SFR.

312 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.

Findings:	The ST does not claim this functionality and this test will not be conducted.

313 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.

314 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.

Findings: The ST does not claim this functionality and this test will not be conducted.

315 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.

Findings: The ST does not claim this functionality and this test will not be conducted.

316 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 the Security Administrator authentication shall be successful.

Findings: The ST does not claim this functionality and this test will not be conducted.

4.3.2 FMT_MTD.1/CryptoKeys Management of TSF Data

4.3.2.1 TSS

317 For distributed TOEs see chapter 2.4.1.1 of the [SD].

Findings: The TOE is not a distributed TOE.

318 For non-distributed TOEs, the evaluator shall ensure the TSS lists the keys the Security Administrator is able to manage to include the options available (e.g.

generating keys, importing keys, modifying keys or deleting keys) and how that how those operations are performed.

Findings:	[ST] / TSS section 6.4.4 states that the TOE restricts generation, importation, or deletion of cryptographic keys to Security Administrators.

4.3.2.2 Guidance Documentation

319 For distributed TOEs see chapter 2.4.1.2 of the [SD].

Findings:	The TOE is not a distributed TOE.

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

Findings:	[USER] section “Viewing and Changing System Settings” provides information on how administrators generate a private key using the CSR generation functionality. When a new CSR is created, the process of generating a new public/private key pair also starts which overwrites the existing key pair. [SUPP] “Annex A: Log Reference” provides a sample log for “Generating/import of, changing, or deleting of cryptographic keys”. [USER] “Enable Server FIPS Encryption” section states that, *“Enabling FIPS Encryption ensures FIPS approved crypto algorithms to comply with other FIPS 140-1 crypto requirements including selftest and zeroization“.* The administrators are limited to use the supported options on key generation and zeroization and no additional configuration is necessary when FIPS Encryption mode is enabled.

4.3.2.3 Tests

321 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.

High-Level Test Description
As a non-admin user, attempt to generate a private key using the CSR generation functionality and show it cannot succeed.
Findings: PASS

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

Findings:	Done as part of FIA_X509_EXT.3.

5 Evaluation Activities for Security Assurance Requirements

5.1 ASE: Security Target

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

Findings:	See above sections.

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

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

Findings:	The TOE is not a distributed TOE.

5.2 ADV: Development

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

326 The functional specification describes the TOE Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces.

327 No additional "functional specification" documentation is necessary to satisfy the Evaluation Activities specified in [SD].

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

329 5.2.1.1 Evaluation Activity: The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

Findings:	From section 7.2.1 of the NDcPP: "For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation." The [ST] and [USER] comprise the functional specification. If the test in [SD] cannot be completed because the [ST] or [USER] is incomplete, then the functional specification is not complete and observations are required. During the evaluator's use of the product and its interfaces (Web GUI, Web API and local serial port), there were no areas that were deficient.

330 5.2.1.2 Evaluation Activity: The evaluator shall check the interface documentation to ensure it identifies and describes the parameters for each TSFI that is identified as being security relevant.

Findings:	See comments in the previous work unit.

331 5.2.1.3 Evaluation Activity: The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

Findings:	See comments in the previous work unit.

5.3 AGD: Guidance

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

333 5.3.1.1 Evaluation Activity: The evaluator shall ensure the Operational guidance documentation is distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

Findings:	This TOE is destined for the NIAP Product Compliance List (PCL) which mandates the public distribution of the Common Criteria supplement [SUPP] document. Therefore, there is a reasonable guarantee that administrators and users are aware of the existence and role of this supplementary guide in establishing and maintaining the evaluated configuration.

	The User Guide [USER] documentation is part of the TOE and is also available at https://support.ixiacom.com/

334 5.3.1.2 Evaluation Activity: The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

Findings:	There is only one operational environment claimed in the [ST]. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency.

335 5.3.1.3 Evaluation Activity: The evaluator shall ensure that the Operational guidance contains instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

Findings:	The TOE implements BouncyCastle for TLS/HTTPS, X.509v3 and Key Encryption. OpenSSL is used for NTP services. Relevant Cryptographic Algorithm Validation Program (CAVP) certificates are shown in Table 4 of [SUPP]. No warning is required related to cryptographic engines when the FIPS Encryption mode is enabled as there are no additional cryptographic engines supported that were not evaluated or tested.

336 5.3.1.4 Evaluation Activity: The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.

Findings:	The [SUPP] document covers configuration of the in-scope functionality where additional configuration might be required.

5.3.1.5 Evaluation Activity

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

a) The guidance documentation shall contain instructions for configuring any cryptographic engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

b) [Modified by TD0536] The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps:

1) Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).

2) Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes instructions that describe at least one method of validating the hash/digital signature.

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

Findings:	See work unit 5.3.1.3 in this document for configuration of the cryptographic engine. See work unit 2.5.4.2 in this document for instructions on obtaining the update and initiating the update process including validating the digital signature. See work unit 5.3.1.4 in this document for details as to what was covered by the EAs.

338 5.3.2.1 Evaluation Activity: The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).

Findings:	Please refer to work unit AGD_OPE.1-6.

339 5.3.2.2 Evaluation Activity: The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

Findings:	There is only one operational environment claimed in the [ST]. All TOE platforms claimed in [ST] are covered by the guidance documents [USER] and [INSTALL].

340 5.3.2.3 Evaluation Activity: The evaluator shall examine the preparative procedures to ensure they include instructions to successfully install the TSF in each Operational Environment.

Findings:	See previous work unit.

341 5.3.2.4 Evaluation Activity: The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.

Findings:	The [USER] and [SUPP] documents provide extensive information on managing the security of the TOE as an individual product. Additional best practice guidance provided within those documents helps instil a culture of secure manageability within a larger operational environment.

342 5.3.2.5 Evaluation Activity: In addition the evaluator shall ensure that the following requirements are also met.

The preparative procedures must

a) include instructions to provide a protected administrative capability; and

b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

Findings:	[SUPP] includes instructions to provide a protected administrative capability under "Configuration Guidance" which refers to different sections in [USER] for further information. [USER] section "Configure Government Security Settings" provides information on the default admin credentials for the TOE.

6 Vulnerability Assessment

343 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.

344 [Modified by TD0547] The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

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

345 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: - **Keysight Security Advisories Page**: https://about.keysight.com/en/quality/security/advisory.shtml https://support.ixiacom.com/support-services/security-advisories (Note that credentials are needed for access to the portal) - **NIST National Vulnerabilities Database** https://web.nvd.nist.gov/view/vuln/search - **Common Vulnerabilities and Exposures**: http://cve.mitre.org/cve/ https://www.cvedetails.com/vulnerability-search.php - **Tenable Network Security**: https://www.tenable.com/plugins - **Tipping Point Zero Day Initiative**: http://www.zerodayinitiative.com/advisories - **Offensive Security Exploit Database**: https://www.exploit-db.com/ - **OpenSSL Vulnerabilities**: https://www.openssl.org/news/vulnerabilities.html - **Google** The public survey for vulnerabilities was conducted between September 27, 2021 and January 25, 2022 using the following search terms:

- Vision Series Network Packet Brokers 5.7.1;
- TOE models, such as Vision ONE, Vision 7300/7303, Vision E40, Vision E100, Vision E10S, Vision X, TradeVision;
- Intel Core i7-3555LE, Atom C2538, Xeon D-1518, Xeon D-1527, Celeron 3965U;
- BouncyCastle
- OpenSSL

Other search terms were used but are deemed proprietary and not included here.

The survey also determined if high risk vulnerabilities, such as Log4J, were applicable. These vulnerabilities were deemed not applicable to the TOE.

The evaluation team determined that no residual vulnerabilities exist, based on these searches, that are exploitable by attackers.

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

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.

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.