NETSCOUT

TestStream Management Software v5.3.0 on nGenius 3900 Series Switches

Assurance Activity Report

Version 1.2

April 2023

Document prepared by

www.lightshipsec.com

Table of Contents

1 Introduction

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

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	Canadian Common Criteria Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	NETSCOUT Systems, Inc.
TOE	TestStream Management Software v5.3.0 on nGenius 3900 Series Switches Build: 5.3.0.54
Security Target	TestStream Management Software v5.3.0 on nGenius 3900 Series Switches Security Target, v1.3
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 does not apply 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 does not apply to the TOE as FCS_TLSC_EXT.2 is not claimed.*
	TD0538: NIT Technical Decision for Outdated link to allowed-with list *This TD does not apply to the TOE as this is applicable to Protection Profile.*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.*
	TD0631: NIT Technical Decision for Clarification of public key authentication for SSH Server *This TD applies to the TOE.*
	TD0632: NIT Technical Decision for Consistency with Time Data for vNDs *This TD applies to the TOE.*
	TD0633: NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance *The TOE does not claim FCS_IPSEC_EXT.1.*
	TD0634: NIT Technical Decision for Clarification required for testing IPv6 *The TOE does not claim FCS_TLSC_EXT.1 or FCS_DTLSC_EXT.1.*
	TD0635: NIT Technical Decision for TLS Server and Key Agreement Parameters *This TD applies to the TOE.*
	TD0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH *This TD applies to the TOE.*
	TD0638: NIT Technical Decision for Key Pair Generation for Authentication *This TD applies to the TOE.*
	TD0639: NIT Technical Decision for Clarification for NTP MAC Keys *This TD does not apply as NTP SFR is not claimed.*
	TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing *This TD does not apply as FCS_TLSC_EXT.1 or FCS_TLSC_EXT.2 not claimed.*

Tools

Tool name	Version	Description
OpenSSH	OpenSSH_8.8p1 Debian-1, OpenSSL 1.1.1n 15 Mar 2022	Used for general purpose SSH CLI access
Firefox	106.0	Used as the web browser for web-based management activities

Wireshark	3.6.0	Used to capture network packets
Lightship Security Greenlight (LS Greenlight)	3.0.34+0	Used to provide automated support for TLS server scanning and report generation.
GNU bash	5.1.16(1)-release	Used for advanced scripting support.
OpenSSL	1.0.2h	Used to perform X.509 operations as required by LS Greenlight (this is provided bundled with LS Greenlight).
OpenSSL	OpenSSL 1.1.1m 14 Dec 2021 (Library: OpenSSL 1.1.1n 15 Mar 2022)	Used for CLI as a general-purpose TLS server and TLS client where needed.
nmap	7.9	Used for IP, TCP and UDP port scanning
OpenVAS	GOS v20.08.13	With plugins updated as of 38 days before the publication of this report.
Tcpdump	tcpdump version 4.9.2 libpcap version 1.5.3 OpenSSL 1.0.2k-fips 26 Jan 2017	Packet Capture
MobaXterm	MobaXterm v22.0 build 4858	SSH client

1.3 Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	TestStream Management Software v5.3.0 on nGenius 3900 Series Switches Security Target, v1.3 | April 2023

Ref	Document
[CC_GUIDE]	NETSCOUT TestStream Management Software v5.3.0 on nGenius 3900 Series Switches Common Criteria Guide v1.3 | April 2023
[ADD]	NETSCOUT TestStream Management Software v5.3.0 Common Criteria Addendum Rev 4.10 | February 2023
[ADMIN]	NETSCOUT TestStream Management Software 5.3.0 Administrator Guide 733-1696 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]	TestStream Management Software v5.3.0 on nGenius 3900 Series Switches Entropy Description, v0.10
[CGE]	Canadian Common Criteria Program Guidance for Evaluators, December 2022

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: - **Generate SSH keys**. Action and key reference. - **Generate CSR**. Action and key reference. - **Import Certificate**. Action and key reference. - **Import CA Certificate**. Action and key reference.

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:	Section 5.2 of the [CC_GUIDE] includes a log reference which shows samples of the auditable events.

6 The evaluator shall also make a determination of the administrative actions related to TSF data related to configuration changes. The evaluator shall examine the guidance documentation and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the configuration (including enabling or disabling) of the mechanisms implemented in the TOE that are necessary to enforce the requirements specified in the cPP. The evaluator shall document the methodology or approach taken while determining which actions in the administrative guide are related to TSF data related to configuration changes. The evaluator may perform this activity as part of the activities associated with ensuring that the corresponding guidance documentation satisfies the requirements related to it.

Findings:	The evaluator performed this activity as part of those AAs associated with ensuring the corresponding guidance documentation satisfied their independent requirements. However, overall, the evaluator considered the administrator guides published by the vendor. The evaluator reviewed the contents of the documentation and looked specifically for functionality related to the scope of the evaluation. Vendor also provided [ADD] which includes procedures for functions that are specific to the common criterial evaluation.

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 which states that audit data is transferred to a Syslog server and log events are sent in real-time over SSH as described by FCS_SSHC_EXT.1.

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 amount of audit data that may be stored locally is dependent on the available disk space and only authorized administrators may view audit records and no capability to modify the audit records is provided. Same TSS section states that the logs are rotated as follows: a) /var/log log files. Log files are rotated when a TestStream software update is installed in addition to an algorithm running every ten seconds which addresses the following: if the /var partition reaches 98% utilization or more, the TOE truncates system log files and removes rotated system log files. Otherwise, If the /var partition reaches 90% utilization or more, rotated log files are removed. If the /var partition reaches 50% utilization or more, auth.log and system log files are truncated, otherwise *logrotate* is run.

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 the TOE consists of a single standalone component that stores audit data locally.

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 details the behaviour of the TOE when the storage space for audit data reaches a certain level of utilization: a) /var/log log files. Log files are rotated when a TestStream software update is installed in addition to an algorithm running every ten seconds which addresses the following: if the /var partition reaches 98% utilization or more, the TOE truncates system log files and removes rotated system log files. Otherwise, If the /var partition reaches 90% utilization or more, rotated log files are removed. If the /var partition reaches 50% utilization or more, auth.log and system log files are truncated, otherwise *logrotate* is run. The amount of audit data that may be stored locally is dependent on the available disk space.

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:	[CC_GUIDE] section 3.8 “Audit Log Offloading” and Section ‘Log offloading’ of the [ADD] describes the requirements on the syslog collector and includes the necessary steps to configure syslog collector properly and any configuration required on the TOE as well.

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:	[CC_GUIDE] section 3.8 “Audit Log Offloading” describes the relationship between the local audit data and the audit data that are sent to the audit log server. By default,

	logs are stored locally on the TOE and a log-audit package is required to be installed for secure offloading and auditing logs. Instructions to configure remote log offloading is found in [ADD] section “Log offloading”.

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:	TSS section 6.1.3 describes that audit logs are stored locally and log events are sent in real-time. Log files are rotated on reaching certain storage partition utilization. These correspond to the SFR component selections of FAU_STG_EXT.1.2 and FAU_STG_EXT1.3. No additional configuration options are available as there is only one audit storage and offloading method available on the TOE. Instructions to configure remote log offloading is found in [ADD] section “Log ffloading”.

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.

Note	Verification that the data is encrypted is satisfied by FTP_ITC.1 for the logging channel. The logging server is a CentOS 7 virtual machine running rsyslogd v8.24.0-34.el7 and OpenSSH_7.4p1, OpenSSL 1.0.2k-fips 26 Jan 2017. It requires authentication over an SSH tunnel. 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
Fill the storage location by copying a large file multiple times to /var/, to simulate full storage location. Then examine the behaviour of the TOE when the storage capacity reaches 90% full. The rotated log files are removed.
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

Test Not Applicable	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.

Test Not Applicable	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 TOE supports key generation for the following asymmetric schemes: a) ECC P-256/P-384/P-521. Used in SSH and 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:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

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:	ECDSA key generation is covered by the CAVP certificate #C2144, which has ECDSA KeyGen 186-4 for NIST curves P-256, P-384, P-512. This is consistent with FCS_CKM.1 in the [ST] section 5.3.2.

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 key establishment schemes and the usage for each scheme. This section matches the schemes in section 6.2.1 and the FCS_CKM.1.1 selection in section 5.3.2.

The information provided in section 6.2.2 of the [ST] / TSS is sufficient and includes a mapping of the scheme, SFR, and service.

Table 14: Key Agreement Mapping

SFR	Service	Key Agreement Schemes
FCS_TLSS_EXT.1	GUI / Administration REST API / Administration	ECC
FCS_SSHC_EXT.1	Audit Server	ECC
FCS_SSHS_EXT.1	CLI / Administration	ECC

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:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

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.

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 schemes

The evaluator shall verify the correctness of the TSF's implementation of RSAES-PKCS1-v1_5 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses RSAES-PKCS1-v1_5.

Findings:	RSA-based key establishment schemes are not claimed.

FFC Schemes using "safe-prime" groups

1 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:	FFC Schemes are not claimed.

2.2.3 FCS_CKM.4 Cryptographic Key Destruction

2.2.3.1 TSS

2 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 16 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.

Table 16: Keys

Key/Password	Storage	Zeroization
TLS Private Keys	Persistent - plaintext	Overwritten with zeroes by administrator command.
TLS DH Keys	RAM - plaintext	Overwritten with zeroes upon termination of the session or reboot of the appliance
TLS Session Keys	RAM - plaintext	Overwritten with zeroes upon termination of the session or reboot of the appliance
TLS Session Authentication Keys	RAM - plaintext	Overwritten with zeroes upon termination of the session or reboot of the appliance
SSH Private Keys	Persistent - plaintext	Overwritten with zeroes by administrator command.
SSH DH Keys	RAM - plaintext	Overwritten with zeroes upon termination of the session or reboot of the appliance
SSH Session Keys	RAM - plaintext	Overwritten with zeroes upon termination of the session or reboot of the appliance

Section 6.2.3 of the [ST] / TSS indicates that, the keys held in volatile memory are zeroized after use by overwriting the key storage area with zeroes. Keys held in persistent memory may be destroyed using a Command Line Interface (CLI) command to overwrite files containing keys.

The zeroization methods used are consistent with the selected destruction method in the SFR.

3 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 the [ST] in table 16. Keys held in persistent (non-volatile) memory may

[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.

	be destroyed using a Command Line Interface (CLI) command to overwrite files containing keys as indicated in [ST] / TSS section 6.2.3.

4 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 6.2.3 states that the keys held in volatile memory are zeroized after use by overwriting the key storage area with zeroes. Keys held in persistent memory may be destroyed using a Command Line Interface (CLI) command to overwrite files containing keys. Table 16 shows the origin, storage location and destruction details for cryptographic keys and passwords. Selections in Section 5.3.2 FCS_CKM.4.1 are consistent with the information in Section 6.2.3 FCS_CKM.4. Unless otherwise stated, the keys are generated by the TOE.

5 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 in table 16. The ST does not claim any keys that are stored in non-plaintext format.

6 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.

7 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

8 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.

9 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:	The guidance does not provide any evidence to suggest there are circumstances where keys are prevented or delayed from being cleared.

2.2.3.3 Tests

10 None

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

2.2.4.1 TSS

11 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:	[ST] / TSS section 6.2.4 states that the TOE provides symmetric encryption and decryption capabilities using 128 and 256-bit AES in CBC, CTR and GCM modes, implemented in the protocols TLS and SSH.

2.2.4.2 Guidance Documentation

12 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:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

2.2.4.3 Tests

AES-CBC Known Answer Tests

13 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.

14 **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

[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).

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.

15 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.

16 **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.

17 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.

18 **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].

19 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.

20 **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].

21 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

22 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.

23 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

24 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]
```

25 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.

26 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

27 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.

28 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.

29 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.

30 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

31 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):

32 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.

33 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.

34 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.

35 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].

36 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

37 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

38 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]
```

39 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.

40 There is no need to test the decryption engine.

Findings:	AES encryption and decryption is covered by the CAVP certificate #C2144, which has AES-CBC, CTR and AES-GCM with key sizes of 128, and 256-bits. This is consistent with FCS_COP.1/DataEncryption in the [ST] section 6.2.4

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

2.2.5.1 TSS

41 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 ECDSA Signature Algorithm with NIST curves P-256, P-384, P-521.

2.2.5.2 Guidance Documentation

42 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:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols. Section ‘Utilities’ of the [ADD] describes how to select cryptographic algorithm and key size.

2.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

43 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

44 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

45 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.

46 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

47 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.

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

Findings:	ECDSA Signature Generation and Verification is covered by the CAVP certificate C2144, which has ECDSA SigGen 186-4 implementing NIST curves P-256, P-384, and P-521. This is consistent with FCS_COP.1/SigGen in the [ST] section 5.3.2.

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

2.2.6.1 TSS

49 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 sections 6.2.6 states that the TOE provides services using SHA-1, SHA-256, SHA-384, and SHA-512. SHS is implemented in the following parts of the TSF:

	a) TLS and SSH, b) Hashing of Passwords in non-volatile storage.

2.2.6.2 Guidance Documentation

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

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

2.2.6.3 Tests

51 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.

52 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

53 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

54 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

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

Selected Long Messages Test - Byte-oriented Mode

56 The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m + 8*99*i, where 1 ≤ i ≤ m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages

and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

57 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 CAVP certificate C2144, which has SHA1, SHA2-256, SHA2-384 and SHA2-512. This is consistent with FCS_COP.1/Hash in the [ST] section 5.3.2 Cryptographic Support (FCS).

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

2.2.7.1 TSS

58 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 15 in section 6.2.7 of [ST] / TSS.

Table 15: HMAC Characteristics

Algorithm	Block Size	Key Size	Digest Size
HMAC-SHA-256	512 bits	256 bits	256 bits
HMAC-SHA-384	1024 bits	384 bits	384 bits
HMAC-SHA-512	1024 bits	512 bits	512 bits

2.2.7.2 Guidance Documentation

59 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: [CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

2.2.7.3 Tests

60 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 CAVP certificate C2144, which has HMAC-SHA2-256, HMAC-SHA2-384 and HMAC-SHA2-512. This is consistent with FCS_COP.1/KeyedHash in the [ST] section 5.3.2.

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

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

2.2.8.1 TSS

62 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 6.2.9 states that the TOE contains a CTR_DRBG that is seeded from the software entropy source (Jitter RNG). Entropy from the noise source is extracted 256 bits at a time, conditioned and used to seed the DRBG with 256 bits of full entropy.

2.2.8.2 Guidance Documentation

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

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

2.2.8.3 Tests

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

65 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).

66 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.

67 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:	AES CTR-mode DRBG is covered by the CAVP certificate C2144.

2.3 Identification and Authentication (FIA)

2.3.1 FIA_AFL.1 Authentication Failure Management

2.3.1.1 TSS

68 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 6.3.1 describes that, the TOE is capable of tracking authentication failures of remote administrators and when a user account has sequentially failed authentication after the configured number of attempts, the account will be locked until a Security Administrator resets the account password.

69 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 6.3.1 states that the TestStream local console does not enforce the lockout mechanism for the built in administrator. 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

70 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:	Instructions for configuring the lockout threshold for unsuccessful login attempts can be found in [CC_GUIDE] section 3.6 Administrator Authentication, and [ADMIN] section ‘User Accounts’ subsection ‘Change Security Policy’.

71 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:	[ADMIN] section “User Accounts” subsection ‘Change Security Policy’, as well as the [CC_GUIDE] Section 3.6 provides information on how administrator access will always be maintained. To prevent any administrators from being completely locked out of the server, any account with administrator rights will be allowed to login through the serial port, even if it is locked. Once logged in through the serial port, the administrator can reset its own and other users password.

2.3.1.3 Tests

72 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 GUI, set the login threshold to 3 attempts.
Using the GUI, log into the TOE twice using an incorrect password. On the third attempt, log in correctly and verify that the threshold has not been reached.
Using the GUI, log into the TOE three times using an incorrect password. On the fourth attempt, log in correctly and verify that the threshold has been reached and that the user cannot log in.
Using a secondary workstation with a distinct IP, log into the TOE using the GUI with the correct password. The attempt should fail.
Attempt to log into the local console using the admin account. The attempt should succeed.
Unlock the user’s account.
Attempt to login using the locked out username and correct password. The attempt should succeed.
Repeat the above test using the REST API instead of the GUI.

High-Level Test Description
Repeat the above test using SSH instead of the GUI.
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.

Note:	See previous test case. The ST only claims the use of an administrative unlock.

2.3.2 FIA_PMG_EXT.1 Password Management

2.3.2.1 TSS

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

Findings:	[ST] / TSS section 6.3.2 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 including but not limited to"!", "@", "#", "$", "%", "^", "&", "*" and the additional characters "+", "=", "_", ".", "-". The minimum password length is configurable to between 1 and 30 characters by the Security Administrator. The maximum password length is 95 characters.

2.3.2.2 Guidance Documentation

74 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:	"Change Security Policy" subsection of "User Accounts" in [ADMIN] provides guidance to Security Administrators on how to enforce complex passwords, which includes information on the minimum requirements for a complex password as well. [CC_GUIDE] section 3.6 identifies the password configuration options: 1) The minimum length for administrator passwords is configurable to between 1 and 30 case sensitive characters and may include special characters: @ # $ % ^ & + = _ ! * - . 2) The maximum length for administrator passwords is configurable up to 95 characters.

2.3.2.3 Tests

75 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
Change the password length to be 15 characters. Change the password for the built-in 'admin' user using the identified TSFI. Show that the password can be used to login to the Web GUI local console, SSH and REST API. Change the password for the built-in 'admin' back to a known good password. Change the password length to be 8 characters. Change the password for the built-in 'admin user to be only 7 characters and show it is rejected. Change the password for the built-in 'admin user to be 8 characters and show it is accepted. Change the password of the 'admin1' user. Use the administrator account to reset the 'admin1' users password. Ensure the 'admin1' user can now login using the default netscout1 password.
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.

Note:	Testing for passwords that do not meet the requirements was performed in Test 1.

2.3.3 FIA_UIA_EXT.1 User Identification and Authentication

2.3.3.1 TSS

76 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 6.3.3 and section 6.3.4 of the [ST] collectively provide the requested information. In section 6.3.3 of the [ST], the various interfaces are enumerated with the interface dispositions (local and remote). In section 6.3.4, each interface is described in terms of what constitutes a successful logon. Interactive interfaces prompt the administrator for a username and password credential (including with each REST API session). Only after successful authentication with authorized credentials will the administrator be granted access to the TOE administrative functions. No access is permitted to the TOE unless an administrator is successfully identified and authenticated.

77 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 6.3.3 indicates that the TOE displays a warning banner prior to authentication as described by FTA_TAB.1. This information matches the list of allowed actions prior to requiring the non-TOE entity to initiate the identification and authentication process in FIA_UIA_EXT.1.1 in section 5.3.3 of the [ST]. [ST] / TSS section 6.3.4 covers authentication and identification for local and remote TOE administration. The TOE uses a local password-based authentication mechanism. No access is permitted to the TOE unless an administrator is successfully identified and authenticated. If the logon is successful, the TOE will allow access to administrative functions according to the role in which the administrator is assigned.

78 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.

79 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

80 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 [CC_GUIDE] does not require any preparatory steps for username and password access aside from ensuring that default passwords have been changed. [CC_GUIDE] section 3.5 specifies that Administrator and Root default password values must be changed for security purposes. [ADMIN] section “SSH Access Support on TestStream Management” provides the default passwords for initial login process with Administrator and Root accounts. [ADD] section “Utilities” describes how to enable SSH Root login and how to manage SSH host keys and management keys (used by Java GUI for ssh and sftp to perform management related operations).

2.3.3.3 Tests

81 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
Log into the identified management interface using a known-good credential and logout. Attempt to log into the identified management interface using a known-bad credential and verify that the TOE does not allow the user to be logged in. Ensure the appropriate audit messages appear.
Findings: PASS

b. Test 2: The evaluator shall configure the services allowed (if any) according to the guidance documentation, and then determine the services available to an external remote entity. The evaluator shall determine that the list of services available is limited to those specified in the requirement.

High-Level Test Description
The device does not have any services configured prior to I&A. Access the TOE GUI without logging in and determine the set of services available. Repeat step 1 for SSH and REST API.
Findings: PASS

c. Test 3: For local access, the evaluator shall determine what services are available to a local administrator prior to logging in, and make sure this list is consistent with the requirement.

High-Level Test Description
The device does not have any services configured prior to I&A. Access the TOE serial console without logging in and determine the set of services available.
Findings: PASS

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

Note:	The TOE is not a distributed TOE.

2.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

82 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

83 None.

2.3.5.2 Guidance Documentation

84 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 the guidance documentation and determined that there are no necessary preparatory steps to ensure authentication data is not revealed when it is entered for each local login.

2.3.5.3 Tests

85 The evaluator shall perform the following test for each method of local login allowed:

a. Test 1: The evaluator shall locally authenticate to the TOE. While making this attempt, the evaluator shall verify that at most obscured feedback is provided while entering the authentication information.

High-Level Test Description
Log into the local management interface. Ensure the password field does not echo characters.
Findings: PASS

2.4 Security management (FMT)

2.4.1 General requirements for distributed TOEs

2.4.1.1 TSS

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

Findings:	The TOE is not a distributed TOE.

2.4.1.2 Guidance Documentation

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

Findings:	The TOE is not a distributed TOE.

2.4.1.3 Tests

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

89 **Note:** This is covered in the FMT SFRs.

2.4.2 FMT_MOF.1/ManualUpdate

2.4.2.1 TSS

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

Findings:	The TOE is not a distributed TOE.

2.4.2.2 Guidance Documentation

91 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:	[CC_GUIDE] section 2 refers to: 1) [ADMIN] "About TestStream Management" to verify the current version of software installed on the TOE. 2) [ADMIN] "Updating TestStream Management Servers" and "Updating nGenius 3900 Series Switches": for information on verifying the integrity hash of the firmware

	package and installing an update. Same sections provide warnings regarding interruptions or waiting times prior to login during and after the update.

92 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 is not a distributed TOE.

2.4.2.3 Tests

93 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.

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.
Findings: PASS

94 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.

95 **Note** This test case is covered in FPT_TUD_EXT.1.

2.4.3 FMT_MTD.1/CoreData Management of TSF Data

2.4.3.1 TSS

96 The evaluator shall examine the TSS to determine that, for each administrative function identified in the guidance documentation; those that are accessible through an interface prior to administrator log-in are identified. For each of these functions, the evaluator shall also confirm that the TSS details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.

Findings:	The administrative functions available prior to identification and authentication are described in [ST] / TSS section 6.3.3 FIA_UIA_EXT.1. [ST] / TSS section 6.3.4 and 6.4.3 both indicate that the users are required to login before being provided with access to any administrative functions. The TOE restricts the ability to manage the TSF data to a TestStream user with Administrator rights and linux user ‘tsadmin’.

97 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 6.4.3 states that the TOE restricts the ability to manage the trust store and X.509v3 certificates to Administrators with the appropriate permissions.

2.4.3.2 Guidance Documentation

98 The evaluator shall review the guidance documentation to determine that each of the TSF-data-manipulating functions implemented in response to the requirements of the c PP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

Findings:	As default, there are three accounts available (with default password values): a) Administrator. Default TestStream management account. b) Tsadmin. Account for most maintenance and management activities. c) Root. Account for system maintenance. (Note: SSH root login is disabled when FIPS mode is enabled.) Additional information on these accounts is available in [ADMIN] section “SSH Access Support on TestStream Management” [ADD] section “Administrative user” indicates that for CC compliance certain instructions are listed that may require running scripts and or commands in a Linux shell. In order to do so, the user will be required to open a terminal or a ssh session using the ‘tsadmin’ user (default password for this user is provided in the same section). When creating a new user, [ADMIN] section “User Accounts” specifies various roles with different security levels that limit certain functions to the users. - Administrator - access to all TestStream Management functions and limited diagnostics - Operator - access to all TestStream Management control functions except Switch - Viewer - no access to TestStream Management control functions; monitor and testing only - Diagnostics - full access to all TestStream management diagnostic functions - Custom – Access functions definable.

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

Findings:	The certificate store on the TOE handles certificate authority certificates and user certificates. [CC_GUIDE] section 3.9 “Certificate Store” provides information on the how to install, list and remove CA certificates and also generate self-signed certificates (user) and CSRs.

2.4.3.3 Tests

100 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.

2.4.4 FMT_SMF.1 Specification of Management Functions

101 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.

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

102 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 [CC_GUIDE] section 3.1 indicate that the TOE may be managed via the CLI (console & SSH) or GUI (TLS). REST API is excluded here since its use pertains only to non-TSF administrative functions.

103 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 6.3.3 indicates that Local console (Serial) is used to administer the TOE locally with direct connection to the TOE appliance. This information also matches the [CC_GUIDE] section 3.1: “CLI / Console. Directly connected via Rollover Console cable.” remote local console is provide via the TestStream CLI.

104 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.

2.4.4.2 Guidance Documentation

105 See section 2.4.4.1.

2.4.4.3 Tests

106 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.

High-Level Test Description
The evaluator tests management functions as part of testing the claimed SFRs. 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: PASS

2.4.5 FMT_SMR.2 Restrictions on security roles

2.4.5.1 TSS

107 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 contains details about the TOE supported roles. The section states that the TOE implements role-based access control and supports the following role: a) Security Administrator Local and remote management of the TOE is restricted to Security Administrators.

2.4.5.2 Guidance Documentation

108 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:	[CC_GUIDE] section 3.1 “Administration Interfaces” lists all available local and remote interfaces for administering the TOE. Same section refers to various [ADMIN] sections for detailed steps on maintaining initial access.

2.4.5.3 Tests

109 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.

110

Note	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

111 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:	The storage of the symmetric keys, session keys and private keys is described in [ST] / TSS section 6.5.1 table 16. The TOE claims that the values are stored in plaintext, but plaintext keys cannot be viewed through an interface designed specifically for that purpose.

2.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

2.5.2.1 TSS

112 The evaluator shall examine the TSS to determine that it details all authentication data that are subject to this requirement, and the method used to obscure the plaintext password data when stored. The TSS shall also detail passwords are stored in such a way that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note.

Findings:	[ST] / TSS section 6.5.2 table 17 describes the administrative passwords that are protected and TSS states that in all cases, plaintext passwords cannot be viewed through an interface designed specifically for that purpose. Locally stored Linux passwords are protected at rest using SHA-512 hash and TestStream passwords (PostgreSQL DB) are protected using SHA-1+AES128-CBC encryption.

2.5.3 FPT_TST_EXT.1 TSF testing

2.5.3.1 TSS

113 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:	Detailed self-tests are described in section 6.5.3 of the [ST] / TSS. These tests include firmware integrity check, known answer tests, CPU and BIOS self-tests. These tests ensure the correct operation of the cryptographic functionality of the TOE, the CPU and BIOS and verify firmware integrity. If the CPU, or BIOS tests fail, the device will not complete the boot up operation. Any failure in the firmware integrity tests will prevent the TOE from starting up.

114 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

115 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:	[CC_GUIDE] section 2.3 states that a failure of firmware integrity, known answer tests, CPU and BIOS self tests will result in the TOE not completing the boot up process. Instructions to resolve the failures will be displayed by the TOE.

116 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

117 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.

118 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.

119 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.

120 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 startup includes an indicator that self-tests were executed and passed permitting the device to operate. View self tests log ‘sw_verification_test.log’ and ‘fips_test_suite.log’ in /var/log/
Findings:PASS

2.5.4 FPT_TUD_EXT.1 Trusted Update

2.5.4.1 TSS

121 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 6.5.4 indicates that the firmware version may be queried using the CLI or GUI. The TOE does not claim delayed activation.

122 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 states that the administrator downloads firmware updates from the my.netscout.com website along with a SHA256 file containing the published hash. The Security Administrator must manually verify the that the published hash matches a SHA256 hash of the update file before initiating an update. The TOE does not claim support for digital signature verification of the firmware**.**

123 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 component does not include the options “support automatic checking for updates” or “support automatic updates” in [ST] section 5.3.5. This requirement does not apply.

124 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.

125 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:	Section 6.5.4 of [ST] / TSS indicates that the Security Administrator must manually verify that the published hash matches a SHA256 hash of the update file before initiating an update. It is not a fully automated process.

2.5.4.2 Guidance Documentation

126 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:	[ADMIN] section “About TestStream Management” provides information on how to verify the current version of software installed on the TOE. The TOE does not claim delayed activation.

127 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:	[CC_GUIDE] section 2.2 provides information on how the verification of the authenticity of the update is performed. The TOE software is verified by means of a published hash as follows: a) Download the installation package and SHA256 file from the MasterCare Portal b) Compute a sha256 checksum of the installation package using the linux or windows commandline and compare with the checksum contained in the downloaded SHA256 file. Detailed instructions on the TOE verification procedures can be found in [ADMIN] “Downloading and Verifying the Upgrade/Installation Package” section.

128 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:	Refer to findings above.

129 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.

130 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.

131 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:	The ST does not indicate that a certificate-based mechanism is used.

2.5.4.3 Tests

132 The evaluator shall perform the following tests:

a. Test 1: The evaluator performs the version verification activity to determine the current version of the product. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). The evaluator obtains a legitimate update using procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ('activation' could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

High-Level Test Description
Get the current version of the TOE. Attempt to install a legitimate version of the TOE for the following circumstances: a "down-grade" (since upgrades are not available at this stage). After a successful 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.

Test Not Applicable	The TOE does not verify digital signatures to authorize the installation of images.

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. 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.

133 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.

Test Not Applicable	The TOE does not support published hashes.

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

Note:	The TOE only supports manual updates. The test cases above are not applicable to automatic checking of updates, since there are no images to install during an automatic check.

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

Findings:	The TOE is not a distributed TOE.

2.5.5 FPT_STM_EXT.1 Reliable Time Stamps

2.5.5.1 TSS

136 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 incorporates an internal clock that is used to maintain date and time. The Security Administrator sets the date and time during initial TOE configuration and may change the time during operation.

137 ***[Modified by TD0632]*** If “obtain time from the underlying virtualization system” is selected, the evaluator shall examine the TSS to ensure that it identifies the VS interface the TOE uses to obtain time. If there is a delay between updates to the time on the VS and updating the time on the TOE, the TSS shall identify the maximum possible delay.

Findings:	N/A as “obtain time from the underlying virtualization system” is not selected in the [ST].

2.5.5.2 Guidance Documentation

138 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 setting of time on the TOE can been accomplished by following instructions in [ADMIN] section "Set Server Date/Time". The use of NTP is not claimed and has not been evaluated.

139 ***[Modified by TD0632]*** If the TOE supports obtaining time from the underlying VS, the evaluator shall verify the Guidance Documentation specifies any configuration steps necessary. If no configuration is necessary, no statement is necessary in the Guidance Documentation. If there is a delay between updates to the time on the VS and updating the time on the TOE, the evaluator shall ensure the Guidance Documentation informs the administrator of the maximum possible delay.

Findings:	N/A as "obtain time from the underlying virtualization system" is not selected in the [ST].

2.5.5.3 Tests

140 The evaluator shall perform the following tests:

a. Test 1: If the TOE supports direct setting of the time by the Security Administrator then the evaluator uses the guidance documentation to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

High-Level Test Description
Using the local host computer hosting the GUI, change the date/time in the past by 1 day, 1 hour and 42 minutes. Synchronize the controllers to this new time and verify the time was set properly. Using the local host computer hosting the GUI, change the date/time in the future by 7 days, 1 hour and 06 minutes. Synchronize the controllers to this new time and verify the time was set properly.
Findings: PASS

a. 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.

Findings:	The TOE does not claim use of an NTP server.

b. ***[Modified by TD0632]*** Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay between the setting the time on the VS and when the time is reflected

on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

Findings:	The TOE does not claim obtaining time from the underlying VS.

141 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

142 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 local interactive session (CLI) supports idle timeout termination. The timeout value is configured by the Security Administrator for a specified period of time and disabled by default.

2.6.1.2 Guidance Documentation

143 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:	Session termination on idle time is supported by the TOE on local administrative sessions. “CLI Access Options” (for local console) under [ADMIN] section “Configure Remote Access” has instructions for configuring the inactivity time period for local administrative sessions.

2.6.1.3 Tests

144 The evaluator shall perform the following test:

a. Test 1: The evaluator follows the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator establishes a local interactive session with the TOE. The evaluator then observes that the session is either locked or terminated after the configured time period. If locking was selected from the component, the evaluator then ensures that re-authentication is needed when trying to unlock the session.

High-Level Test Description
For each of 1, 3, 5 minutes: Change the idle timeout to this value; Log into the device; With 30 seconds before the timeout expires, verify the session is still alive by sending a keep alive as described above in the TSFI commands. This should reset the timeout clock. The purpose is to ensure the timeout is not premature. Wait another minute. Verify the session is still alive by sending a keep alive. This should reset the timeout clock. The purpose is to ensure the timeout has been reset by the initial keep alive action above. 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

145 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 6.6.2 states that the Security Administrator may configure the TOE to terminate an inactive remote interactive session via CLI (console, SSH) and TestStream Management GUI following a specified period of time. The default keep-alive timeout interval for the REST API is 15 minutes.

2.6.2.2 Guidance Documentation

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

Findings:	Session termination on idle time is supported by the TOE on remote administrative sessions. [CC_GUIDE] Section 3.2 Remote Access Configuration, as well as, “CLI Access Options” (for remote SSH) and “Web Access Options” under [ADMIN] section “Configure Remote Access” has instructions for configuring the inactivity time period for remote administrative sessions.

2.6.2.3 Tests

147 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 3, 5 minutes: Change the idle timeout to this value; Log into the device; With 30 seconds before the timeout expires, verify the session is still alive by sending a keep alive as described above in the TSFI commands. 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

2.6.3 FTA_SSL.4 User-initiated Termination

2.6.3.1 TSS

148 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 claims that the administrative users may terminate their own sessions at any time. This includes both local and remote sessions since it is not distinctively specified.

2.6.3.2 Guidance Documentation

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

Findings:	[ADMIN] sections “Log Off TestStream Management” and “Exit from TestStream Management” provide information on how to terminate a GUI session. [ADMIN] section “CLI Access - SSH” includes information on terminating a CLI session (local console and remote SSH).

2.6.3.3 Tests

150 For each method of remote administration, the evaluator shall perform the following tests:

a. Test 1: The evaluator initiates an interactive local session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

High-Level Test Description
Log into the serial console Log out using the TSFI previous discussed. Verify that the session has been terminated.
Findings: PASS

b. Test 2: The evaluator initiates an interactive remote session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

High-Level Test Description
Log into the GUI interface.
Log out using the TSFI previous discussed.
Log into the TOE over SSH.
Log out using the TSFI previous discussed.
Log into the REST API.
Log out using the TSFI previous discussed.
Findings: PASS

2.6.4 FTA_TAB.1 Default TOE Access Banners

2.6.4.1 TSS

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

Findings:	[ST] / TSS section 6.6.4 states: Once enabled, the TOE displays an administrator configurable message to all users prior to login at interactive interfaces including the CLI (console, SSH), TestStream Management GUI, and the REST API.

2.6.4.2 Guidance Documentation

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

Findings:	[ADMIN] section “Configure Logon Message” describes how to configure the banner message for GUI and CLI user logon.

2.6.4.3 Tests

153 The evaluator shall also perform the following test:

a. Test 1: The evaluator follows the guidance documentation to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

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

2.7 Trusted path/channels (FTP)

2.7.1 FTP_ITC.1 Inter-TSF trusted channel

2.7.1.1 TSS

154 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 lists the IT entities that the TOE supports secure communication with. The TOE provides a trusted channel to an external audit server (syslog) as per FCS_SSHC_EXT.1. The method of non-TOE endpoint authentication is described in section 6.2.10 of the [ST] / TSS.

2.7.1.2 Guidance Documentation

155 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 TOE claims a trusted channel for remote logging. In [CC_GUIDE] section 3.8, the audit logging trusted channel is described. Detailed instructions on configuring remote log offloading per the steps below can be found in [ADD] section “Log offloading”. Additional information on log offloading functionality can be found in [ADMIN] Section “Configure Syslog”.

2.7.1.3 Tests

156 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.

157 The evaluator shall perform the following tests:

a. Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

Note	The TOE maintains trusted channels to the remote audit log, 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 TOE and restart the audit tunnel. Examine wireshark and verify that the traffic is 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

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. Attempt to log into the TOE over the Web GUI using a predefined user and a bad password. Physically disconnect the remote logging by disconnecting the physical cable that connects the TOE into the engineering switch. Wait 2 minutes. Log into the TOE over the Web GUI using a predefined user and a good password. You will be unable to confirm the login.

High-Level Test Description
Physically reconnect the TOE back into the switch. Log into the TOE over the Web GUI using a predefined user and a good password. Logout. Examine wireshark and verify that the log continues to send encrypted Application Data packets without any user intervention. Repeat the above test with 15 minutes of interruption.
Findings: PASS

Further assurance activities are associated with the specific protocols.

158 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.

159 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

160 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 provides information on the methods of remote administration, including how these communications are protected: a) Remote CLI (SSHv2). Remotely connecting via SSH protocol. b) TestStream Management GUI (TLS). Remotely connecting to the TOE via TLS using the TestStream Management GUI applet. c) REST API (HTTPS). Programmatic administration of TestStream functionality.

2.7.2.2 Guidance Documentation

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

Findings:	[CC_GUIDE] section 3.1 describes how to establish remote administrative sessions. These instructions are further clarified in [ADMIN] under “SSH Access Support on TestStream Management” for remote SSH access, and “Installing and Starting the TestStream Management Client” for GUI. Detailed Rest API usage steps can be found in [ADMIN] “Appendix C TestStream Restful API”

2.7.2.3 Tests

162 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.

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

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

Findings:	Refer to previous test

163 Further assurance activities are associated with the specific protocols.

164 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

165 No optional requirements have been selected by this evaluation.

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

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

Findings:	The TOE web GUI is a Java application downloaded via an HTTPS connection and uses the TLS implementation described by FCS_TLSS_EXT.1 for functional communication once installed. The TOE does not use HTTPS in a client capacity. [ST] / TSS section 6.2.8 provides a description on how the TOE conforms with RFC 2818 which specifies HTTP over TLS. Specifically, section 6.2.8 provides a rationale for the various sections in RFC 2818 for the TOE's compliance.

4.1.1.2 Guidance Documentation

167 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:	[ADMIN] section "Configure Remote Access" instructs the Administrator how to enable HTTPS access and configure port (default: 443) for web access. [ADD] section "Certificate Store" provides instructions on how to install new certificates on the TOE for HTTPS use.

4.1.1.3 Tests

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

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

170 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_SSHC_EXT.1 SSH Client

4.1.2.1 TSS

FCS_SSHC_EXT.1.2

[Modified by TD0636]

171 The evaluator shall check to ensure that the TSS contains a list of the public key algorithms that are acceptable for use for user authentication and that this list is consistent with asymmetric key generation algorithms selected in FCS_CKM.1,

hashing algorithms selected in FCS_COP.1/Hash, and signature generation algorithms selected in FCS_COP.1/SigGen. The evaluator shall confirm the TSS is unambiguous in declaring the TOE's ability to authenticate itself to a remote endpoint with a user-based public key.

172 If password-based authentication method has been selected in the FCS_SSHC_EXT.1.2, then the evaluator shall confirm it is also described in the TSS.

Findings:	[ST] / TSS section 6.2.10 states that the TOE supports public key authentication using ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521 algorithms and will reject all other algorithms. These public key algorithms conform to FCS_SSHC_EXT.1.5 selections in section 5.3.2 of the [ST]. Password-based authentication methods are not selected in the FCS_SSHC_EXT.1.2 selections of the [ST].

FCS_SSHC_EXT.1.3

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

Findings:	[ST] / TSS section 6.2.10 states that the TOE examines the size of each received SSH packet. If the packet is greater than 256KB, it is automatically dropped. This information is consistent with the component FCS_SSHC_EXT.1.3 in section 5.3.2.

FCS_SSHC_EXT.1.4

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

Findings:	[ST] / TSS section 6.2.10 states that the TOE utilises encryption algorithms AES-CTR-128 and AES-CTR-256 for SSH transport implementation which are consistent with the selections of component FCS_SSHC_EXT.1.4 in section 5.3.2. No optional characteristics are defined.

FCS_SSHC_EXT.1.5

[Modified by TD0636]

175 The evaluator shall confirm the TSS describes how a host-key public key (i.e., SSH server's public key) is associated with the server identity.

Findings:	[ST] / TSS section 6.2.10 states that the TOE authenticates the identity of the SSH server by using a local database that contains associations for the host name and corresponding public key.

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

Findings:	[ST] / TSS section 6.2.10 states that the TOE supports public key authentication using ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521 algorithms and will reject all other algorithms which is consistent with the selections of component FCS_SSHC_EXT.1.5 in section 5.3.2. No optional characteristics are defined.

177 If x509v3-based public key authentication algorithms are claimed, the evaluator shall confirm that the TSS includes the description of how the TOE establishes the server's identity and how this identity is confirmed with the one that is presented in the provided certificate. For example, the TOE could verify that a server's configured IP address matches the one presented in the server's x.509v3 certificate.

Findings:	No x509v3-based public key authentication algorithms are claimed.

FCS_SSHC_EXT.1.6

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

Findings:	[ST] / TSS section 6.2.10 states that the TOE supports data integrity MAC algorithms HMAC-SHA2-256 and HMAC-SHA2-512 for SSH transport implementation which are consistent with the selections of component FCS_SSHC_EXT.1.6 in section 5.3.2.

FCS_SSHC_EXT.1.7

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

Findings:	[ST] / TSS section 6.2.10 states that the TOE supports ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521, diffie-hellman-group14-sha256, and diffie-hellman-group16-sha512 for SSH key exchanges which are consistent with the selections of component FCS_SSHC_EXT.1.7 in section 5.3.2.

FCS_SSHC_EXT.1.8

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

a) Both thresholds are checked by the TOE.

b) Rekeying is performed upon reaching the threshold that is hit first.

Findings:	[ST] / TSS section 6.2.10 states that the TOE will re-key SSH connections after 60 minutes or after an aggregate of 500MB of data has been exchanged (whichever occurs first). This is consistent with the component FCS_SSHC_EXT.1.8 in section 5.3.2.

4.1.2.2 Guidance Documentation

FCS_SSHC_EXT.1.2

[Modified by TD0636]

181 The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections initiated by the TOE.

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. No additional configuration is required by the administrator to ensure that only the allowed mechanisms (public-key) are used in SSH connections initiated by the TOE.

FCS_SSHC_EXT.1.4

182 The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHC_EXT.1.5

183 The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHC_EXT.1.6

184 The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the "none" MAC algorithm is not allowed).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols. Specifically, the "none" MAC algorithm is not allowed and cannot be configured to be allowed.

FCS_SSHC_EXT.1.7

185 The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections with the TOE.

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHC_EXT.1.8

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

Findings:	The ability to configure thresholds for SSH rekeying is not claimed in the ST.

4.1.2.3 Tests

FCS_SSHC_EXT.1.2

[Modified by TD0636]

187 Test objective: The purpose of these tests is to check the authentication of the client to the server using each claimed authentication method.

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

NOTE	The TOE accepts ecdsa public key base method which is tested throughout the evaluation.

High-Level Test Description
On the TOE edit the file ts-audit-tunnel-keys.sh and edit the #Generate ssh keys function. Change the -b option to desired size (256, 384. or 512). Run the ts-audit-tunnel-keys.sh -f to generate new key of desired size. Copy the key located in the log-audit/remote directory to the syslog collector in the TestStream Audit directory. On the syslog collector run the ./ts-audit-tunnel-remote-setup.sh root On the TOE run ts-audit-tunnel-config.sh root 10.100.1.95 <fingerprint outputted from ./ts-audit-tunnel-remote-setup.sh root on the syslog collector> Examine the logs.
Findings: PASS

189

190 Test 2: [Conditional] If password-based authentication method has been selected in the FCS_SSHC_EXT.1.2, then following the guidance documentation the evaluator shall configure the TOE to perform password-based authentication with a remote SSH server to demonstrate that the TOE can successfully authenticate using a password as an authentication method.

Test Not Applicable	The TOE does not claim password based authentication for SSHC.

FCS_SSHC_EXT.1.3

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

High-Level Test Description
Permit the TOE SSH client to connect to a test SSH daemon. Once the test client has connected and authenticated, transmit a packet containing less than the maximum packet size bytes and show that the packet is not discarded. Then transmit a packet containing more than the maximum packet size and show that the packet is discarded.
Findings: PASS

FCS_SSHC_EXT.1.4

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

High-Level Test Description
Permit the TOE client to connect to a test SSH server and capture the TOE client's advertised supported cipher algorithms. Verify that the advertised set matches the claimed set. Forcibly use an SSH server to permit connections from the TOE client using only one of those claimed ciphers and show that the connection is successful.
Findings: PASS

FCS_SSHC_EXT.1.5

193 Test 1: The evaluator shall establish a SSH connection using each of the public key algorithms specified by the requirement to authenticate an SSH server to the TOE. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test. Test objective: The purpose of this positive test is to check the authentication of the server by the client (when establishing the transport layer connection), and not for checking generation of the authentication message from the client (in the User Authentication Protocol). The evaluator shall therefore establish sufficient separate SSH connections (with an appropriately configured server) to cause the TOE to demonstrate use of all public key algorithms claimed in FCS_SSHC_EXT.1.5 in the ST.

High-Level Test Description
Use the TOE client and connect to a test SSH server which only provides a host key using the specified public key algorithms in turn. Show that the client authenticates the host.
Findings: PASS

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

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

High-Level Test Description
Ensure the TOE has a supported public/private key pair. Off-TOE, use a different public key (generated with the same public key algorithm). Permit the TOE client to connect to the non-TOE server. The connection attempt should fail.
Findings: PASS

FCS_SSHC_EXT.1.6

196 Test 1: [conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST] The evaluator shall establish an SSH connection using each of the algorithms, except “implicit”, specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

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

High-Level Test Description
Using an SSH Server, forcibly permit only the claimed integrity algorithms and show that connections by the TOE SSH client are accepted to form a successful connection.
Findings: PASS

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

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

High-Level Test Description
Using an SSH Server, forcibly permit an integrity algorithm which is not claimed by the TOE and show that a TOE SSH client connection results in a failed connection.
Findings: PASS

FCS_SSHC_EXT.1.7

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

High-Level Test Description
Using an SSH server, forcibly permit only one claimed key exchange mechanism at a time and show that the TOE client will successfully connect using that algorithm.
Findings: PASS

FCS_SSHC_EXT.1.8

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

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

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

High-Level Test Description
Start packet capture on the syslog server. Wait for the configured rekey time duration to pass. Inspect the packet capture for rekey message.
Findings: PASS

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

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

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

High-Level Test Description
Using a custom SSH server, permit the TOE client to connect to the server. The server will send large amounts of data over the channel back to the client. Ensure that the TOE rekeys before 1 GB in the aggregate has been transmitted. Ensure that the TOE is responsible for sending the rekey initiation.
Findings: PASS

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

208

Findings:	Thresholds are not configurable.

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

a) An argument is present in the TSS section describing this hardware-based limitation and

b) All hardware components that are the basis of such argument are definitively identified in the ST. For example, if specific Ethernet Controller or WiFi radio chip is the root cause of such limitation, these chips must be identified.

Findings:	The TOE does not have hardware limitations.

FCS_SSHC_EXT.1.9

210 Test 1: The evaluator shall delete all entries in the TOE's list of recognized SSH server host keys and, if selected, all entries in the TOE's list of trusted certification authorities. The evaluator shall initiate a connection from the TOE to an SSH server. The evaluator shall ensure that the TOE either rejects the connection or displays the SSH server's public key (either the key bytes themselves or a hash of the key using any allowed hash algorithm) and prompts the Security Administrator to accept or deny the key before continuing the connection.

High-Level Test Description
Clear the known host key database. Using the TOE SSH client, connect to an SSH server and show that the TOE either warns the administrator that the host is unknown or that it rejects the connection attempt until after the host key has been manually added.
Findings: PASS

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

High-Level Test Description
Add a host key to the known hosts database either explicitly or implicitly depending on the mechanism for inserting the key. Generate a different host key for the non-TOE SSH server. Using the TOE SSH client, connect to the SSH server that advertises the wrong host key and show that the TOE rejects the connection. Verify that passwords (if claimed) are not transmitted; verify public key authentication fails.
Findings: PASS

4.1.3 FCS_SSHS_EXT.1 SSH Server

4.1.3.1 TSS

FCS_SSHS_EXT.1.2

[Modified by TD0631]

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

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

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE supports password-based or public key authentications using ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and ecdsa-sha2-nistp521 algorithms and will reject all other algorithms. These public key algorithms conform to FCS_SSHS_EXT.1.5 selections in section 5.3.2 of the [ST]. In the case of public keys, the TOE verifies the identity of the client using entries listed in the local authorized_keys file which identifies authorized hosts and maps them to their corresponding public key.

FCS_SSHS_EXT.1.3

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE examines the size of each received SSH packet. If the packet is greater than 256KB, it is automatically dropped. This information is consistent with the component FCS_SSHS_EXT.1.3 in section 5.3.2.

FCS_SSHS_EXT.1.4

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE utilises encryption algorithms AES-CTR-128 and AES-CTR-256 for SSH transport implementation which are consistent with the selections of component FCS_SSHS_EXT.1.4 in section 5.3.2. No optional characteristics are defined.

FCS_SSHS_EXT.1.5

[Modified by TD0631]

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE supports public key authentication using ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and ecdsa-sha2-nistp521 algorithms and will reject all other algorithms which is consistent with the selections of component FCS_SSHS_EXT.1.5 in section 5.3.2.

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE verifies the identity of the client using entries listed in the local authorized_keys file which identifies authorized hosts and maps them to their corresponding public key.

FCS_SSHS_EXT.1.6

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE supports data integrity MAC algorithms HMAC-SHA2-256 and HMAC-SHA2-512 for SSH transport implementation which are consistent with the selections of component FCS_SSHS_EXT.1.6 in section 5.3.2.

FCS_SSHS_EXT.1.7

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

Findings:	[ST] / TSS section 6.2.11 states that the TOE supports ecdh-sha2-nistp256, ecdh-sha2-nistp384, ecdh-sha2-nistp521, diffie-hellman-group14-sha1, diffie-hellman-group14-sha256, and diffie-hellman-group16-sha512 for SSH key exchanges which are consistent with the selections of component FCS_SSHS_EXT.1.7 in section 5.3.2.

FCS_SSHS_EXT.1.8

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

a) Both thresholds are checked by the TOE.

b) Rekeying is performed upon reaching the threshold that is hit first.

Findings:	[ST] / TSS section 6.2.11 states that the TOE will re-key SSH connections after 60 minutes or after an aggregate of 500MB of data has been exchanged (whichever occurs first). This is consistent with the component FCS_SSHS_EXT.1.8 in section 5.3.2.

4.1.3.2 Guidance Documentation

FCS_SSHS_EXT.1.4

222 The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHS_EXT.1.5

223 The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHS_EXT.1.6

224 The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed data integrity algorithms are used in SSH connections with the TOE (specifically, that the "none" MAC algorithm is not allowed).

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols. Specifically, the "none" MAC algorithm is not allowed and cannot be configured to be allowed.

FCS_SSHS_EXT.1.7

225 The evaluator shall also check the guidance documentation to ensure that it contains instructions to the Security Administrator on how to ensure that only the allowed key exchange algorithms are used in SSH connections with the TOE.

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_SSHS_EXT.1.8

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

Findings:	The limits are not configurable.

4.1.3.3 Tests

FCS_SSHS_EXT.1.2

[Modified by TD0631]

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

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

High-Level Test Description
Using an SSH client, connect to the TOE server using the specified public key algorithms in turn. This requires the TOE to be loaded with a public key corresponding to the key pair. Observe the successful completion of the SSH authentication protocol.
Findings: PASS

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

High-Level Test Description
Using an SSH client, connect to the TOE server using a private key that does not match the public key half loaded in the TOE. The TOE will reject the authentication attempt.
Findings: PASS

230 Test 3: [Conditional] If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, the evaluator shall configure the TOE to accept password-based authentication and demonstrate that user authentication succeeds when the correct password is provided by the connecting SSH client.

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

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

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

FCS_SSHS_EXT.1.3

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

High-Level Test Description
Using a custom tool, transmit a packet larger than the expected TOE buffer size and show that the TOE rejects the packet in some way.
Findings: PASS

FCS_SSHS_EXT.1.4

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

High-Level Test Description
Using an SSH client, connect to the TOE server and capture the TOE server's advertised supported cipher algorithms. Verify that the advertised set matches the claimed set. Forcibly use an SSH client to connect using only one of those ciphers and show that the connection is successful.
Findings: PASS

FCS_SSHS_EXT.1.5

[Modified by TD0631]

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

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

High-Level Test Description
Use an SSH client to confirm that the client can authenticate the TOE host public key using the claimed host public key algorithm.
Findings: PASS

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

237 Test 2: The evaluator shall configure a non-TOE SSH client to only allow it to authenticate an SSH server host public key algorithm that is not included in the ST

selection. The evaluator shall attempt to establish an SSH connection from the non-TOE SSH client to the TOE SSH server and observe that the connection is rejected.

High-Level Test Description
Use an SSH client to confirm that the connection is rejected when the client attempts to authenticate the TOE via a host public key algorithm that is not claimed in the ST.
Findings: PASS

FCS_SSHS_EXT.1.6

238 Test 1: [conditional, if an HMAC or AEAD_AES_*_GCM algorithm is selected in the ST] The evaluator shall establish an SSH connection using each of the algorithms, except "implicit", specified by the requirement. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

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

High-Level Test Description
Using an SSH client, forcibly negotiate only the claimed integrity algorithms and show that they are accepted to form a successful connection.
Findings: PASS

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

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

High-Level Test Description
Using an SSH client, forcibly negotiate an integrity algorithm which is not claimed by the TOE and show that it results in a failed connection.
Findings: PASS

FCS_SSHS_EXT.1.7

242 Test 1: The evaluator shall configure an SSH client to only allow the diffie-hellman-group1-sha1 key exchange. The evaluator shall attempt to connect from the SSH client to the TOE and observe that the attempt fails.

High-Level Test Description
Using an SSH client, forcibly negotiate the diffie-hellman-group1-sha1 key exchange algorithm which is not supported by the TOE and show that it results in a failed connection.
Findings: PASS

243 Test 2: For each allowed key exchange method, the evaluator shall configure an SSH client to only allow that method for key exchange, attempt to connect from the client to the TOE, and observe that the attempt succeeds.

High-Level Test Description
Using an SSH client, forcibly negotiate each of the claimed key exchange algorithms in turn and show that it results in a successful connection.
Findings: PASS

FCS_SSHS_EXT.1.8

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

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

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

High-Level Test Description
Using a custom SSH client, connect to the TOE and trickle data over the channel to avoid disconnection due to idle timeout. Ensure that the TOE rekeys before 1 hour has elapsed. Ensure that the TOE is responsible for sending the rekey initiation.
Findings: PASS

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

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

1 The evaluator shall verify that more data has been transmitted within the SSH session than the threshold allows and a corresponding audit event has been generated by the TOE.

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

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

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

a.

Findings:	These limits are not configurable for this TOE.

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

b. An argument is present in the TSS section describing this hardware-based limitation and

c. All hardware components that are the basis of such argument are definitively identified in the ST. For example, if specific Ethernet Controller or WiFi radio chip is the root cause of such limitation, these chips must be identified.

Findings:	The TOE does not have hardware limitations.

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

4.1.4.1 TSS

FCS_TLSS_EXT.1.1

5 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 states that the server only allows TLS protocol version 1.2 and any other protocol versions are rejected. The TOE is restricted to the following ciphersuites by default: a) TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 b) TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 c) TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 d) TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 This information is consistent with the selections made in the component FCS_TLSS_EXT.1.1 in section 5.3.2 of the [ST].

FCS_TLSS_EXT.1.2

6 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 and any other protocol versions are rejected. This information is consistent with the selections made in the component FCS_TLSS_EXT.1.2 in section 5.3.2 of the [ST]: The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1.

FCS_TLSS_EXT.1.3

[Modified by TD0635]

7 If using ECDHE and/or DHE ciphers, the evaluator shall verify that the TSS lists all EC Diffie-Hellman curves and/or Diffie-Hellman groups used in the key establishment by the TOE when acting as a TLS Server. For example, if the TOE supports TLS_DHE_RSA_WITH_AES_128_CBC_SHA cipher and Diffie-Hellman parameters with size 2048 bits, then list Diffie-Hellman Group 14.

Findings:	[ST] / TSS section 6.2.12 indicates that TLS server performs key establishment for stunnel using ECDHE curve secp256r1, and key establishment for apache using ECDHE curves secp256r1, secp384r1, and secp521r1. This information is consistent with the selections of the component FCS_TLSS_EXT.1.3 in section 5.3.2 of the [ST].

FCS_TLSS_EXT.1.4

8 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 that the TOE supports session resumption based on session tickets that are compliant with RFC5077.

9 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:	[ST] / TSS section 6.2.12 states that the TOE supports session resumption based on session tickets that are compliant with RFC5077. Session tickets are protected using the supported 128-bit and 256-bit AES encryption in CBC, CTR, and GCM modes as described in FCS_COP.1/DataEncryption.in CBC, CTR, and GCM modes as described in FCS_COP.1/DataEncryption.

10 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:	[ST] / TSS section 6.2.12 states that the TOE supports session resumption based on session tickets that are compliant with RFC5077. Session tickets are protected using the supported 128-bit and 256-bit AES encryption in CBC, CTR, and GCM modes as described in FCS_COP.1/DataEncryption.

11 ***[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:	A description of session resumption is indirectly provided in ST] / TSS section 6.2.12, the TOE supports session resumption based on session tickets that are compliant with RFC5077.

4.1.4.2 Guidance Documentation

FCS_TLSS_EXT.1.1

12 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:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_TLSS_EXT.1.2

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

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_TLSS_EXT.1.3

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

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

FCS_TLSS_EXT.1.4

[Modified by TD0569]

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

Findings:	[CC_GUIDE] sections 3.3.1.2 and 3.3.1.3 provide detailed instructions on how to enable/verify FIPS mode on the TOE. Once FIPS mode is enabled no additional configuration is necessary to meet the cryptographic requirements for the in-scope protocols.

4.1.4.3 Tests

FCS_TLSS_EXT.1.1

16 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, and observe the successful negotiation of a ciphersuites.
Findings: PASS

17 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

18 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

19 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 2.0, SSL 3.0, TLS 1.0 and any unsupported, but otherwise valid TLS protocol versions contained in the PP.
Findings: PASS

FCS_TLSS_EXT.1.3

20 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.
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

21 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:	DHE and RSA are not supported by the TOE.

22 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:	The TOE does not support RSA.

FCS_TLSS_EXT.1.4

23 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).

24 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:

[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.

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.

Findings:	The TOE supports session resumption based on session tickets according to RFC5077.

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.

[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.

Findings:	The TOE does not support Session ID.

25 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.

[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.

High-Level Test Description
Show that the TOE will handle session resumption via Session Tickets as per the provided test steps. Start packet capture Use the Lightship gl tools TLS client to connect to the TOE Inspect the packet capture and verify session resumption via session tickets.
Findings: PASS

4.2 Identification and Authentication (FIA)

4.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

4.2.1.1 TSS

26 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 indicates that the TOE performs X.509 certificate validation when certificates are loaded into the TOE, such as when importing CAs, certificate responses and other device-level certificates (such as the web server certificate presented by the TOE TLS web GUI). Certificate revocation checking is performed using OCSP on intermediate CA and leaf certificates at load time and hourly thereafter. TSS also identifies the rules for extendedKeyUsage fields (that are also specified in FIA_X509_EXT.1.1) that are not supported by the TOE. As X.509 certificates are not used for trusted updates, firmware integrity self-tests or client authentication, the code-signing and clientAuthentication purpose is not checked in the extendedKeyUsage for related certificates.

27 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 states that certificate revocation checking is performed using OCSP on intermediate CA and leaf certificates at load time and hourly thereafter. 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

28 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:	[CC_GUIDE] section 3.9 provides information on certificates, and references user to [ADD] for CA and User certificate generation, installation, listing and deletion.

4.2.1.3 Tests

29 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 root CA and intermediate CA (signed by the root). Load the root and intermediate into the TOE trust store. Show that this is successful. Remove the root and intermediate from the TOE trust store. Attempt to load only the intermediate into the TOE trust store and show that this fails.
Findings: PASS

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

High-Level Test Description
Create an X.509 certificate with a ‘not After’ date in the past. Attempt to load it into the TOE. The certificate import shall fail.
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 OCSP has no revoked certificates. Verify that a certificate results in a successful connection. Then revoke the server certificate and restart the OCSP server. Verify the connection now fails due to the certificate being revoked. Then unrevoked the certificate from the OCSP and restart the OCSP server. Revoke the intermediate CA and restart the root CA OCSP server. Verify the connection now fails due to the certificate being revoked. Then unrevoked the intermediate CA and restart the OCSP server. Verify that a certificate now results in a successful connection.
Findings: PASS

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

High-Level Test Description
Load the CA into the TOE trust store. Create an OCSP signing certificate using a known good CA certificate that has the OCSPSigning extendedKeyUsage flag enabled. Clone the known good CA certificate and **remove** the OCSPSigning extendedKeyUsage. The OCSP signature only depends on the (cloned) private key of the CA used to sign it and the TOE does not engage in any certificate pinning. Replace the old CA with the newly cloned CA. Verify the connection now fails due to the OCSP response being signed by a CA without the proper flag.
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
Modify the first eight bytes of the server certificate, and attempt to install it. The installation shall fail to parse.
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
Modify the last 2 bytes of the server certificate, and attempt to install it. The installation shall fail with signature verification error.
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
Using OpenSSL client connect to a Lightship test server which will send back an X.509 certificate in which the public key of the certificate is modified, extract the modified certificate.

High-Level Test Description
Attempt to install the modified certificate on the TOE. The certificate shall not validate.
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.

High-Level Test Description
Construct a chain of three ECDSA certificates: Root CA, intermediate CA, and a Leaf with named curve parameters. Start properly configured OCSP responders. Import the ECDSA Root CA into the trust store. Import the Intermediate CA. Import the leaf and observe that it is successful.
Findings: PASS

[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.

30

Findings:	The TOE does not process CA certificates presented in certificate message

[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.

High-Level Test Description
Construct a chain of three ECDSA certificates: a leaf, an intermediate CA and a trust anchor. Show that the leaf and chain are valid. Create a clone of the Intermediate CA, such that the public key is explicitly defined rather than being a named curve. Show that intermediate CA with explicitly defined parameters is not validated correctly.
Findings: PASS

31 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.

32 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).

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

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

High-Level Test Description
Load a known-good CA into the TOE trust store. Verify that connecting to our test server will yield a successful result. Clone the known good CA certificate and remove the basicConstraints extension. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings: PASS

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

34 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

35 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 states that the instructions for configuring the trusted IT entities to supply appropriate X.509 certificates are captured in the guidance documents. The same section indicates that the TOE has a single X509 trust store where all root CA and intermediate CA certificates can be stored and managed. The trust store is not cached: if a certificate is deleted, it is immediately untrusted. If a certificate is added to the trust store, it is immediately trusted for its given scope. [ST] / TSS section 6.3.6 includes information on how the certificates are checked with a list of validation characteristics and that X.509 certificates are validated using the certificate path validation algorithm defined in RFC 5280.

36 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 indicates that as part of the verification process, OCSP is used to determine whether the certificate is revoked or not. If the OCSP responder cannot be contacted at certificate load time, then the TOE will reject the certificate. The OCSP service will periodically attempt to contact the OCSP responder according to a configurable interval (default is one hour).

4.2.2.2 Guidance Documentation

37 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:	[CC_GUIDE] section 3.9 “Certificate Store” provides information on certificates, and references user to [ADD] for CA and User certificate generation, installation, listing and deletion. Section 3.10 “OCSP Service” provides information on OCSP service installation, configuration, status and on-demand check.

4.2.2.3 Tests

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

39 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
With OCSP responders disabled, ensure that the TOE attempts to rely on the result of an OCSP responder lookup to validate the certificates. Show that when the OCSP responder is unavailable, that any attempt to validate a certificate will fail.
Findings: PASS

4.2.3 FIA_X509_EXT.3 Extended: X509 Certificate Requests

4.2.3.1 TSS

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

Findings:	“Device-specific information” is not selected for the component FIA_X509_EXT.3.1 in section 5.3.3 of the [ST].

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:	[CC_GUIDE] section 3.9.2 and [ADD] section “Generate a Certificate Signing Request” include information on generating a CSR.

4.2.3.3 Tests

41 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 and reimported into the TOE. The certificate is then assigned a purpose, at which point the certificate is validated. If it cannot be validated, it cannot be assigned a purpose and therefore cannot be used.
Findings: PASS

4.3 Security management (FMT)

4.3.1 FMT_MOF.1/Functions Management of security functions behaviour

4.3.1.1 TSS

42 For distributed TOEs see chapter 2.4.1.1.

Findings:	The TOE is not a distributed TOE.

43 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 6.4.2 states that the TOE restricts the ability to modify (enable/disable) transmission of audit records to an external audit server to a TestStream user with Administrator rights and linux user ‘tsadmin’.

4.3.1.2 Guidance Documentation

44 For distributed TOEs see chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

45 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:	Detailed instructions on configuring remote log offloading can be found in [ADD] section “Log offloading”. Additional information on log offloading functionality can be found in [ADMIN] Section “Configure Syslog”.

4.3.1.3 Tests

46 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
Using the built-in ‘monitor’ user (a read-only user), attempt to change the port and IP address of a syslog target and show that the attempt is unsuccessful.
Findings: PASS

47 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.

48 The evaluator does not have to test all possible values of the security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity but at least one allowed value per parameter.

Note:	Successfully accessing the configuration parameters for the syslog server are shown in the previous test. Further examples of a privileged administrator modifying

	the parameters of the SSH tunnel are shown in the test findings for FCS_SSHC_EXT.1.

49 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.

High-Level Test Description
Using the built-in ‘monitor’ user (a read-only user), attempt to change the port and IP address of a syslog target and show that the attempt is unsuccessful.
Findings: PASS

50 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.

51 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
Handling of audit data is not selected.

52

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

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

54 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.

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

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

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

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

57 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 Security Administrator authentication shall be successful.

Note:	The TOE 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

58 For distributed TOEs see chapter 2.4.1.1.

Findings:	The TOE is not a distributed TOE.

59 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 6.4.4 states that the TOE restricts the ability to manage SSH, TLS and any configured X.509 private keys to linux user ‘tsadmin’. This statement implicitly states that these management tasks can only be done via CLI, which the user ‘tsadmin’ is limited to.

4.3.2.2 Guidance Documentation

60 For distributed TOEs see chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

61 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:	[ADD] section “Utilities” provides information on the user ‘tsadmin’ on how to manage the host keys used by SSH daemon and SSH Management key that is used by the GUI client. [ADD] section “Certificate Store” describes the steps for ‘tsadmin’ to generate, import, modify or delete private keys for TLS (Subsections Generate Self-signed Certificate”, “Generating a Certificate Signing Request” and “Install Certificate”). ‘tsadmin’ is limited to the use of the scripts provided based on the information provided.

4.3.2.3 Tests

62 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.

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

High-Level Test Description
Attempt to log into the management port 22 as a non security administrator and delete a certificate from the trust store. The attempt should fail. The privileged case is performed as part of FIA_X509_EXT.3.
Findings: PASS

5 Evaluation Activities for Security Assurance Requirements

5.1 ASE: Security Target

64 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.

65 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

66 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.

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

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

69 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.

70 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 the AGD comprise the functional specification. If the test in [SD] cannot be completed because the [ST] or the AGD is incomplete, then the functional specification is not complete and observations are required. During the evaluator's use of the product and its interfaces (the REST API, Web GUI, SSH CLI, local serial port), there were no areas that were deficient.

71 5.2.1.2 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:	See comments in the previous work unit.

72 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

73 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.

74 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:	The ST states that the TOE includes the following guidance documents in PDF file format which are available through the NETSCOUT Mastercare web portal.

75 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.

76 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:	**[**CC_GUIDE] section 3.3.1.3 provides instructions for configuring FIPS mode which addresses this requirement.

77 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:	[CC_GUIDE] section 3.1 specifies the interfaces used by the TOE in the evaluated configuration.

78 5.3.1.5 Evaluation Activity

79 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.

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

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

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

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

Findings:	[CC_GUIDE] section 3.3.1.3 provides instructions for configuring FIPS mode which addresses this requirement.

	[CC_GUIDE] section 2 provides instructions for the download and verification of the TOE updates. The [CC_GUIDE] provides a list of excluded functions.

80 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:	[CC_GUIDE] section 3 provides instructions for configuration of the TOE and the Operational Environment.

81 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 documentations [ADMIN], [CC_GUIDE] and [ADD].

82 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.

83 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 guidance documentation provides 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.

84 5.3.2.5 Evaluation Activity:

85 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:	[CC_GUIDE] Section 3 provides instructions to provide protected administrative capabilities. Section 3.5 and 3.6 identify TOE passwords that have default values associated with them and instructions are provided for how these can be changed.

5.4 ATE: Tests

5.4.1 Independent Testing – Conformance (ATE_IND.1)

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

87 The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

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

89 Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section A.9.3.1

90

Findings:	The evaluator performed the CEM work units associated with ATE_IND.1 and recorded them in a report provided to the CB.

6 Vulnerability Assessment

91 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.

92 ***[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).

93 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: a) NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search b) Common Vulnerabilities and Exposures: https://cve.mitre.org/cve/ https://www.cvedetails.com/vulnerability-search.php c) US-CERT: https://www.kb.cert.org/vuls/html/search d) CCS – Alerts and advisories: https://cyber.gc.ca/en/alerts-advisories e) NIST NVD API: https://services.nvd.nist.gov/rest/json/cves/2.0?cpeName=cpe:2.3:o:linux:linux_kernel:3.8.13:*:*:*:*:*:*:* Type 1 Hypothesis searches were conducted last on October 24th, 2022. Including search terms for components, and third party programs and libraries. The evaluation team determined based on these searches that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential. Type-2 hypotheses identified for the NDcPP are found in the vulnerability

	assessment. The evaluation team developed Type 3 flaw hypotheses in the vulnerability assessment. No residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.