Cisco Systems, Inc.

Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9

Assurance Activity Report

Version 1.2

November 2022

Document prepared by

www.lightshipsec.com

Table of Contents

1 Introduction

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

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	Canadian Common Criteria Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	Cisco Systems, Inc.
TOE	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9
Security Target	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9 Security Target, v0.19
Protection Profile	collaborative Protection Profile for Network Devices, v2.2E (NDcPP), 23-March-2020 PP-Module for Virtual Private Network (VPN) Gateways (MOD_VPNGW), v1.1, 2020-06-18

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 (CPP_ND_V2.2-SD), December 2019 Supporting Document Mandatory Technical Document PP-Module for Virtual Private Network (VPN) Gateways, v1.1 (mod_vpngw_v1.1-sd), June 2020
Interpretations	**NDcPP v2.2E** TD 0527: Updates to Certificate Revocation Testing (FIA_X509_EXT.1) FIA_X509_EXT.1 EA activities modified accordingly.

TD 0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4 SFR not claimed.
TD 0536: NIT Technical Decision for Update Verification Inconsistency NDcPP v2.2e SD section 5.3.1.5 updated accordingly.
TD 0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3 Application Note 113 in the NDcPP v2.2e modified accordingly.
TD 0538: NIT Technical Decision for Outdated link to allowed-with list Links updated in last paragraph of section 2 of the NDcPP v2.2e.
TD 0546: NIT Technical Decision for DTLS - clarification of Application Note 63 SFR not claimed.
TD 0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN NDcPP SD v2.2 section 5.6.1.1 paragraph 674 updated accordingly.
TD 0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test Potential issue dismissed by NIAP.
TD 0556: NIT Technical Decision for RFC 5077 question SFR not claimed.
TD 0563: NiT Technical Decision for Clarification of audit date information Application Note 3 of FAU_GEN.1.2 updated.
TD 0564: NiT Technical Decision for Vulnerability Analysis Search Criteria NDcPP SD v2.2 paragraph 681 and 682 updated accordingly.
TD 0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7 SFR not claimed.
TD 0570: NiT Technical Decision for Clarification about FIA_AFL.1 FIA_AFL.1 requires at least one remote administrative interface support password authentication.

TD 0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1 This TD applies to the TOE and will be adhered to.
TD 0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers DNS resolution not mandatory for FTP_ITC.1. This TD applies to the TOE and will be adhered to.
TD 0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e Remove DH group 14 from FCS_CKM.2. FCS_CKM.1.1 and FCS_CKM.2.1 evaluation activities updated accordingly.
TD 0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3 Additional ECC option added to FCS_CKM.2.1 SFR. This TD applies to the TOE and will be adhered to.
TD 0591: NIT Technical Decision for Virtual TOEs and hypervisors This TD does not apply to the TOE.
TD 0592: NIT Technical Decision for Local Storage of Audit Records NDcPPv2.2e section A.2.1 updated accordingly.
TD 0631: NIT Technical Decision for Clarification of public key authentication for SSH Server This TD applies to the TOE and will be adhered to.
TD 0632: NIT Technical Decision for Consistency with Time Data for vNDs The TOE is not a vND. This TD does not apply to the TOE.
TD 0633: NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance This TD applies to the TOE and will be adhered to.
TD 0634: NIT Technical Decision for Clarification required for testing IPv6 The TOE does not claim (D)TLS Client. This TD does not apply to the TOE.
TD 0635: NIT Technical Decision for TLS Server and Key Agreement Parameters The TOE does not claim TLS Server.

	This TD does not apply to the TOE. TD 0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH The TOE does not claim SSH Client. This TD does not apply to the TOE. TD 0638: NIT Technical Decision for Key Pair Generation for Authentication This TD applies to the TOE and will be adhered to. TD 0639: NIT Technical Decision for Clarification for NTP MAC Keys This TD applies to the TOE and will be adhered to. TD 0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing The TOE does not claim TLS Client. This TD does not apply to the TOE. **MOD_VPNGW v1.1** TD 0549: Consistency of Security Problem Definition update for MOD_VPNGW_v1.0 and MOD_VPNGW_v1.1 Update to section 6.1.2 of the MOD_VPNGW_v1.1 to include A.CONNECTIONS assumption and OSP statement. TD 0590: Mapping of operational environment objectives Update to section 4.3 to include mapping of A.CONNECTIONS to OE.CONNECTIONS. TD 0597: VPN GW IPv6 Protocol Support FPF_RUL_EXT.1.6 TSS and Test activities updated to include pre-filtering packet filtering behaviour.
Tools	Please refer to the NDcPP2.2E and MOD_VPNGW 1.1 test plans.

1.3 Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9 Security Target, Version 0.19, 24 October 2022
[PP]	collaborative Protection Profile for Network Devices, v2.2E (NDcPP), 23-March-2020
[MOD_VPN]	PP-Module for Virtual Private Network (VPN) Gateways, v1.1 (MOD_VPNGW) 2020-06-18

Ref	Document
[SD]	Evaluation Activities for Network Device cPP, v2.2 (CPP_ND_V2.2-SD), December 2019
[SD_VPN]	Supporting Document Mandatory Technical Document PP-Module for Virtual Private Network (VPN) Gateways, v1.1 (mod_vpngw_v1.1-sd), June 18, 2020
[GUIDE]	Cisco 900 Series Integrated Services Routers Software Configuration Guide, June 6, 2019
[AGD]	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9 Common Criteria Operational User Guidance And Preparative Procedures, v0.9, 28 November 2022
[CMD_REF]	Cisco IOS Security Command Reference A to Z, http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/mcl/allreleasemcl/all-book.html
[TEST]	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9 NDcPP 2.2E Test Plan, Version 0.5, November 2022
[TEST2]	Cisco 900 Series Integrated Services Routers (ISR) running IOS v15.9 VPNGW MOD 1.1 Test Plan, Version 0.4, October 2022
[VULN]	Cisco ISR900 IOSv15.9 NDcPP 2.2E Vulnerability Assessment, Version 0.7, October 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_GEN.1: "Each of the events is specified in syslog records in enough detail to identify the user for which the event is associated, when the event occurred, where the event occurred, the outcome of the event, and the type of event that occurred such as generating keys, including the type of key."

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

Findings:	The TOE is not a distributed TOE.

2.1.1.2 Guidance Documentation

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

Findings:	[AGD] Section 5.2 Reviewing Audited Events – Table 13 provides samples of each auditable event required by FAU_GEN.1.

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.

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.

Findings:	These tests are conducted throughout the test plan.

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:	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:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_STG_EXT.1: "The TOE is configured to export syslog records to a specified, external syslog server in real-time. The TOE protects communications with an external syslog server via IPsec."

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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_STG_EXT.1: "For audit records stored internally to the TOE the audit records are stored in a circular log file where the TOE overwrites the oldest audit records when the audit trail becomes full. The size of the logging files on the TOE is configurable by the administrator with the minimum value being 4096 (default) to 2147483647 bytes of available disk space Refer to the Common Criteria Operational User Guidance and Preparative Procedures for command description and usage information. **"**Only Authorized Administrators are able to clear the local logs, and local audit records are stored in a directory that does not allow administrators to modify the contents."

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:	[ST] Section 6.1 describes the TOE as a standalone TOE that stores the audit records locally in a circular log file that overwrites the oldest audit records when the audit trail becomes full.

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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_STG_EXT.1: "For audit records stored internally to the TOE the audit records are stored in a circular log file where the TOE overwrites the oldest audit records when the audit trail becomes full. "

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 as well as acceptable frequency for the transfer of audit data.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_STG_EXT.1: "The TOE is configured to export syslog records to a specified, external syslog server in real-time. The TOE protects communications with an external syslog server via IPsec. If the IPsec connection fails, the TOE will store audit records on the TOE when it discovers it can no longer communicate with its configured syslog server. When the connection is restored, the TOE will transmit the buffer contents when connected to the syslog server."

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:	[AGD] Section 3.4.3 Remote Logging states, that to protect against audit data loss the TOE must be configured to send the audit records securely (through an IPsec tunnel) to an external TCP syslog server. The configuration of the logging server communication details are found in [AGD] Section 3.4.3 and the description of how to establish the trusted channel is found in [AGD] 3.3.9.1 and 3.3.9.2. The evaluator was able to configure the logging server using the provided guidance.

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:	[AGD] Section 5.2 states, "The ISR900 Series maintains logs in multiple locations: local storage of the generated audit records, and simultaneous offload of those events to the external syslog server."

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:	[AGD] Section 3.4.3 describes how the TOE will log messages to the logging buffer and when the buffer is full, the oldest messages will be overwritten with new messages. This behaviour is consistent with what the configuration claimed in FAU_STG_EXT.1.3.

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 syslog-ng v3.27.1 as described in the Test Setup. Due to the log-forwarding mechanism used on logging server, the audit records are therefore confirmed to have been successfully received by the audit server whenever the test cases are run.

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

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

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

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

High-Level Test Description
Show the local logging event queue. Generate additional events and then review the local logging event queue again. Confirm that older events are dropped off the queue.
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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.1: "The TOE can create an RSA public-private key pair, with a minimum RSA key size of 2048-bit (for CSfC purposes, the TOE is capable of a minimum RSA key size of 3072-bit) and ECDSA key pairs using NIST curves P-256 and P-384." FCS_CKM.1 section of Table 19 also provides the following table to identify the usage for each scheme:

Scheme	SFR	Service
RSA Key generation	FCS_SSHS_EXT.1	SSH Remote Administration
	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity
ECC Key generation Key establishment	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity
FFC Key generation Key establishment	FCS_SSHS_EXT.1	SSH Remote Administration
	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity

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: The Security Target claims RSA 2048-bit or greater, ECC schemes using P-256 and P-384 curves, and FFC scheme using Diffie-Hellman group 14. The [AGD] Section 3.3.6 instructs the administrator how to configure the TOE to use the selected RSA, ECC key generation schemes and key sizes. [AGD] 3.3.8 provides instructions for configuring the FFC scheme using Diffie-Hellman group 14. [AGD] Section 3.3.1 provides instructions on generating RSA keys for use with SSH server trusted path.

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.

FFC Schemes using "safe-prime" groups

[Modified by TD 0580]

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

Findings:	RSA key generation is covered by the following CAVP certificate all of which claim RSA KeyGen 186-4 for 2048-bit and 3072-bit RSA keys: C1802. These claims are consistent with FCS_CKM.1 in the [ST] section 5.3.2. ECDSA key generation is covered by the following CAVP certificate all of which claim ECDSA KeyGen 186-4 for NIST curves P-256 and P-384: C1802. These claims are consistent with FCS_CKM.1 in the [ST] section 5.3.2. FFC EC key generation is covered by the following CAVP certificate all of which claim KAS ECC Component for NIST curves P-256 and P-384: C1802. These claims are consistent with FCS_CKM.1 and FCS_CKM.2 in the [ST] section 5.2.2. Diffie-Hellman group 14 is covered by additional testing below.

2.2.2 FCS_CKM.2 Cryptographic Key Establishment

2.2.2.1 TSS

[Modified by TD 0580]

41 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.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.2: "The TOE implements DH group 14 key establishment schemes that meets RFC 3526, Section 3, Elliptic curve key establishment (conformant to NIST SP 800-56A). The TOE acts as both a sender and receiver for Diffie-Helman based key establishment schemes." FCS_CKM.2 section of Table 19 also provides the following table to identify the usage for each scheme:

Scheme	SFR	Service
RSA Key generation	FCS_SSHS_EXT.1	SSH Remote Administration
	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity
ECC Key generation Key establishment	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity
FFC Key generation Key establishment	FCS_SSHS_EXT.1	SSH Remote Administration
	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity

The evaluator confirmed the key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1.

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.

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: [AGD] section 3.3.1 describes how to configure the TOE to use DH group 14 for SSH server.

[AGD] section 3.3.8.2 describes configuring the selected key establishment schemes for IPsec VPN trusted channels.

2.2.2.3 Tests

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

46 The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of

the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

47 The Function test verifies the ability of the TOE to implement the key agreement schemes correctly. To conduct this test the evaluator shall generate or obtain test vectors from a known good implementation of the TOE supported schemes. For each supported key agreement scheme-key agreement role combination, KDF type, and, if supported, key confirmation role- key confirmation type combination, the tester shall generate 10 sets of test vectors. The data set consists of one set of domain parameter values (FFC) or the NIST approved curve (ECC) per 10 sets of public keys. These keys are static, ephemeral or both depending on the scheme being tested.

48 The evaluator shall obtain the DKM, the corresponding TOE's public keys (static and/or ephemeral), the MAC tag(s), and any inputs used in the KDF, such as the Other Information field OI and TOE id fields.

49 If the TOE does not use a KDF defined in SP 800-56A, the evaluator shall obtain only the public keys and the hashed value of the shared secret.

50 The evaluator shall verify the correctness of the TSF's implementation of a given scheme by using a known good implementation to calculate the shared secret value, derive the keying material DKM, and compare hashes or MAC tags generated from these values.

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

Validity Test

52 The Validity test verifies the ability of the TOE to recognize another party's valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

53 The evaluator shall inject an error in some of the test vectors to test that the TOE recognizes invalid key agreement results caused by the following fields being incorrect: the shared secret value Z, the DKM, the other information field OI, the data to be MACed, or the generated MACTag. If the TOE contains the full or partial (only ECC) public key validation, the evaluator will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

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

Findings: FFC EC key generation is covered by the following CAVP certificate all of which claim KAS ECC Component for NIST curves P-256 and P-384: C1802. These claims are consistent with FCS_CKM.1 and FCS_CKM.2 in the [ST] section 5.3.2.

RSA-based key establishment schemes

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

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

[Modified by TD0580]

FFC Schemes using "safe-prime" groups

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

High-Level Test Description
The TSF's implementation of safe-prime groups was tested in conjunction with protocol testing in FCS_IPSEC_EXT.1.11 Test 1 and FCS_SSHS_EXT.1.7 Test 2 using known good implementations of Strongswan and OpenSSH. Each safe-prime group selected in the ST for each protocol was tested.
Findings: PASS

2.2.3 FCS_CKM.4 Cryptographic Key Destruction

2.2.3.1 TSS

57 The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted for[1]). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

[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.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.4: "See Table 21: TOE Key Zeroization in Section 7 Key Zeroization. The information provided in the table includes all of the secrets, keys and associated values, the description, and the method used to zeroization when no longer required for use." The evaluator reviewed Table 21 and found a list of all relevant keys and their destruction methods.

58 The evaluator shall check to ensure the TSS identifies how the TOE destroys keys stored as plaintext in non-volatile memory, and that the description includes identification and description of the interfaces that the TOE uses to destroy keys (e.g., file system APIs, key store APIs).

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.4: "See Table 21: TOE Key Zeroization in Section 7 Key Zeroization. The information provided in the table includes all of the secrets, keys and associated values, the description, and the method used to zeroization when no longer required for use." The evaluator reviewed Table 21 and found a list of all relevant keys and their destruction methods.

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.4: "See Table 21: TOE Key Zeroization in Section 7 Key Zeroization. The information provided in the table includes all of the secrets, keys and associated values, the description, and the method used to zeroization when no longer required for use." The evaluator reviewed Table 21 and found that keys stored in volatile memory (SDRAM) are automatically overwritten with 0x00 after DH key exchange, IKE session termination, IPsec session termination and SSH session termination. Keys stored in non-volatile memory (NVRAM) can be erased by an administrator using the commands listed in Table 21. Keys stored in non-volatile memory are overwritten by either 0x00 or 0x0d.

60 Where the TSS identifies keys that are stored in a non-plaintext form, the evaluator shall check that the TSS identifies the encryption method and the key-encrypting-key used, and that the key-encrypting-key is either itself stored in an encrypted form or that it is destroyed by a method included under FCS_CKM.4.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_SKP_EXT.1/FPT_APW_EXT.1: "The TOE includes a Master Passphrase feature that can be used to configure the TOE to encrypt all locally defined user passwords using AES. The Master Passphrase is set by using the key config-key password-encryption command. The Master

	Passphrase is stored in a secure directory that cannot be viewed by an Administrator. If the Master Passphrase is lost, it cannot be recovered. The Master Passphrase can be overwritten by creating a new passphrase or deleted by the Administrator.”

61 The evaluator shall check that the TSS identifies any configurations or circumstances that may not conform to the key destruction requirement (see further discussion in the Guidance Documentation section below). Note that reference may be made to the Guidance Documentation for description of the detail of such cases where destruction may be prevented or delayed.

Findings:	The TSS does not identify any configurations or circumstances that may not conform to the key destruction requirement.

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

Findings:	The ST does not specify the use of “a value that does not contain any CSP”.

2.2.3.2 Guidance Documentation

63 A TOE may be subject to situations that could prevent or delay key destruction in some cases. The evaluator shall check that the guidance documentation identifies configurations or circumstances that may not strictly conform to the key destruction requirement, and that this description is consistent with the relevant parts of the TSS (and any other supporting information used). The evaluator shall check that the guidance documentation provides guidance on situations where key destruction may be delayed at the physical layer.

64 For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command[2] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings:	There are no obvious circumstances where delayed or prevented key destruction can occur.

2.2.3.3 Tests

65 None

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

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

2.2.4.1 TSS

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_COP.1/DataEncryption: **"**The TOE provides symmetric encryption and decryption capabilities using AES in GCM and CBC mode (128, 192 and 256 bits) as described in ISO 18033-3, ISO 19772 and ISO 10116 respectively."

2.2.4.2 Guidance Documentation

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

Findings:	[AGD] section 3.3.1 describes how to configure the TOE to use the selected modes and key sizes for the SSH server. [AGD] section 3.3.8.2 describes how to configure the TOE to use the selected modes and key sizes for IPsec.

2.2.4.3 Tests

AES-CBC Known Answer Tests

68 There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

69 **KAT-1.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 plaintext values and obtain the ciphertext value that results from AES-CBC encryption of the given plaintext using a key value of all zeros and an IV of all zeros. Five plaintext values shall be encrypted with a 128-bit all-zeros key, and the other five shall be encrypted with a 256-bit all-zeros key.

70 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using 10 ciphertext values as input and AES-CBC decryption.

71 **KAT-2.** To test the encrypt functionality of AES-CBC, the evaluator shall supply a set of 10 key values and obtain the ciphertext value that results from AES-CBC encryption of an all-zeros plaintext using the given key value and an IV of all zeros. Five of the keys shall be 128-bit keys, and the other five shall be 256-bit keys.

72 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using an all-zero ciphertext value as input and AES-CBC decryption.

73 **KAT-3.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the two sets of key values described below and obtain the ciphertext value that results from AES encryption of an all-zeros plaintext using the given key value and an IV of all zeros. The first set of keys shall have 128 128-bit keys, and the second set shall have 256 256-bit keys. Key *i* in each set shall have the leftmost *i* bits be ones and the rightmost *N-i* bits be zeros, for *i* in [1,N].

74 To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key and ciphertext value pairs described below and obtain the plaintext value that results from AES-CBC decryption of the given ciphertext using the given key and an IV of all zeros. The first set of key/ciphertext pairs shall have 128 128-bit key/ciphertext pairs, and the second set of key/ciphertext pairs shall have 256 256-bit key/ciphertext pairs. Key i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1,N]. The ciphertext value in each pair shall be the value that results in an all-zeros plaintext when decrypted with its corresponding key.

75 **KAT-4.** To test the encrypt functionality of AES-CBC, the evaluator shall supply the set of 128 plaintext values described below and obtain the two ciphertext values that result from AES-CBC encryption of the given plaintext using a 128-bit key value of all zeros with an IV of all zeros and using a 256-bit key value of all zeros with an IV of all zeros, respectively. Plaintext value i in each set shall have the leftmost i bits be ones and the rightmost 128-i bits be zeros, for i in [1,128].

76 To test the decrypt functionality of AES-CBC, the evaluator shall perform the same test as for encrypt, using ciphertext values of the same form as the plaintext in the encrypt test as input and AES-CBC decryption.

AES-CBC Multi-Block Message Test

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

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

AES-CBC Monte Carlo Tests

79 The evaluator shall test the encrypt functionality using a set of 200 plaintext, IV, and key 3-tuples. 100 of these shall use 128 bit keys, and 100 shall use 256 bit keys. The plaintext and IV values shall be 128-bit blocks. For each 3-tuple, 1000 iterations shall be run as follows:

```
# Input: PT, IV, Key
for i = 1 to 1000:
        if i == 1:
                CT[1] = AES-CBC-Encrypt(Key, IV, PT)
                PT = IV
        else:
                CT[i] = AES-CBC-Encrypt(Key, PT)
                PT = CT[i-1]
```

80 The ciphertext computed in the 1000th iteration (i.e., CT[1000]) is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation.

81 The evaluator shall test the decrypt functionality using the same test as for encrypt, exchanging CT and PT and replacing AES-CBC-Encrypt with AES-CBC-Decrypt.

AES-GCM Test

82 The evaluator shall test the authenticated encrypt functionality of AES-GCM for each combination of the following input parameter lengths:

128 bit and 256 bit keys

a. **Two plaintext lengths**. One of the plaintext lengths shall be a non-zero integer multiple of 128 bits, if supported. The other plaintext length shall not be an integer multiple of 128 bits, if supported.

a. **Three AAD lengths**. One AAD length shall be 0, if supported. One AAD length shall be a non-zero integer multiple of 128 bits, if supported. One AAD length shall not be an integer multiple of 128 bits, if supported.

b. **Two IV lengths**. If 96 bit IV is supported, 96 bits shall be one of the two IV lengths tested.

83 The evaluator shall test the encrypt functionality using a set of 10 key, plaintext, AAD, and IV tuples for each combination of parameter lengths above and obtain the ciphertext value and tag that results from AES-GCM authenticated encrypt. Each supported tag length shall be tested at least once per set of 10. The IV value may be supplied by the evaluator or the implementation being tested, as long as it is known.

84 The evaluator shall test the decrypt functionality using a set of 10 key, ciphertext, tag, AAD, and IV 5-tuples for each combination of parameter lengths above and obtain a Pass/Fail result on authentication and the decrypted plaintext if Pass. The set shall include five tuples that Pass and five that Fail.

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

AES-CTR Known Answer Tests

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

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

87 There are four Known Answer Tests (KATs) described below to test a basic AES encryption operation (AES-ECB mode). For all KATs, the plaintext, ~~IV,~~ and ciphertext values shall be 128-bit blocks. The results from each test may either be obtained by the validator directly or by supplying the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

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

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

90 KAT-3 To test the encrypt functionality, the evaluator shall supply a set of key values for each selected keysize as described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the given key values. A set of 128 128-bit keys, a set of 192 192-bit keys, and/or a set of 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N].

91 KAT-4 To test the encrypt functionality, the evaluator shall supply the set of 128 plaintext values described below and obtain the ciphertext values that result from encryption of the given plaintext using each selected keysize with a key value of all zeros (e.g. 256 ciphertext values will be generated if 128 bits and 256 bits are selected and 384 ciphertext values will be generated if all keysizes are selected). Plaintext value i in each set shall have the leftmost bits be ones and the rightmost 128-i bits be zeros, for i in [1, 128]

AES-CTR Multi-Block Message Test

92 The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 less-than i less-than-or-equal to 10 (test shall be performed using AES-ECB mode). For each i the evaluator shall choose a key and plaintext message of length i blocks and encrypt the message, using the mode to be tested, with the chosen key. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key using a known good implementation. The evaluator shall perform this test using each selected keysize.

AES-CTR Monte-Carlo Test

93 The evaluator shall test the encrypt functionality using 100 plaintext/key pairs. The plaintext values shall be 128-bit blocks. For each pair, 1000 iterations shall be run as follows:

```
# Input: PT, Key
for i = 1 to 1000:
CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i]
```

94 The ciphertext computed in the 1000th iteration is the result for that trial. This result shall be compared to the result of running 1000 iterations with the same values using a known good implementation. The evaluator shall perform this test using each selected keysize.

95 There is no need to test the decryption engine.

Findings:	AES encryption and decryption is covered by the following CAVP certificate all of which claim AES-CBC and AES-GCM with key sizes of 128-, 192- and 256-bits: C1802. These claims are consistent with FCS_COP.1/DataEncryption in the [ST] section 5.3.2.

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

2.2.5.1 TSS

96 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_COP.1/SigGen: **"**The TOE provides cryptographic signature services using RSA Digital Signature Algorithm with key size of 2048 and greater as specified in ISO 9796-2, Digital signature scheme 2 or Digital Signature scheme 3. In addition, the TOE will provide cryptographic signature services using ECDSA with key size of 256 or greater as specified in FIPS PUB 186-4, "Digital Signature Standard". The TOE provides cryptographic signature services using ECDSA that meets ISO 14888-3, Section 6.4 with NIST curves P-256 and P-384.**"**

2.2.5.2 Guidance Documentation

97 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:	[AGD] section 3.3.1 describes how to configure the TOE to use the selected cryptographic algorithms and key sizes. "crypto key generate rsa with an RSA key size of 2048 bits [6] Commands A to C -> crypto isakmp aggressive-mode disable -> crypto key generate." [AGD] section 3.3.6.1 describes how to configure the TOE to use the selected cryptographic algorithms and key sizes for IPsec. "RSA and ECDSA keys are generated in pairs, one public key and one private key: (config)# crypto key generate rsa modulus 2048 -or- (config)# crypto key generate ec keysize <256 | 384> exportable"

2.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

98 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

99 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

100 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.

101 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

102 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.

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

Findings:	RSA Signature Generation and Verification is covered by the following CAVP certificate which claims RSA SigGen 186-4 using PKCS 1.5 and 2048- and 3072-bit RSA keys as well as RSA SigVer 186-4 using PKCS 1.5 and 2048- and 3072-bit RSA keys: C1802. These claims are consistent with FCS_COP.1/SigGen in the [ST] section 5.3.2. ECDSA Signature Generation and Verification is covered by the following CAVP certificate which claims ECDSA SigGen 186-4 implementing NIST curves P-256 and P-384 as well as RSA SigVer 186-4 implementing NIST curves P-256 and P-384: C1802. These claims are 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

104 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_COP.1/Hash describes the following hash function to TSF associations:

IKE (ISAKMP): HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512

Software Image Verification: SHA-512

SSH: SHA-1, SHA-256, SHA-512 (all from SHS)

RADIUS Key Wrap: HMAC-SHA1

IPsec SA Authentication/Integrity: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512

2.2.6.2 Guidance Documentation

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

Findings: [AGD] Section 3.3.8 provides instructions on configuring the required hash sizes for IPsec. Example provided below.

```
Router#conf  t
Router(config)#crypto ikev2 proposal sample
Router(config-ikev2-proposal)#integrity sha1
```

2.2.6.3 Tests

106 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.

107 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

108 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

109 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

110 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

111 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

112 This test is for byte-oriented implementations only. The evaluators randomly generate a seed that is n bits long, where n is the length of the message digest produced by the hash function to be tested. The evaluators then formulate a set of 100 messages and associated digests by following the algorithm provided in Figure 1 of [SHAVS]. The evaluators then ensure that the correct result is produced when the messages are provided to the TSF.

Findings:	Hashing is covered by the following CAVP certificate all of which claim SHA1, SHA2-256, SHA2-384 and SHA2-512: C1802. These claims are consistent with FCS_COP.1/Hash in the [ST] section 5.3.2 FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm).

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

2.2.7.1 TSS

113 The evaluator shall examine the TSS to ensure that it specifies the following values used by the HMAC function: key length, hash function used, block size, and output MAC length used.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_COP.1/KeyedHash: “The TOE provides keyed-hashing message authentication services using HMAC-SHA-1 and HMAC-SHA-256 operates on 512-bit blocks and HMAC-SHA-512 operate on 1024-bit blocks of data, with key sizes and message digest sizes of 160-bits, 256 bits and 512 bits respectively) as specified in ISO 9797-2:2011, Section 7 “MAC Algorithm 2”.” Output MAC lengths are 160, 256, and 512 for HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-512, respectively.

2.2.7.2 Guidance Documentation

114 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:	[AGD] Section 3.3.1 describes how the administrator configures the TOE to use the values used by the HMAC function with SSH protocol. "The TOE also needs to be configured to only support hmac-sha1, hmac-sha256, and hmac-sha512 MAC algorithms using the following command [12] How to Configure SSH Algorithms for Common Criteria Certification -> Configuring a MAC Algorithm for a Cisco IOS SSH Server and Client: Ip ssh server algorithm mac hmac-sha1 hmac-sha256 hmac-sha512"

2.2.7.3 Tests

115 For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and message data using a known good implementation.

Findings:	HMAC is covered by the following CAVP certificate which claims HMAC-SHA1, HMAC-SHA2-256 and HMAC-SHA2-512: C1802. These claims are 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)

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

2.2.8.1 TSS

117 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_RBG_EXT.1: **"**The TOE implements a NIST-approved AES-CTR Deterministic Random Bit Generator (DRBG), as specified in SP 800-90 seeded by an entropy source that accumulates entropy from a TSF-platform based noise source. "The deterministic RBG is seeded with a minimum of 256 bits of entropy, which is at least equal to the greatest security strength of the keys and hashes that it will generate."

2.2.8.2 Guidance Documentation

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

Findings:	There are no additional instructions required to configure the RNG functionality. It is preconfigured and enabled by default.

2.2.8.3 Tests

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

120 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).

121 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.

122 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 following CAVP certificate claims use of a 256-bit key: C1802. These claims are consistent with FCS_RBG_EXT.1 in the [ST] section 5.3.2.

2.3 Identification and Authentication (FIA)

2.3.1 FIA_AFL.1 Authentication Failure Management

2.3.1.1 TSS

123 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_AFL.1: "The TOE provides the privileged administrator the ability to specify the maximum number of unsuccessful authentication attempts before privileged administrator or non-privileged administrator is locked out through the administrative CLI using a privileged CLI command. While the TOE supports a range from 1-25, in the evaluated configuration, the maximum number of failed attempts is recommended to be set to 3. "When a privileged administrator or non-privileged administrator attempting to log into the administrative CLI reaches the administratively set maximum number of failed authentication attempts, the user will not be granted access to the administrative functionality of the TOE until a privileged administrator resets the user's number of failed login attempts through the administrative CLI."

124 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_AFL.1: **"**Administrator lockouts are not applicable to the local console."

2.3.1.2 Guidance Documentation

125 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:	[AGD] Section 3.2.8 provides the command to configure the number of successive unsuccessful authentication attempts. "aaa local authentication attempts max-fail [number of failures]" The TOE only allows an authorized administrator to unlock the offending account, and the method for unlocking the account can be found in the same section of [AGD].

126 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:	[AGD] Section 3.2.8 User Lockout – advises that an additional Administrator account be created to only be used in an emergency to unlock the locked privileged administrator account if it gets locked out due to number of failed attempts. Although, it is noted that the lockout is not applicable to the local console administrators.

2.3.1.3 Tests

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

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

High-Level Test Description
Configure the account locking mechanism to 3 attempts. Lock out the user with 3 bad attempts and show that the 4th is denied even if the proper password is used. Using the administrator account, unlock the locked user and then show that the user can log in again.
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

128 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_PMG_EXT.1: “The passwords can be composed of any combination of upper and lower case letters, numbers, and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”. Minimum password length is settable by the Authorized Administrator, and can be configured for minimum password lengths of 15 characters.”

2.3.2.2 Guidance Documentation

129 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:	[AGD] Section 4.2 Passwords – describes the characters that may be used for passwords and provides guidance on the composition of strong passwords, as well as provides instructions on setting the minimum password length. "The password can be composed of any combination of characters that includes characters for at least 3 of these four character sets: upper case letters, lower case letters, numerals, and the following special characters: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”.” “You can also set the password minimum length in the aaa common-criteria policy using the min-length <length>” The evaluated configuration requires the administrator to set the minimum password length to 15 characters.

2.3.2.3 Tests

130 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 current admin user using the identified TSFI. Show that the password can be used to login to the SSH interface. Change the password for the current admin to a password which is less than the configured minimum and show it is rejected. Change the “enable” password using the identified TSFI. Show that the new password can be used to elevate access after it is changed. Change the password length to be 8 characters. As one admin, change the password for another admin to be only 7 characters and show it is rejected. Change the password for another admin to be 8 characters and show it is accepted. Change the “enable” password using the identified TSFI to a password that is only 7 characters and show it is rejected. Change the “enable” password” to be 8 characters and show it is accepted.
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 were performed in Test 1.

2.3.3 FIA_UIA_EXT.1 User Identification and Authentication

2.3.3.1 TSS

131 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_UIA_EXT.1: “Once a potential administrative user attempts to access the CLI of the TOE through either a directly connected console or remotely through an SSHv2 secured connection, the TOE prompts the user for a user name and password. Only after the administrative user presents the correct authentication credentials will access to the TOE administrative functionality be granted. “Local password-based authentication for administrative users accessing the TOE through SSHv2, and optionally supports deferring password-based authentication to a remote AAA server. In addition, public key algorithm for authentication is RSA Signature Verification, which is configured during the TOE installation. “The administrator authentication policies include authentication to the local user database or redirection to a remote authentication server. Interfaces can be configured to try one or more remote authentication servers, and then fail back to the local user database if the remote authentication servers are inaccessible.

	"The process for authentication is the same for administrative access whether administration is occurring via a directly connected console or remotely via SSHv2 secured connection. "At initial login, the administrative user is prompted to provide a username. After the user provides the username, the user is prompted to provide the administrative password associated with the user account. The TOE then either grant administrative access (if the combination of username and password is correct) or indicate that the login was unsuccessful."

132 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_UIA_EXT.1: "The TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed except for the login warning banner that is displayed prior to user authentication and any network packets as configured by the authorized administrator may flow through the switch."

133 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.

134 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

135 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 following login methods were identified by the evaluator: Console, and SSH. The Console and SSH login can use password-based authentication, [AGD] Section 4.2 provides guidance on configuring the TOE to enforce password complexity. The TOE supports local or remote password-based authentication. For remote password-based authentication [AGD] Section 3.3.8 provides instructions for configuring the

	IPsec tunnel between the TOE and remote AAA server used for authentication. [AGD] Section 3.3.6 also provides instructions for generating x509 certificates used in the IPsec tunnel. SSH can be configured to use RSA Public Keys and [AGD] Section 3.3.1 describes how to generate the RSA key as well as how to configure the TOE to use Public Keys to login.

2.3.3.3 Tests

136 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. Login into the identified management interface using a known-bad credential and logout. Ensure the appropriate audit messages appear. The tests were repeated for the local (Serial console) using local passwords and RADIUS authentication, as well as remote (SSH) using local passwords, RADIUS authentication and Public keys.
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. All claimed services available to remote entities are identified as part of AVA_VAN.1 test scanning.
Findings: PASS

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

High-Level Test Description
The device does not have any services configured prior to I&A. All claimed services available to remote entities are identified as part of AVA_VAN.1 test scanning.
Findings: PASS

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

Note:	The TOE is not a distributed TOE.

2.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

137 Evaluation Activities for this requirement are covered under those for FIA_UIA_EXT.1. If other authentication mechanisms are specified, the evaluator shall include those methods in the activities for FIA_UIA_EXT.1.

2.3.5 FIA_UAU.7 Protected Authentication Feedback

2.3.5.1 TSS

138 None

2.3.5.2 Guidance Documentation

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

Findings:	There are no preparatory steps required to ensure authentication data is not revealed while entering for each local login allowed.

2.3.5.3 Tests

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

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

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

2.4 Security management (FMT)

2.4.1 General Requirements for distributed TOEs

2.4.1.1 TSS

141 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.

Note:	This is covered in the FMT SFRs.

2.4.1.2 Guidance Documentation

142 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.

Note:	This is covered in the FMT SFRs.

2.4.1.3 Tests

143 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).

Note:	This is covered in the FMT SFRs.

2.4.2 FMT_MOF.1/ManualUpdate

2.4.2.1 TSS

144 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

145 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:	[AGD] Section 2 provides steps required to verify authenticity of the update image. Router# show software authenticity file <filename> [AGD] Section 4.8 provides the steps to perform manual updates to the TOE. "9. Specify new image to be loaded "#boot system flash: <system image file name from Step 5 above>"

146 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

147 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 TOE using an account with privileges which should not permit upgrades. Attempt to upgrade the device. The action should fail.
Findings: PASS

148 The evaluator shall try to perform the update with prior authentication as Security Administrator using a legitimate update image. This attempt should be successful. This test case should be covered by the tests for FPT_TUD_EXT.1 already.

Note	This 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

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_MTD.1/CoreData: “No administrative functionality is available prior to administrative login.”

150 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_MTD.1/CoreData: Access to the trust store (TOE data) is restricted through role-based access controls- “The TOE performs role-based authorization, using TOE platform authorization mechanisms, to grant access to the semi-privileged and privileged roles.”

	"Each of the predefined and administratively configured roles has create (set), query, modify, or delete access to the TOE data." "No administrative functionality is available prior to administrative login." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMF.1: "The management functionality of the TOE is provided through the TOE CLI. The specific management capabilities available from the TOE include: … • Ability to import X.509v3 certificates to the TOE's trust store; … • Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;"

2.4.3.2 Guidance Documentation

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

Findings:	The combination of the [AGD] and links to the online Cisco documentation provided in the AGD lists all the functions that can be used to manipulate TSF data. By default only privileged administrators have access to those functions, and the [AGD] Section 3.2.1 provides configuration information to ensure only administrators have access to the functions.

152 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:	[AGD] section 3.3.6 provides information on generating key pairs, creating certificate signing requests, authenticating the CA certificates and configuring a CA certificate trust anchor. The evaluator determined that the AGD provides sufficient information for the administrator to configure and maintain the trust store in a secure way.

2.4.3.3 Tests

153 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

154 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), 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)

155 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMF.1: "The management functionality of the TOE is provided through the TOE CLI." The FMT_SMF.1 section of Table 19 also lists the security management functions supported by the TOE and identified that they were accessible on all interfaces. The TOE CLI is available through the local console or SSH as described in other parts of the TSS. The evaluator examined the Guidance Documentation and the TOE as observed during all other testing and confirmed that the management functions specified in FMT_SMF.1 are provided by the TOE.

156 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] Section 6.1 states "The TOE supports [..] local administration via a directly connected console cable." "The management functionality of the TOE is provided through the TOE CLI." "Local and remote administration of the TOE and the services provided by the TOE via the TOE CLI, as described above;" [AGD] Section 1.5 states "This includes any IT Environment Console that is directly connected to the TOE via the Serial Console Port and is used by the TOE administrator to support TOE administration." The evaluator examined the TSS and Guidance Documentation and determined they both described the local administrative interface, and the Guidance Documentation included appropriate warnings to ensure the interface is local.

157 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

158 See section 2.4.4.1.

2.4.4.3 Tests

159 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.

2.4.5 FMT_SMR.2 Restrictions on security roles

2.4.5.1 TSS

160 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMR.1: "The TOE platform maintains privileged and semi-privileged administrator roles. The TOE performs role-based authorization, using TOE platform authorization mechanisms, to grant access to the semi-privileged and privileged roles. For the purposes of this evaluation, the privileged role is equivalent to full administrative access to the CLI, which is the default access for IOS privilege level 15; and the semi-privileged role equates to any privilege level that has a subset of the privileges assigned to level 15. Privilege levels 0 and 1 are defined by default and are customizable, while levels 2-14 are undefined by default and are also customizable. Note: the levels are not hierarchical."

2.4.5.2 Guidance Documentation

161 The evaluator shall review the guidance documentation to ensure that it contains instructions for administering the TOE both locally and remotely, including any configuration that needs to be performed on the client for remote administration.

Findings:	The TOE is able to be administered both locally from the console and remotely through SSH. The [AGD] Section 3.2.1 and 3.3.1 provide instructions on configuring the TOE for remote administration. [AGD] Section 3.2 states, "The console can be any environment console that is physically connected to the router, via the Serial Console Port (RJ-45)."

2.4.5.3 Tests

162 In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

Note:	There are no explicit test activities and therefore none are recorded here. 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

163 The evaluator shall examine the TSS to determine that it details how any pre-shared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through an interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_SKP_EXT.1: "The TOE stores all private keys in a secure directory protected from access as there is no interface in which the keys can be accessed. "The TOE includes CLI command features that can be used to configure the TOE to encrypt all locally defined user passwords. In this manner, the TOE ensures that plaintext user passwords will not be disclosed even to administrators." "Additionally, enabling the 'hidekeys' command in the logging configuration ensures that passwords are not displayed in plaintext. "The TOE includes a Master Passphrase feature that can be used to configure the TOE to encrypt all locally defined user passwords using AES."

2.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

2.5.2.1 TSS

164 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_APW_EXT.1: "The command service password-encryption applies encryption to all passwords, including username passwords, authentication key passwords, the privileged command password, console, and virtual terminal line access passwords." Passwords are encrypted using AES. There are no administrative interfaces available that allow passwords to be viewed in plaintext as they are encrypted via the password-encryption service.

2.5.3 FPT_TST_EXT.1 TSF testing

2.5.3.1 TSS

165 The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory

is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

Findings:

[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_TST_EXT.1:

"• AES Known Answer Test –

For the encrypt test, a known key is used to encrypt a known plain text value resulting in an encrypted value. This encrypted value is compared to a known encrypted value to ensure that the encrypt operation is working correctly. The decrypt test is just the opposite. In this test a known key is used to decrypt a known encrypted value. The resulting plaintext value is compared to a known plaintext value to ensure that the decrypt operation is working correctly.

• RSA Signature Known Answer Test (both signature/verification) –

This test takes a known plaintext value and Private/Public key pair and used the public key to encrypt the data. This value is compared to a known encrypted value to verify that encrypt operation is working properly. The encrypted data is then decrypted using the private key. This value is compared to the original plaintext value to ensure the decrypt operation is working properly.

• RNG/DRBG Known Answer Test –

For this test, known seed values are provided to the DRBG implementation. The DRBG uses these values to generate random bits. These random bits are compared to known random bits to ensure that the DRBG is operating correctly.

• HMAC Known Answer Test –

For each of the hash values listed, the HMAC implementation is fed known plaintext data and a known key. These values are used to generate a MAC. This MAC is compared to a known MAC to verify that the HMAC and hash operations are operating correctly.

• SHA-1/256/384/512 Known Answer Test –

For each of the values listed, the SHA implementation is fed known data. These values are used to generate a hash. This hash is compared to a known value to verify they match, and the hash operations are operating correctly.

• ECDSA self-test –

This test takes a known plaintext value and Private/Public key pair and used the public key to encrypt the data. This value is compared to a known encrypted value to verify that encrypt operation is working properly. The encrypted data is then decrypted using the private key. This value is compared to the original plaintext value to ensure the decrypt operation is working properly.

• Software Integrity Test –

The Software Integrity Test is run automatically whenever the IOS system image is loaded and confirms that the image file that's about to be loaded has maintained its integrity."

	"These tests are sufficient to verify that the correct version of the TOE software is running as well as that the cryptographic operations are all performing as expected because any deviation in the TSF behavior will be identified by the failure of a self-test."

166 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

167 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:	[AGD] Section 3.2.4 Administration of Cryptographic Self-Tests describes that when the self-tests fail the TOE will transition into an error state and all secure data transmission is halted and the TOE outputs status information indicating the failure. [AGD] Section 3.2.5 and Section 2 describes the Software Integrity Test that is performed on TOE startup or reload. If the image was tampered with in any way, an error would be displayed, and the image ill not boot. The [AGD] directs administrators to refer to the Cisco System Messages Overview for troubleshooting error messages and actions to take.

168 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

169 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.

170 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.

171 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.

172 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. Note: The TOE does not output the results of the cryptographic self-tests on boot. The evaluator witnessed a FIPS build with debug logging that confirmed the tests are run at boot time, and the results of a self-test failure.
Findings: PASS

2.5.4 FPT_TUD_EXT.1 Trusted Update

2.5.4.1 TSS

173 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_TUD_EXT.1: “The current active version can be verified by executing the “show version” command from the TOE’s CLI. When software updates are made available by Cisco, an administrator can obtain, verify the integrity of, and install those updates.”

174 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_TUD_EXT.1: “An Authorized Administrator can query the software version running on the TOE and can initiate updates to software images.” “When software updates are made available by Cisco, an administrator can obtain, verify the integrity of, and install those updates. The updates can be downloaded from the software.cisco.com.” “Digital signatures and published hash mechanisms are used to verify software/firmware update files (to ensure they have not been modified from the originals distributed by Cisco) before they are used to actually update the applicable

	TOE components. The TOE image files are digitally signed so their integrity can be verified during the boot process, and an image that fails an integrity check will not be loaded. "To verify the digital signature prior to installation, the "show software authenticity file" command allows you to display software authentication related information that includes image credential information, key type used for verification, signing information, and other attributes in the signature envelope, for a specific image file. If the output from the "show software authenticity file" command does not provide the expected output, contact Cisco Technical Assistance Center (TAC) https://tools.cisco.com/ServiceRequestTool/create/launch.do."

175 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:	The TOE does not support such functionality.

176 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.

177 If a published hash is used to protect the trusted update mechanism, then the evaluator shall verify that the trusted update mechanism does involve an active authorization step of the Security Administrator, and that download of the published hash value, hash comparison and update is not a fully automated process involving no active authorization by the Security Administrator. In particular, authentication as Security Administration according to FMT_MOF.1/ManualUpdate needs to be part of the update process when using published hashes.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_TUD_EXT.1: "The cryptographic hashes (i.e., SHA-512) are used to verify software update files (to ensure they have not been modified from the originals distributed by Cisco) before they are used to actually update the applicable TOE components. Authorized Administrators can download the approved image file from Cisco.com onto a trusted computer system for usage in the trusted update functionality. The hash value can be displayed by hovering over the software image name under details on the Cisco.com web site. The verification should not be performed on the TOE during the update process. If the hashes do not match, contact Cisco Technical Assistance Center (TAC)."

2.5.4.2 Guidance Documentation

178 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:	[AGD] Section 2 Secure Acceptance of the TOE – Step 11 provides the method to query the version as “show version”. Step 9 provides the method for confirming the verify the digital signature of the loaded software prior to installation.

179 The evaluator shall verify that the guidance documentation describes how the verification of the authenticity of the update is performed (digital signature verification or verification of published hash). The description shall include the procedures for successful and unsuccessful verification. The description shall correspond to the description in the TSS.

Findings:	The TOE uses both a published hash and a digital signature to verify the authenticity of the update. [AGD] Section 2 Secure Acceptance of the TOE – Step 9 describes how the administrator should verify the published hash of the downloaded firmware prior to installation. If the hashes do not match, the administrator should contact the Cisco Technical Assistance Center (TAC). Prior to installation the administrator can verify the digital signature to confirm authenticity of the image. If the software authenticity command does not provide expected output, the administrator is advised to contact the Cisco TAC. The TOE will automatically display the hash verification on boot or by using the reload command. If the image was tampered with in any way, an error would be displayed, and the image will not boot. The ROM monitor will confirm the digital signature for the image used for booting. If there was an issue with the digital signature verification, an error message would be displayed, and the image will not boot.

180 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:	AGD] Section 2 Secure Acceptance of the TOE – Step 8 describes that the Security Administrator must use the hash identified in the Common Criteria Operational User Guidance And Preparative Procedures document which can be downloaded from the Cisco website.

181 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.

182 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.

183 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:	TOE does not use a certificate-based mechanism for software update digital signature verification.

2.5.4.3 Tests

184 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.

High-Level Test Description
Attempt to install a bad image, an unsigned image and a badly signed image for both downgrades and upgrades. After each attempt, get the current version of the TOE using all available means and ensure they are consistent.
Findings: PASS

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

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

2) The evaluator uses a legitimate update and tries to perform verification of the hash value without providing the published hash value to the TOE. The evaluator confirms that this attempt fails. Depending on the implementation of the TOE it might not be possible to attempt the verification of the hash value without providing a hash value to the TOE, e.g. if the hash value needs to be handed over to the TOE as a parameter in a command line message and the syntax

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

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

185 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.

Findings:	The TOE supports published hashes but the verification is not claimed to be done by the TOE.

186 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.

187 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

[Modified by TD 0632]

188 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.

189 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:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_STM_EXT.1: "The TOE provides a source of date and time information used in audit event timestamps, and for certificate validity checking. The clock function is reliant on the system clock provided by the underlying hardware. This date and time is used as the time stamp that is applied to TOE generated audit records and used to track inactivity of administrative sessions. The time information is also used in various routing protocols such as, OSPF, BGP, and ERF; Set system time, Calculate IKE stats (including limiting SAs based on times); determining AAA timeout, and administrative session timeout."

2.5.5.2 Guidance Documentation

[Modified by TD 0632]

190 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.

191 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:	In the evaluated configuration the TOE does not support the use of an NTP server. [AGD] Section 4.3 System Clock Management provides instructions on how to set the time.

2.5.5.3 Tests

[Modified by TD 0632]

192 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, 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.

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

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

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

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

Findings:	The TOE is not a vND and does not obtain time from an underlying VS.

193 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

194 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_SSL_EXT.1: “An administrator can configure maximum inactivity times individually for both local and remote administrative sessions through the use of the “session-timeout” setting applied to the console. When a session is inactive (i.e., no session input from the administrator) for the configured period of time the TOE will terminate the session,

	and no further activity is allowed requiring the administrator to log in (be successfully identified and authenticated) again to establish a new session." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_SSL.3: "The allowable inactivity timeout range is from 1 to 65535 seconds."

2.6.1.2 Guidance Documentation

195 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:	[AGD] Section 4.6 Use of Administrative Session Termination – states local administrative session locking is supported, and the administrator is required to re-authenticate after the session has become locked. The evaluator confirmed the [AGD] provided instructions for configuring the inactivity time period.

2.6.1.3 Tests

196 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 10, 12 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

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_SSL_EXT.1: "An administrator can configure maximum inactivity times individually for both local and remote administrative sessions through the use of the "session-timeout" setting applied to the console. When a session is inactive (i.e., no session input from the administrator) for the configured period of time the TOE will terminate the session, and no further activity is allowed requiring the administrator to log in (be successfully identified and authenticated) again to establish a new session. If a remote user session is inactive for a configured period of time, the session will be terminated and will require authentication to establish a new session. [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_SSL.3 "The allowable inactivity timeout range is from 1 to 65535 seconds."

2.6.2.2 Guidance Documentation

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

Findings:	[AGD] Section 3.2.7 Session Termination – includes instructions for configuring the inactivity time period for remote administrative session termination. "line vty <first> <last> exec-timeout <time> "where first and last are the range of vty lines on the box (i.e. "0 4"), and time is the period of inactivity after which the session should be terminated for remote administration access via SSH."

2.6.2.3 Tests

199 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 10, 12 minutes: Change the idle timeout to this value;

High-Level Test Description
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. Note that because the system uses a single command to control all idle timers, we will set in one interface and check in another.
Findings: PASS

2.6.3 FTA_SSL.4 User-initiated Termination

2.6.3.1 TSS

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_SSL.4: "An administrator is able to exit out of both local and remote administrative sessions. Each administrator logged onto the TOE can manually terminate their session using the "exit" or "logout" command."

2.6.3.2 Guidance Documentation

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

Findings:	[AGD] Section 3.2.7 Session Termination – states how to terminate a local or remote interactive session. "In addition to session timeouts, an administrator can manually logout from the TOE with the following command: exit"

2.6.3.3 Tests

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

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

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

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

High-Level Test Description
Log into the SSH CLI interface. Log out using the TSFI previous discussed.
Findings: PASS

2.6.4 FTA_TAB.1 Default TOE Access Banners

2.6.4.1 TSS

203 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTA_TAB.1: "The TOE displays a privileged Administrator specified banner on the CLI management interface prior to allowing any administrative access to the TOE. This interface is applicable for both local (via console) and remote (via SSH) TOE administration."

2.6.4.2 Guidance Documentation

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

Findings:	[AGD] Section 4.5 Administrative Banner Configuration – describes how to configure the banner message. "…to create a banner of text "This is a banner" use the command banner login d This is a banner d"

2.6.4.3 Tests

205 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

206 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTP_ITC.1: “The TOE also requires that peers and other TOE instances establish an IKE/IPSec connection in order to forward routing tables used by the TOE. The TOE also requires that peers establish an IKE/IPsec connection to a CA server for sending certificate signing requests. “The TOE protects communications between the TOE and the remote audit server using IPsec. This provides a secure channel to transmit the log events. Communications between the TOE and AAA servers are secured using IPsec. “The distinction between “remote VPN peer” and “another instance of the TOE” is that “another instance of the TOE” would be installed in the evaluated configuration, and likely administered by the same personnel, whereas a “remote VPN peer” could be any interoperable IPsec gateway/peer that is expected to be administered by personnel who are not administrators of the TOE, and who share necessary IPsec tunnel configuration and authentication credentials with the TOE administrators. For example, the exchange of X.509 certificates for certificate based authentication.” As is quoted above, the method of assured identification of non-TSF endpoints is accomplished using X.509 certificates.

	The evaluator confirmed all secure communication mechanisms are outlined in sufficient detail in the TSS and correspond to the claimed cryptographic protocol SFRs in the [ST].

2.7.1.2 Guidance Documentation

207 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:	IPsec is the only allow protocol to secure trusted channels between TOE and authorized IT entities. [AGD] Section 3.3.8 Configuration of IPsec – contains instructions for establishing IPsec with all authorized IT entities. "If an IPsec session with a peer is unexpectedly interrupted, the connection will be broken. The IPsec session will be re-established (a new SA set up) once the peer is back online."

2.7.1.3 Tests

208 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.

209 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 audit server, RADIUS server and a trusted CA using IPSec, which are set up as per the evaluated configuration. They are constantly tested throughout the evaluation. Note that because they all operate over IPSec which is implemented independent of the trusted channel application layer, we can assert that a single IPSec connection meets the requirements.

b. Test 2: For each protocol that the TOE can initiate as defined in the requirement, the evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be initiated from the TOE.

Note:	This test is performed in FCS_IPSEC_EXT.1 (various). The TOE can operate as an initiator in the IPSec peering relationship.

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

Note:	See FCS_IPSEC_EXT.1 (various). Data is sent using Encapsulating Security Payload (ESP).

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

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

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

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

High-Level Test Description
Engage wireshark over the logging interface.
Log into the TOE using the serial interface to avoid being disconnected.
Log into the TOE over the SSH CLI using a predefined RADIUS user and a good password. Invoke the trusted CA interface by querying an CRL Distribution Point. Verify that log messages have appeared on the logging server.
Physically disconnect the TOE from the environment and immediately reconnect. Show that the protected communications have not been affected.
Physically disconnect the TOE from the environment and wait for 30 minutes. During this time, at the serial console, attempt to log in using a predefined RADIUS user and good password. Invoke the trusted CA interface by querying a CRL Distribution Point. Monitor the local audit log events to detect that the protected communications have been identified as offline.
Physically reconnect the TOE back to the network.
Log into the TOE over the SSH CLI using a predefined RADIUS user and a good password. Invoke the trusted CA interface by querying a CRL Distribution Point. Verify that log messages have appeared on the logging server.
Examine wireshark and verify that all protected interfaces have automatically re-established encrypted communications without any user intervention.
Findings: PASS

Further assurance activities are associated with the specific protocols.

210 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:	The TOE is not a distributed TOE.

211 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

212 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FTP_TRP.1: "All remote administrative communications take place over a secure encrypted SSHv2 session. The SSHv2 session is encrypted using AES encryption. The remote users are able to initiate SSHv2 communications with the TOE." The remote administration protocol described in the TSS (SSHv2) is consistent with the protocol described in the requirement, and is included in the requirements in the ST.

2.7.2.2 Guidance Documentation

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

Findings:	The TOE supports SSH for remote administrative sessions. [AGD] Section 3.3.1 contains instructions on configuring the TOE for establishing SSHv2 remote administrative sessions.

2.7.2.3 Tests

214 The evaluator shall perform the following tests:

a. Test 1: The evaluators shall ensure that communications using each specified (in the guidance documentation) remote administration method is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

Note:	The only trusted path is the SSH CLI, which is set up as per the evaluated configuration. The SSH CLI is constantly tested throughout the evaluation.

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

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

High-Level Test Description
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

215 Further assurance activities are associated with the specific protocols.

216 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

217 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_IPSEC_EXT.1 IPsec Protocol

4.1.1.1 TSS

FCS_IPSEC_EXT.1.1

218 The evaluator shall examine the TSS and determine that it describes what takes place when a packet is processed by the TOE, e.g., the algorithm used to process the packet. The TSS describes how the SPD is implemented and the rules for processing both inbound and outbound packets in terms of the IPsec policy. The TSS describes the rules that are available and the resulting actions available after matching a rule. The TSS describes how those rules and actions form the SPD in terms of the BYPASS (e.g., no encryption), DISCARD (e.g., drop the packet), and PROTECT (e.g., encrypt the packet) actions defined in RFC 4301.

219 As noted in section 4.4.1 of RFC 4301, the processing of entries in the SPD is non-trivial and the evaluator shall determine that the description in the TSS is sufficient to determine which rules will be applied given the rule structure implemented by the TOE. For example, if the TOE allows specification of ranges, conditional rules, etc., the evaluator shall determine that the description of rule processing (for both inbound and outbound packets) is sufficient to determine the action that will be applied, especially in the case where two different rules may apply. This description shall cover both the initial packets (that is, no SA is established on the interface or for that particular packet) as well as packets that are part of an established SA.

Findings: [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1:

"A crypto map (the Security Policy Definition (SPD)) set can contain multiple entries, each with a different access list. The crypto map entries are searched in a sequence - the router attempts to match the packet to the access list (acl) specified in that entry. Separate access lists define blocking and permitting at the interface). For example:

Router# access-list 101 permit ip 192.168.3.0 0.0.0.255 10.3.2.0 0.0.0.255

"When a packet matches a permit entry in a particular access list, the method of security in the corresponding crypto map is applied. If the crypto map entry is tagged as ipsec-isakmp, IPsec is triggered. For example:

Router# crypto map MAP_NAME 10 ipsec-isakmp

"The match address 101 command means to use access list 101 in order to determine which traffic is relevant. For example:

Router# (config-crypto-map)#match address 101

"The traffic matching the permit acls would then flow through the IPSec tunnel and be classified as "PROTECTED".

"Traffic that does not match a permit acl and is also blocked by other non-crypto acls on the interface would be DISCARDED.

	"Traffic that does not match a permit acl in the crypto map, but that is not disallowed by other acls on the interface is allowed to BYPASS the tunnel."

FCS_IPSEC_EXT.1.3

220 The evaluator checks the TSS to ensure it states that the VPN can be established to operate in transport mode and/or tunnel mode (as identified in FCS_IPSEC_EXT.1.3).

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "In addition to tunnel mode, which is the default IPsec mode, the TOE also supports transport mode, allowing for only the payload of the packet to be encrypted."

FCS_IPSEC_EXT.1.4

221 The evaluator shall examine the TSS to verify that the selected algorithms are implemented. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1/KeyedHash Cryptographic Operations (for keyed-hash message authentication) and if the SHA-based HMAC function truncated output is utilized it must also be described.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The IPsec protocol ESP is implemented using the cryptographic algorithms AES-GCM-128, AES-GCM-192, AES-GCM-256, AES-CBC-128, AES-CBC-192 and AES-CBC-256 together with HMAC-SHA-1, HMAC-SHA-256, and HMAC-SHA-512." The IPsec ESP algorithms described in the TSS are consistent with those selected in the FCS_IPSEC_EXT.1.4 and FCS_COP.1/KeyedHash elements of the [ST]. The TOE does not support truncated SHA-based HMAC.

FCS_IPSEC_EXT.1.5

222 The evaluator shall examine the TSS to verify that IKEv1 and/or IKEv2 are implemented.

223 For IKEv1 implementations, the evaluator shall examine the TSS to ensure that, in the description of the IPsec protocol, it states that aggressive mode is not used for IKEv1 Phase 1 exchanges, and that only main mode is used. It may be that this is a configurable option.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The TOE supports IKEv2 session establishment." The TOE does not claim support for IKEv1.

FCS_IPSEC_EXT.1.6

224 The evaluator shall ensure the TSS identifies the algorithms used for encrypting the IKEv1 and/or IKEv2 payload, and that the algorithms chosen in the selection of the requirement are included in the TSS discussion.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1:

	"The TOE provides AES-CBC-128, AES-CBC-192, AES-CBC-256, AES-GCM-128 and AES-GCM-256 for encrypting IKEv2 payloads." The IPsec IKEv2 payload encryption algorithms described in the TSS are consistent with those selected in the FCS_IPSEC_EXT.1.6 element of the [ST].

FCS_IPSEC_EXT.1.7

225 The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 1 SA lifetime and/or the IKEv2 SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The TOE supports configuration lifetimes of both Phase 1 SAs and Phase 2 SAs using the following command, "lifetime". The time values for Phase 1 SAs can be limited up to 24 hours and for Phase 2 SAs up to 8 hours." The description given in the TSS is consistent with the selections made in the FCS_IPSEC_EXT.1.5 and FCS_IPSEC_EXT.1.7 elements of the [ST].

FCS_IPSEC_EXT.1.8

226 The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 2 SA lifetime and/or the IKEv2 Child SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The TOE supports configuration lifetimes of both Phase 1 SAs and Phase 2 SAs using the following command, "lifetime"." "The Phase 2 SA lifetimes can also be configured by an Administrator based on number of packets." "The TOE supports configuring the maximum amount of traffic that is allowed to flow for a given IPsec SA using the following command, 'crypto ipsec security-association lifetime'. The default amount is 2560KB, which is the minimum configurable value. The maximum configurable value is 4GB." The description given in the TSS is consistent with the selections made in the FCS_IPSEC_EXT.1.5 and FCS_IPSEC_EXT.1.8 elements of the [ST].

FCS_IPSEC_EXT.1.9

227 The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating "x". The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of "x" meets the stipulations in the requirement.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The TOE supports Diffie-Hellman Group 14 (2048-bit keys), 19 (256-bit Random ECP), and 20 (384-bit Random ECP) in support of IKE Key Establishment."

	"The secret value 'x' used in the IKE Diffie-Hellman key exchange ("x" in g^x mod p) is generated using a NIST-approved AES-CTR Deterministic Random Bit Generator (DRBG) and shall have possible lengths from 224 – 384-bits (112 – 192-bits security strength)."

FCS_IPSEC_EXT.1.10

228 If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The nonces used in IKE exchanges are generated using a NIST-approved AES-CTR DRBG and in a manner such that the probability that a specific nonce value will be repeated during the life a specific IPsec SA is less than 1 in 2^[128]. The nonce is generated according to the security strength associated with the negotiated DH group, is at least 128 bits in size, and is at least half the output size of the negotiated pseudorandom function (PRF) hash. The PRF is 256- 512-bits in length." The evaluator verified that the random number generator meets the required length of the nonce as specified in NIST SP 800-90 and "Table 2:Comparable Strengths" in the NIST SP 800-57 Pt. 1 Rev4.

229 If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The nonces used in IKE exchanges are generated using a NIST-approved AES-CTR DRBG and in a manner such that the probability that a specific nonce value will be repeated during the life a specific IPsec SA is less than 1 in 2^[128]. The nonce is generated according to the security strength associated with the negotiated DH group, is at least 128 bits in size, and is at least half the output size of the negotiated pseudorandom function (PRF) hash. The PRF is 256- 512-bits in length." Table 20 of this section provides a mapping of security strength and output length to supported DH groups. The evaluator verified that the random number generator meets the required length of the nonce as specified in NIST SP 800-90 and "Table 2:Comparable Strengths" in the NIST SP 800-57 Pt. 1 Rev4.

FCS_IPSEC_EXT.1.11

230 The evaluator shall check to ensure that the DH groups specified in the requirement are listed as being supported in the TSS. If there is more than one DH group supported, the evaluator checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1:

	"The TOE supports Diffie-Hellman Group 14, 19, and 20. To set the DH group, use the 'group [x]' command replacing the group number for each of the supported groups, for example 'group 14'." The description given in the TSS is consistent with the selection made in the FCS_IPSEC_EXT.1.11 element of the [ST].

FCS_IPSEC_EXT.1.12

231 The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges. The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "IPsec Internet Key Exchange, also called ISAKMP, is the negotiation protocol that lets two peers agree on how to build an IPsec Security Association (SA). The strength of the symmetric algorithm negotiated to protect the IKEv2 IKE_SA connection is greater than or equal to the strength of the symmetric algorithm negotiated to protect the IKEv2 CHILD_SA connection. The Security Administrator must configure the IKEv2 Transform Sets to ensure that the symmetric algorithm used in the connection is great or equal to the symmetric algorithm use to protect the IKEv2 CHILD_SA connection. The strength of the IKEv2 IKE_SA connection is checked during tunnel negotiation." "The IPsec protocol ESP is implemented using the cryptographic algorithms AES-GCM-128, AES-GCM-192, AES-GCM-256, AES-CBC-128, AES-CBC-192 and AES-CBC-256…." "The TOE provides AES-CBC-128, AES-CBC-192, AES-CBC-256, AES-GCM-128 and AES-GCM-256 for encrypting IKEv2 payloads."

FCS_IPSEC_EXT.1.13

232 The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1/SigGen Cryptographic Operations (for cryptographic signature).

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "The IKE protocols implement Peer Authentication using RSA and ECDSA along with X.509v3 certificates, or pre-shared keys." The description given in the TSS is consistent with the algorithms specified in the FCS_COP.1/SigGen component of the [ST].

233 If pre-shared keys are chosen in the selection, the evaluator shall check to ensure that the TSS describes how pre-shared keys are established and used in authentication of IPsec connections. The description in the TSS shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_PSK_EXT.1:

	"Through the implementation of the CLI, the TOE supports use of IKEv2 pre-shared keys for authentication of IPsec tunnels. Preshared keys can be entered as ASCII character strings, or HEX values. The TOE supports keys that are from 22 characters in length up to 127 bytes in length. The data that is input is conditioned by the cryptographic module prior to use via SHA-1." As per the selection made in the FIA_PSK_EXT.1.4 element of the [ST], the TOE does not generate PSKs.

FCS_IPSEC_EXT.1.14

234 The evaluator shall ensure that the TSS describes how the TOE compares the peer's presented identifier to the reference identifier. This description shall include which field(s) of the certificate are used as the presented identifier (DN, Common Name, or SAN). If the TOE simultaneously supports the same identifier type in the CN and SAN, the TSS shall describe how the TOE prioritizes the comparisons (e.g. the result of comparison if CN matches but SAN does not). If the location (e.g. CN or SAN) of non-DN identifier types must explicitly be configured as part of the reference identifier, the TSS shall state this. If the ST author assigned an additional identifier type, the TSS description shall also include a description of that type and the method by which that type is compared to the peer's presented certificate, including what field(s) are compared and which fields take precedence in the comparison.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "When certificates are used for authentication, the distinguished name (DN) is verified to ensure the certificate is valid and is from a valid entity. The DN naming attributes in the certificate is compared with the expected DN naming attributes and deemed valid if the attribute types are the same and the values are the same and as expected. The fully qualified domain name (FQDN) can also be used as verification where the attributes in the certificate are compared with the expected CN: FQDN, CN: user FQDN and CN: IP Address." "Certificate maps provide the ability for a certificate to be matched with a given set of criteria. You can specify which fields within a certificate should be checked and which values those fields may or may not have. There are six logical tests for comparing the field with the value: equal, not equal, contains, does not contain, less than, and greater than or equal. ISAKMP and ikev2 profiles can bind themselves to certificate maps, and the TOE will determine if they are valid during IKE authentication."

4.1.1.2 Guidance Documentation

FCS_IPSEC_EXT.1.1

235 The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to construct entries into the SPD that specify a rule for processing a packet. The description includes all three cases – a rule that ensures packets are encrypted/decrypted, dropped, and flow through the TOE without being encrypted. The evaluator shall determine that the description in the guidance documentation is consistent with the description in the TSS, and that the level of detail in the guidance documentation is sufficient to allow the administrator to set up the SPD in an unambiguous fashion. This includes a discussion of how ordering of rules impacts the processing of an IP packet.

Findings:	The TOE configures the SPD information flow control and VPN capabilities using the access control functionality. Access Lists rule definition for allowing traffic to flow over the VPN, be blocked, and bypass are defined in [AGD] Section 3.3.5 Information Flow Policies. Traffic matching is done based on a top-down approach in the access list. The first entry that a packet matches will be the one applied to it.

FCS_IPSEC_EXT.1.3

236 The evaluator shall confirm that the guidance documentation contains instructions on how to configure the connection in each mode selected.

Findings:	Tunnel mode is the default mode for all IKE connections in the TOE. If transport mode is desired, [AGD] Section 3.3.8.5 contains instructions on how to configure the by using the "mode transport" setting. To revert back to the tunnel mode, the option "mode tunnel" can be used.

FCS_IPSEC_EXT.1.4

237 The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Findings:	[AGD] Section 3.3.8.3 provides instructions on how to configure the TOE to use the algorithms selected. To configure IPsec ESP to use HMAC-SHA-1 and AES-CBC-128 use the following command: crypto ipsec transform-set example esp-aes 128 esp-sha-hmac "NOTE: In the evaluated configuration esp-aes also supports 192- and 256-bit key sizes and AES-GCM 128 and 256. HMAC-SHA 256 and HMAC-SHA 512 are supported." The algorithms are consistent with those selected in the ST.

FCS_IPSEC_EXT.1.5

238 The evaluator shall check the guidance documentation to ensure it instructs the administrator how to configure the TOE to use IKEv1 and/or IKEv2 (as selected), and how to configure the TOE to perform NAT traversal (if selected).

Findings:	Instructions for configuring IKEv2 can be found in [AGD] 3.3.8.2 IKEv2 Transform sets. IKEv1 is not claimed in the evaluated configuration. NAT traversal is configured in the global configuration mode with the command: "crypto ipsec nat-transparency spi-matching" as per [AGD] Section 3.3.8.4.

239 If the IKEv1 Phase 1 mode requires configuration of the TOE prior to its operation, the evaluator shall check the guidance documentation to ensure that instructions for this configuration are contained within that guidance.

Findings:	IKEv1 is not claimed in the evaluated configuration.

FCS_IPSEC_EXT.1.6

240 The evaluator ensures that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Findings:	[AGD] Section 3.3.8.2 describes the configuration of all selected algorithms for IKEv2. This is consistent with the selections in FCS_IPSEC_EXT.1.6 "Router(config-ikev2-proposal)#encryption aes-cbc-128 Note: AES key sizes of 192 and 256 are allowed in the evaluated configuration. AES-GCM mode with key sizes of 128 and 256 are allowed in the evaluated configuration. The authorized administrator may configure the TOE to use aes-cbc or aes-gcm."

FCS_IPSEC_EXT.1.7

[Modified by TD0633]

241 The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 24 hours is exceeded (e.g. configure a time value of 23h 45min to ensure the actual rekey is performed no later than 24h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 1 SA value of 24 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 24 hours. It is not permitted to configure a value of 24 hours if that leads to an actual rekey after more than 24hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Findings:	[AGD] Section 3.3.8.2 IKEv2 Transform Sets - describes the default time value for Phase 1 SAs is 24 hours (86400 seconds), but this setting can be changed using the command above with different values. The SA lifetimes are configured with the command “lifetime <time in secs>”. The TOE is not claiming Phase 1 lifetime based on number of bytes.

FCS_IPSEC_EXT.1.8

[Modified by TD0633]

242 The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 8 hours is exceeded (e.g. configure a time value of 7h 45min to ensure the actual rekey is performed no later than 8h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 2 SA value of 8 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 8 hours. It is not permitted to configure a value of 8 hours if that leads to an actual rekey after more than 8hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Findings:	[AGD] Section 3.3.8.3 IPsec Transform and Lifetimes – provides instructions to configure IKEv2 Child SA lifetimes based on time and based on number of bytes: “crypto ipsec security-association lifetime seconds <num_seconds>” and “crypto ipsec security-association lifetime kilobytes <num_kilobytes>”

FCS_IPSEC_EXT.1.11

243 The evaluator ensures that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Findings:	[AGD] Section 3.3.8.2 describe configuring the key exchange algorithms in the transform sets with the “group 14 or <dh_group>” command.

FCS_IPSEC_EXT.1.13

244 The evaluator ensures the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

Findings:	Using RSA and ECDSA x509 certificates is described in [AGD] Section 3.3.6.10 as, Once X.509v3 keys are installed on the TOE, they can be set for use with IKEv2 with the commands: TOE-common-criteria(config-ikev2-profile)#authentication [remote | local] rsa-sig -or- TOE-common-criteria(config-ikev2-profile)#authentication [remote | local] ecdsa-sig

245 The evaluator shall check that the guidance documentation describes how pre-shared keys are to be generated and established. The description in the guidance documentation shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Findings:	[AGD] Section 3.3.8.2 IKEv2 Transform sets – describes how pre-shared keys are to be generated and established. **“**Router (config-ikev2-keyring)#peer peer1 Router (config-ikev2-keyring-peer)#address 0.0.0.0 0.0.0.0 Router (config-ikev2-keyring-peer)#pre-shared-key cisco123!cisco123!CISC “This section creates a keyring to hold the pre-shared keys referenced in the steps above. In IKEv2 these pre-shared keys are specific to the peer. “Pre-shared keys on the TOE must be at least 22 characters in length and can be composed of any combination of upper-case letters (A-Z), and lower case letters (a-z), numbers (0-9), and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”). “The TOE supports pre-shared keys up to 127 bytes in length. While longer keys increase the difficulty of brute-force attacks, longer keys increase processing time.” As per [ST] FIA_PSK_EXT.1.4 and [AGD] Section 3.3.8.2 the TOE does not generate bit-based pre-shared keys in the evaluated configuration.

246 The evaluator will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA, and ensure a valid certificate for that CA is loaded into the TOE and marked “trusted”.

Findings:	In the evaluated configuration the TOE connects to the trusted CA through an IPsec tunnel using a pre-shared key as detailed in [AGD] Section 3.3.6.3. Per [AGD] Section 3.3.6.2, to declare the trustpoint that the TOE should use, use the crypto pki trustpoint name command in configuration mode “crypto pki trustpoint <name>” Where the <name> creates the name of the trustpoint (crypto pki trustpoint ciscotest) Once the trustpoint has been declared, the administrator must authenticate the CA, by getting the certificate of the CA, using the command: “crypto ca authenticate <trustpoint-name>”

FCS_IPSEC_EXT.1.14

247 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE does not guarantee unique identifiers, the evaluator shall ensure that the operational

guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings:	[AGD] / 3.3.8.1 Configure Reference Identifier - Table 9 Reference Identifier Configuration describes the following supported identifiers which are consistent with the ST. “field-name—Specifies one of the following case-insensitive name strings (CN: Fully Qualified Domain Name (FQDN), CN: user FQDN, CN: IP Address. Distinguished Name (DN))” [AGD] / 3.3.8.1 explicitly states that the TOE does not support the SAN extension and includes instructions on how to configure the reference identifier.

4.1.1.3 Tests

FCS_IPSEC_EXT.1.1

248 The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

a. Test 1: The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The selectors used in the construction of the rule shall be different such that the evaluator can generate a packet and send packets to the gateway with the appropriate fields (fields that are used by the rule - e.g., the IP addresses, TCP/UDP ports) in the packet header. The evaluator performs both positive and negative test cases for each type of rule (e.g. a packet that matches the rule and another that does not match the rule). The evaluator observes via the audit trail, and packet captures that the TOE exhibited the expected behaviour: appropriate packets were dropped, allowed to flow without modification, encrypted by the IPsec implementation.

High-Level Test Description
Create three rules in the IPv4 firewall table which will permit traffic to bypass the VPN or enter (and pass) the VPN or enter (and be blocked) at the VPN. Generate traffic to show positive and negative tests that meet the rules.
Findings: PASS

b. Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. As with Test 1, the evaluator ensures both positive and negative test cases are constructed. These scenarios must exercise the range of possibilities for SPD entries and processing modes as outlined in the TSS and guidance documentation. Potential areas to cover include rules with overlapping ranges and conflicting entries, inbound and outbound packets, and packets that establish SAs as well as packets that belong to established SAs. The evaluator shall verify, via the audit trail and packet captures, for each scenario that the expected behavior is exhibited, and is consistent with both the TSS and the guidance documentation.

Note:	The TOE’s SPD is configured using access list rules tested throughout FPF_RUL_EXT.1 from the MOD_VPNGW v1.1 which the TOE also claims conformance. Testing throughout FPF_RUL_EXT.1.5 and FPF_RUL_EXT.1.6 of the MOD_VPNGW v1.1 exercises the range of possibilities for SPD entries including overlapping ranges, conflicting entries, inbound and outbound traffic. Packets that establish SAs as well as packets that belong to SAs are tested throughout

	FCS_IPSEC_EXT.1 as access lists rules are required to establish the trusted channel.

FCS_IPSEC_EXT.1.2

249 The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

250 The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

251 The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The evaluator may use the SPD that was created for verification of FCS_IPSEC_EXT.1.1. The evaluator shall construct a network packet that matches the rule to allow the packet to flow in plaintext and send that packet. The evaluator should observe that the network packet is passed to the proper destination interface with no modification. The evaluator shall then modify a field in the packet header; such that it no longer matches the evaluator-created entries (there may be a "TOE created" final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was dropped.

Note:	This test is performed as part of FCS_IPSEC_EXT.1.1 in which a plaintext packet was successfully transmitted through the TOE. The evaluator confirmed that packets that do not match the evaluator-created entries are dropped as part of the FPF_RUL_EXT testing activities of which this TOE also claims conformance.

FCS_IPSEC_EXT.1.3

252 The evaluator shall perform the following test(s) based on the selections chosen:

a. Test 1: If tunnel mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in tunnel mode and also configures a VPN peer to operate in tunnel mode. The evaluator configures the TOE and the VPN peer to use any of the allowable cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator shall then initiate a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

High-Level Test Description
Configure the outside workstation VPN and TOE to operate the VPN in tunnel mode. Initiate a connection and show that the VPN is established.
Findings: PASS

b. Test 2: If transport mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in transport mode and also configures a VPN peer to operate in transport mode. The evaluator configures

the TOE and the VPN peer to use any of the allowed cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

High-Level Test Description
Configure the outside workstation VPN and TOE to operate the VPN in transport mode. Initiate a connection and show that the VPN is established.
Findings: PASS

FCS_IPSEC_EXT.1.4

253 The evaluator shall configure the TOE as indicated in the guidance documentation configuring the TOE to use each of the supported algorithms, attempt to establish a connection using ESP, and verify that the attempt succeeds.

High-Level Test Description
Configure the TOE to negotiate phase 2 with the given encryption and integrity algorithm. Send traffic through the VPN and verify that the packets are encapsulated.
Findings: PASS

FCS_IPSEC_EXT.1.5

254 Tests are performed in conjunction with the other IPsec evaluation activities.

a. Test 1: If IKEv1 is selected, the evaluator shall configure the TOE as indicated in the guidance documentation, and attempt to establish a connection using an IKEv1 Phase 1 connection in aggressive mode. This attempt should fail. The evaluator should then show that main mode exchanges are supported.

Note:	The TOE does not claim IKEv1.

b. Test 2: If NAT traversal is selected within the IKEv2 selection, the evaluator shall configure the TOE so that it will perform NAT traversal processing as described in the TSS and RFC 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

High-Level Test Description
After configuring the peer to operate behind a NAT, initiate an IPSec connection from the peer to the TOE and show that the IPSec connection is established and that it traverses the NAT successfully.
Findings: PASS

FCS_IPSEC_EXT.1.6

255 The evaluator shall configure the TOE to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is

configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

High-Level Test Description
For IKEv2: • Configure the TOE to negotiate phase 1 with the claimed ciphersuites. • Configure the peer to only accept the payload using the indicated ciphersuite. • Send traffic through the VPN and verify that the packets are encapsulated.
Findings: PASS

FCS_IPSEC_EXT.1.7

256 When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

257 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a. Test 1: If 'number of bytes' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.

Note:	The TOE does not claim volume-based rekeying for phase 1 SAs.

b. [Modified by TD0633] Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 24 hours for the Phase 1 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 1 SA lifetime that exceeds the Phase 1 SA lifetime on the TOE.

High-Level Test Description
Configure the TOE to rekey after 24 hours has elapsed to show it can be done. Then configure the TOE to rekey after 255 seconds have elapsed for phase 1 and show that it actually rekeys after the given amount of time.
Findings: PASS

FCS_IPSEC_EXT.1.8

258 When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC "A difference between IKEv1 and IKEv2 is

that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered."

259 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a. Test 1: If 'number of bytes' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish an SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

High-Level Test Description
Configure the TOE to only accept 1 GB of traffic before a new phase 2 SA is negotiated to confirm the TOE is able to accept the maximum bytes of 1GB. Configure the peer to only accept 5 MB of traffic before a new phase 2 SA is negotiated. Send traffic from the peer to the TOE and verify the phase 2 SA is negotiated after the configured limit is reached.
Findings: PASS

b. [Modified by TD0633] Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 8 hours for the Phase 2 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 2 SA lifetime that exceeds the Phase 2 SA lifetime on the TOE.

High-Level Test Description
Configure the TOE to rekey phase 2 after 8 hours has elapsed to show that the TOE can accept the configured time period. Then configure the TOE to rekey after 255 seconds have elapsed for phase 2 and show that it actually rekeys after the given amount of time.
Findings: PASS

FCS_IPSEC_EXT.1.10

260 Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a. Test 1: If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	This is a TSS requirement and no testing is performed. Please see TSS verdict for FCS_IPSEC_EXT.1.10 paragraph 228.

b. Test 2: If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	This is a TSS requirement and no testing is performed. Please see TSS verdict for FCS_IPSEC_EXT.1.10 paragraph 229.

FCS_IPSEC_EXT.1.11

261 For each supported DH group, the evaluator shall test to ensure that all supported IKE protocols can be successfully completed using that particular DH group.

High-Level Test Description
Configure the TOE and VPN peer to us each of the claimed DH groups and attempt to establish an IPsec connection. Confirm that the TOE can successfully establish a connection using each supported DH group.
Findings: PASS

FCS_IPSEC_EXT.1.12

262 The evaluator simply follows the guidance to configure the TOE to perform the following tests.

a. Test 1: This test shall be performed for each version of IKE supported. The evaluator shall successfully negotiate an IPsec connection using each of the supported algorithms and hash functions identified in the requirements.

Note:	This test was conducted in full for FCS_IPSEC_EXT.1.4.

b. Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP that selects an encryption algorithm with more strength than that being used for the IKE SA (i.e., symmetric algorithm with a key size larger than that being used for the IKE SA). Such attempts should fail.

High-Level Test Description
The VPN peer is configured to present a SA for ESP that selects an encryption algorithm with more strength than that is being used for the IKE SA. The TOE is configured to only establish a SA for ESP with an encryption algorithm that is equal or less than that being used for the IKE SA. Note: Per the guidance documentation “Note: The authorized administrator must ensure that the keysize for this setting [IKEv2 transform set] is greater than or equal to the keysize selected for ESP in Section 4.6.2 below. If AES 128 is

High-Level Test Description
selected here, then the highest keysize that can be selected on the TOE for ESP is AES 128 (either CBC or GCM)." Attempt to establish an IPsec connection between the TOE and the VPN peer and confirm that it fails to establish.
Findings: PASS

c. Test 3: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an IKE SA using an algorithm that is not one of the supported algorithms and hash functions identified in the requirements. Such an attempt should fail.

High-Level Test Description
Configure the VPN tunnel on the TOE to be able to select amongst all supported ciphersuites for phase 1. Configure the peer to use an unsupported ciphersuite for phase 1 and show the connection is not established.
Findings: PASS

d. Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an SA for ESP (assumes the proper parameters where used to establish the IKE SA) that selects an encryption algorithm that is not identified in FCS_IPSEC_EXT.1.4. Such an attempt should fail.

High-Level Test Description
Configure the IKE SA to be a valid ciphersuite. Configure the TOE to allow all ESP SA ciphersuites. Attempt to connect to the TOE with a ciphersuite that is not allowed.
Findings: PASS

FCS_IPSEC_EXT.1.13

263 For efficiency sake, the testing that is performed may be combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

FCS_IPSEC_EXT.1.14

264 For each the context of the tests below, a valid certificate is a certificate that passes FIA_X509_EXT.1 validation checks but does not necessarily contain an authorized subject.

265 The evaluator shall perform the following tests:

266 Test 1: [conditional] For each CN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes CN checking over SAN (through explicit configuration of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the SAN so it contains an

incorrect identifier of the correct type (e.g. the reference identifier on the TOE is example.com, the CN=example.com, and the SAN:FQDN=otherdomain.com) and verify that IKE authentication succeeds.

High-Level Test Description
Configure the TOE to only allow the peer reference identifier that matches the claimed CN/identifier types. Generate peer certificates containing each of the CN/identifier types and attempt to establish a connection with the TOE. Confirm that the IKE authentication succeeds.
Findings: PASS

267 Test 2: [conditional] For each SAN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes SAN checking over CN (through explicit specification of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the CN so it contains an incorrect identifier formatted to be the same type (e.g. the reference identifier on the TOE is DNS-ID; identify certificate has an identifier in SAN with correct DNS-ID, CN with incorrect DNS-ID (and not a different type of identifier)) and verify that IKE authentication succeeds.

Note:	There are no SAN/identifier types selected in the evaluated configuration.

268 Test 3: [conditional] For each CN/identifier type combination selected, the evaluator shall:

e) Create a valid certificate with the CN so it contains the valid identifier followed by '\0'. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the evaluator shall configure the SAN so it matches the reference identifier.

High-Level Test Description
Using the valid certificates from FCS_IPSEC_EXT.1.14 – Test 1 generate certificates with a null characters for all claimed CN/identifier types. Confirm that the certificates are valid.
Findings: PASS

f) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the CN without the '\0' and verify that IKE authentication fails.

High-Level Test Description
Attempt to establish a connection with the VPN peer that is using the certificates with null terminated identifiers created in Test 3a and ensure the authentication fails.
Findings: PASS

269 Test 4: [conditional] For each SAN/identifier type combination selected, the evaluator shall:

a. Create a valid certificate with an incorrect identifier in the SAN. The evaluator shall configure a string representation of the correct identifier in the DN. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the addition/modification shall be to any non-CN field of the DN. Otherwise, the addition/modification shall be to the CN.

Note:	There are no SAN/identifier types selected in the evaluated configuration.

b. Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the correct identifier (expected in the SAN) and verify that IKE authentication fails.

Note:	There are no SAN/identifier types selected in the evaluated configuration.

270 Test 5: [conditional] If the TOE supports DN identifier types, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the subject DN in the peer's presented certificate and shall verify that the IKE authentication succeeds.

High-Level Test Description
Configure the TOE to use the subject DN as the peer's reference identifier and ensure that the IKE authentication succeeds.
Findings: PASS

271 Test 6: [conditional] If the TOE supports DN identifier types, to demonstrate a bit-wise comparison of the DN, the evaluator shall create the following valid certificates and verify that the IKE authentication fails when each certificate is presented to the TOE:

a. Duplicate the CN field, so the otherwise authorized DN contains two identical CNs.

High-Level Test Description
Generate a VPN Peer certificate with a duplicate CN field. Using the configuration from FCS_IPSEC_EXT.1.14 – Test 5, attempt to establish a VPN connection using the certificate with a duplicate CN field for the VPN Peer and verify that the authentication fails.
Findings: PASS

b. Append '\0' to a non-CN field of an otherwise authorized DN.

High-Level Test Description
Generate a VPN Peer certificate with a NULL character in a non-CN field (Locality field). Using the configuration from FCS_IPSEC_EXT.1.14 – Test 5, attempt to establish a VPN connection using the certificate with a NULL character appended to the non-CN field for the VPN Peer and verify that the authentication fails.

High-Level Test Description
Findings: PASS

4.1.2 FCS_SSHS_EXT.1 SSH Server

4.1.2.1 TSS

FCS_SSHS_EXT.1.2

[Modified by TD0631]

272 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).

273 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.

274 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "Public key algorithm for authentication is RSA Signature Verification (ssh-rsa) which is configured during the TOE installation." This is consistent with signature verification algorithms selected in FCS_COP.1/SigGen. "Local password-based authentication for administrative users accessing the TOE through SSHv2, and optionally supports deferring password-based authentication to a remote AAA server." The description given in the TSS is consistent with signature verification algorithms selected in FCS_COP.1/SigGen of the [ST]. Additionally, password-based methods described in the TSS correspond to the selection made in FCS_SSHS_EXT.1.

FCS_SSHS_EXT.1.3

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

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "Packets greater than 65,535 bytes in an SSH transport connection are dropped. Large packets are detected by the SSH implementation, and dropped internal to the SSH process."

FCS_SSHS_EXT.1.4

276 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "Encryption algorithms, AES-CBC-128, AES-CBC-256 to ensure confidentiality of the session." The evaluator confirmed these encryption algorithms are identical to those listed for this component and that the implementation and optional characteristics of the protocol are specified.

FCS_SSHS_EXT.1.5

[Modified by TD0631]

277 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "The TOE supports ssh-rsa host public key algorithms." This is consistent with those listed in FCS_SSHS_EXT.1.5.

FCS_SSHS_EXT.1.6

278 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "The TOE's implementation of SSHv2 supports hashing algorithms hmac-sha1, hmac-sha256, and hmac-sha512 to ensure the integrity of the session." The description of supported data integrity algorithms for SSH in the TSS is consistent with the selections made in the FCS_SSHS_EXT.1.6 element of the [ST].

FCS_SSHS_EXT.1.7

279 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "The TOE's implementation of SSHv2 can be configured to only allow Diffie-Hellman Group 14 (2048-bit keys) Key Establishment." The description of supported key exchange algorithms for SSH in the TSS is consistent with the selections made in the FCS_SSHS_EXT.1.7 element of the [ST].

FCS_SSHS_EXT.1.8

280 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_SSHS_EXT.1: "The TOE can also be configured to ensure that SSH re-key of no longer than one hour and no more than one gigabyte of transmitted data for the session key. Note, the TOE will react to first threshold limit reached and perform an ip-ssh rekey."

4.1.2.2 Guidance Documentation

FCS_SSHS_EXT.1.4

281 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:	[AGD] Section 3.3.1 Remote Administration Protocols – provides guidance on configuring the TOE only accept AES128-CBC and AES256-CBC, with the following command "ip ssh server algorithm encryption aes128-cbc aes256cbc".

FCS_SSHS_EXT.1.5

282 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:	Per [AGD] Section 3.3.1, the TOE accepts three types of user authentication methods: Public-key, keyboard-interactive, and password. In the evaluated configuration keyboard-interactive is disabled. To disable keyboard-interactive authentication use the command: "no ip ssh server authenticate user keyboard".

FCS_SSHS_EXT.1.6

283 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:	Per [AGD] Section 3.3.1, to configure the TOE to only support hmac-sha1 and hmac-sha1-96 the administrator must use the command: "ip ssh server algorithm mac hmac-sha1 hmac-sha256 hmac-sha512".

FCS_SSHS_EXT.1.7

284 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:	Per [AGD] Section 3.3.1, to configure the key exchange algorithm to Diffie-hellman-group14-sha1, the administrator must use the command: "ip ssh dh min size 2048".

FCS_SSHS_EXT.1.8

285 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:	Per [AGD] Section 3.3.1, the TOE configures the thresholds of no longer than one hour (time 60) and no more than one gigabyte (volume 1000000) with the following command: “ip ssh rekey time 60” and “ip ssh rekey volume 1000000”. “Note, the TOE will react to first threshold limit reached and perform an ip-ssh rekey.”

4.1.2.3 Tests

FCS_SSHS_EXT.1.2

[Modified by TD0631]

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

287 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

288 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
Load a supported public key into the TOE. Off-TOE, use a different private key (generated with the same public key algorithm) to try to connect. The connection attempt should fail.
Findings: PASS

289 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.

290 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

291 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

292 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 a 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]

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

294 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
In the evaluated configuration the TOE only uses ssh-rsa for host public key algorithms. Using an SSH client attempt to connect using the allowed host key algorithm and confirm that the client is able to authenticate the TOE server public key using the claimed algorithm.
Findings: PASS

295 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.

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

High-Level Test Description
Using an SSH client configured to only allow ssh-dss host key algorithms, attempt to establish an SSH session with the TOE. The SSH connection fails.
Findings: PASS

FCS_SSHS_EXT.1.6

297 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.

298 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

299 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.

300 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

301 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

302 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

303 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.

304 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).

305 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

306 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).

307 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).

308 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
Copy a file more than the 1 GB volume limit to the TOE and use SCP (Secure copy) that uses the SSH protocol to transfer the file. The TOE will rekey after the 1 GB limit is reached.
Findings: PASS

309 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).

High-Level Test Description
Log into the TOE as a privileged administrator and modify the configurable time- or volume-based rekeying thresholds. Verify the new limit takes effect by rerunning the time- or volume-based test case above with the new limit. Log into the TOE as an unprivileged administrator and attempt to modify the configurable time- or volume- based rekeying thresholds. Verify that the attempt fails.
Findings: PASS

310 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.

4.2 Identification and Authentication (FIA)

4.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

4.2.1.1 TSS

311 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_X509_EXT.1/Rev:

"CRL is used for certificate revocation checking. The authorized administrator could use the "revocation-check" command to specify at least one method of revocation checking; CRL is not the default method and must be selected in the evaluated configuration. The authorized administer sets the trust point and its name and the revocation-check method. If the TOE does not have the applicable CRL and is unable to obtain one, the TOE will reject the peer's certificate."

"Checking is also done for the basicConstraints extension and the CA flag to determine whether they are present and set to TRUE. The local certificate that was imported must contain the basic constraints extension with the CA flag set to true, the check also ensure that the key usage extension is present, and the keyEncipherment bit or the keyAgreement bit or both are set. If they are not, the certificate is not accepted.

"The certificate chain path validation is configured on the TOE by first setting crypto pki trustpoint name and then configuring the level to which a certificate chain is processed on all certificates including subordinate CA certificates using the chain-validation command. If the connection to determine the certificate validity cannot be established, the certificate is not accepted and the connection will not be established.

"Note, certificate revocation check is performed when certificates are loaded on the device and on each use during the authentication step and is the same process for all certificates."

312 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_X509_EXT.1/Rev:

"Note, certificate revocation check is performed when certificates are loaded on the device and on each use during the authentication step and is the same process for all certificates."

4.2.1.2 Guidance Documentation

313 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:	[AGD] Section 3.3.6 describes that the check of validity takes place when certificates are loaded on the device and on each use during an authentication step. The revocation process is the same for all certificates. There are no rules described for extendedKeyUsage fields in FIA_X509_EXT.1.1 that are not supported by the TOE.

4.2.1.3 Tests

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

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

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

High-Level Test Description
Create a sequence of three X.509 certificates: a root CA, an intermediate CA signed by the root CA and a leaf node certificate signed by the intermediate CA. Load the root CA and intermediate CA into the TOE trust store. Configure the VPN peer to use the leaf certificate and attempt to establish an IPsec connection. The connection succeeds. Remove the root CA from the TOE trust store. Force the TOE to connect to a TLS server show that the connection is no longer accepted.
Findings: PASS

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

High-Level Test Description
Create an X.509 leaf certificate with a 'notAfter' date in the past. Attempt to connect to a VPN Peer that sends back this certificate and show it is not accepted. As the intermediate CA certificate is loaded in the trust store, the certificate must be valid at load time. Create an intermediate CA certificate that will expire in 10 mins. Load the intermediate CA certificate into the TOE's trust store. Wait until intermediate CA certificate has expired and attempt to establish an IPsec connection, the connection fails.
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 CRL has no revoked certificates. Verify that a certificate results in a successful connection. Then revoke the server certificate. Verify the connection now fails due to the certificate being revoked. Then unrevoke the certificate from the CRL. Revoke the intermediate CA and restart the root CA CRL server. Verify the connection now fails due to the certificate being revoked. 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
Generate a duplicate intermediate CA without the cRLSigning bit. Ensure the CRL has no revoked certificates.

High-Level Test Description
Verify that a certificate results in a successful connection. Then revoke the server certificate. Verify the connection now fails due to the CRL being signed by the invalid intermediate CA.
Findings: PASS

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

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back a properly mangled X.509 certificate in which the ASN.1 header bytes in the first 8 bytes are modified. The Lightship strongswan tool has been configured to modify the first byte in the peer certificate when generating the AUTH packet as strongswan needs to verify the packet before allowing it to be used in an IPsec tunnel.
Findings: PASS

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

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

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

High-Level Test Description
Force the TOE to connect to a Lightship test server which will send back an X.509 certificate in which the public key of the certificate is modified.
Findings: PASS

[Added by TD0527]

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

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

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.

Note:	In the evaluated configuration the TOE requires that the subordinate CAs be included in the trust store (trustpoints). With the intermediate CAs loaded in the trust store the TOE does not process CA certificates presented in certificate message. This test is considered not applicable.

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.

Note:	In the evaluated configuration the TOE requires that the subordinate CAs be included in the trust store (trustpoints). With the intermediate CAs loaded in the trust store the TOE does not process CA certificates presented in certificate message. This test is considered not applicable.

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 two ECDSA certificates: Root CA and Intermediate CA with named curve parameters.
Import the ECDSA Root CA into the trust store.
Attempt to import the Intermediate CA with the named curve parameters into the trust store and observe that it is successful.
Create a clone of the Intermediate CA, such that the public key is explicitly defined rather than being a named curve.
Attempt to import the Intermediate CA with the explicit curve into the trust store and observe that it is rejected.

High-Level Test Description
Findings: PASS

316 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.

317 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).

318 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

319 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

320 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_IPSEC_EXT.1: "Certificate maps provide the ability for a certificate to be matched with a given set of criteria. You can specify which fields within a certificate should be checked and which values those fields may or may not have. There are six logical tests for comparing the field with the value: equal, not equal, contains, does not contain, less than, and greater than or equal. ISAKMP and ikev2 profiles can bind themselves to certificate maps, and the TOE will determine if they are valid during IKE authentication." Review of AGD-specific requirements will be done in the AGD review phase which includes verification that the AGD contains all necessary instructions for configuring the operating environment so that the TOE can use the certificates.

321 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_X509_EXT.2: "If the connection to determine the certificate validity cannot be established, the certificate is not accepted and the connection will not be established."

4.2.2.2 Guidance Documentation

322 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:	[AGD] Section 3.3.6 X.509 Certificates and subsections include configuration instructions for generating RSA and ECDSA Key pairs, importing and authenticating Certificate Authorities to the trust store. [AGD] Section 3.3.6 X.509 Certificates describes that the administrator must use PEM X.509v3 certificates in the operating environment so the TOE can use the certificates. For the TOE to use the certificates the [AGD] states, “The certificate chain path validation is configured on the TOE by first setting crypto pki trustpoint name and then configuring the level to which a certificate chain is processed on all certificates, including subordinate CA certificates using the chain-validation command. If the connection to determine the certificate validity cannot be established, the certificate is not accepted, and the connection will not be established.” [AGD] Section 3.3.6.10 Setting X.509 for use with IKE provides instructions on setting the type of certificate that will be used to authenticate both the TOE and the IPsec peer.

4.2.2.3 Tests

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

324 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 CRL HTTP server disabled, ensure that the TOE attempts to obtain the CRL to validate the certificates. Show that when the CRL server 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

325 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:	This selection is not made in 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:	In the [AGD] Section 3.3.6.2, there are instructions for creating a Certificate Signing Request (CSR). This section instructs the administrator to configure the x509 fields using the following command: “subject-name CN=routerTOE.cisco.com,O=cisco,OU=TAC,C=U” To send CSR requests to the certification authority (CA) the administrator must specify the enrolment parameters of the CA with the following command: “enrollment url http://ip_address”.

4.2.3.3 Tests

326 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 imported into the TOE. The certificate fails to be imported. Authenticate the trustpoint by adding the signing CA certificate to the TOE, then attempt to import the signed certificate. The certificate is successfully imported.
Findings: PASS

4.3 Security management (FMT)

4.3.1 FMT_MOF.1/Functions Management of security functions behaviour

4.3.1.1 TSS

327 For distributed TOEs see chapter 2.4.1.1.

Findings:	The TOE is not a distributed TOE.

328 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_GEN.1: "The administrator can set the level of the audit records to be displayed on the console or sent to the syslog server." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_STG.1: "The TOE is configured to export syslog records to a specified, external syslog server in real-time. The TOE protects communications with an external syslog server via IPsec. If the IPsec connection fails, the TOE will store audit records on the TOE when it discovers it can no longer communicate with its configured syslog server. When the connection is restored, the TOE will transmit the buffer contents when connected to the syslog server."

4.3.1.2 Guidance Documentation

329 For distributed TOEs see chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

330 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:	[AGD] Section 3.4.1 Managing Audit Records - provides details on configuring the audit log severity level. "Configuring the audit log severity level is done with the logging buffered command. Router(config)# logging buffered <0-7> Severity levels: 1 – Alerts 2 – Critical 3 – Errors 4 – Warnings 5 – Notifications 6 – Informational 7 – Debugging" [AGD] Section 3.4.3 Remote Logging - provides instructions for transmitting audit data to a remote IT entity. "logging host <ip address of syslog server>"

4.3.1.3 Tests

331 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
The TOE uses an IPsec tunnel to transmit audit data to an external IT entity.
As a non-privileged admin, attempt to modify the syslog ipsec tunnel and show it is unsuccessful.
As a privileged admin, attempt to modify the syslog ipsec tunnel.
As a non-privileged admin, attempt to stop the logging service and show it is unsuccessful.
As a privileged admin, attempt to stop the logging service and show it is successful.
Findings: PASS

332 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.

333 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 IPsec tunnel are shown in the test findings for FCS_IPSEC_EXT.1.

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

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

335 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.

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

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

337 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.

338 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.

339 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.

340 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.

341 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_MOF.1/Services Management of security functions behaviour

4.3.2.1 TSS

342 For distributed TOEs see chapter 2.4.1.1.

Findings:	The TOE is not a distributed TOE.

343 For non-distributed TOEs, the evaluator shall ensure the TSS lists the services the Security Administrator is able to start and stop and how that how that operation is performed.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_MOF.1/Services: “The TOE provides the ability for Security Administrators to access TOE data, such as audit data, configuration data, security attributes, routing tables, and session thresholds and to perform manual updates to the TOE. The Security Administrator has the ability to start and stop services.” [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMF.1: “The management functionality of the TOE is provided through the TOE CLI.”

4.3.2.2 Guidance Documentation

344 For distributed TOEs see chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

345 For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the TSS lists the services the Security Administrator is able to start and stop and how that how that operation is performed.

Findings:	The entire [AGD] describes all the services the Security Administrator is able to start and stop including accessing TOE data, such as audit data, configuration data, security attributes, routing tables. Throughout the [AGD] the Security Administrator starts the services by explicitly enabling the command, unless it's enabled by default, and adding a "no" to the beginning of any command to disable the service. For instance, configuring the remote logging host uses the command "logging host <ip_address>" To disable the service the Security Administrator would repeat the command by adding a "no". "no logging host <ip_address>" In the TOE's default configuration an SNMP server and HTTP(s) server are enabled. In [AGD] Section 3.3.1 Remote Administration Protocols – the Security Administrator is instructed to disable the SNMP server and HTTP(s) server services with the following commands: # no snmp-server # no ip http server # no ip http secure-server

4.3.2.3 Tests

346 The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) without prior authentication as Security Administrator (either by authenticating as a user with no administrator privileges, if possible, or without prior authentication at all). The attempt to enable/disable this service/these services should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to enable/disable this service/these services can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

347 The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) with prior authentication as Security Administrator. The attempt to enable/disable this service/these services should be successful.

High-Level Test Description
As a non-privileged admin, attempt to start the logging service and show it is unsuccessful.
As a privileged admin, attempt to start the logging service and show it is successful.
As a non-privileged admin, attempt to stop the logging service and show it is unsuccessful.
As a privileged admin, attempt to stop the logging service and show it is successful.
Findings: PASS

4.3.3 FMT_MTD.1/CryptoKeys Management of TSF Data

4.3.3.1 TSS

348 For distributed TOEs see chapter 2.4.1.1.

Findings:	The TOE is not a distributed TOE.

349 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] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_MTD.1/CryptoKeys: "The Security Administrator enters the crypto key generate command in the CLI to generate RSA and ECDSA key pairs." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMF.1: "The management functionality of the TOE is provided through the TOE CLI. The specific management capabilities available from the TOE include: … • Ability to manage the cryptographic keys;"

4.3.3.2 Guidance Documentation

350 For distributed TOEs see chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

351 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:	[AGD] Section 3.3.1 Remote Administration Protocols – Provides instructions on generating an RSA key pair for the TOE's SSH server. "crypto key generate rsa" "Only one set of keys can be configured using the crypto key generate command at a time. Repeating the command overwrites the old keys." [AGD] Section 3.3.6.1 Generate a Key Pair – provides instructions on generating RSA ECDSA key pairs to be used for the TOE's local X.509 Certificates used in the trusted channel IPsec connections. [AGD] Section 3.3.6 provides instructions on importing and deleting signed X.509 certificates. [AGD] Section 3.3.8.5 provides instructions on using Pre-shared keys in IPsec connection.

4.3.3.3 Tests

352 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.

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

High-Level Test Description
As an unprivileged user, attempt to create a new private key and associate it with a trustpoint. The attempt should fail. The privileged case it performed as part of FIA_X509_EXT.3. As an unprivileged user, attempt to create a new private key and associate it with the SSH host. The attempt should fail. Then, as a privileged user, assign a new key to the SSH host. The attempt should be successful.
Findings: PASS

5 Evaluation Activities for PP-Module Virtual Private Network (VPN) Gateways

5.1 Security Audit (FAU)

5.1.1 FAU_GEN.1 Audit data generation

5.1.1.1 TSS

354 The evaluator shall verify that the TSS describes how the TSF can be configured to log network traffic associated with applicable rules. Note that this activity may be addressed in conjunction with the TSS Evaluation Activities for FPF_RUL_EXT.1.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "By implementing rules that defines the permitted flow of traffic between interfaces of the TOE for unauthenticated traffic. These rules control whether a packet is transferred from one interface to another…" "When network packets are received on a TOE interface, the TOE verifies whether the network traffic is allowed or not and performs one of the following actions, pass/not pass information, as well as optional logging."

355 The evaluator shall verify that the TSS describes how the TOE behaves when one of its interfaces is overwhelmed by network traffic. It is acceptable for the TOE to drop packets that it cannot process, but under no circumstances is the TOE allowed to pass packets that do not satisfy a rule that allows the permit operation or belong to an allowed established session. It may not always be possible for the TOE to audit dropped packets due to implementation limitations. These limitations and circumstances in which the event of dropped packets is not audited shall be described in the TSS.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_GEN.1: "When the incoming traffic to the TOE exceeds what the interface can handle, the packets are dropped at the input queue itself, and for each interface, the TOE indicates the number of dropped packets." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "The TOE is capable of inspecting network packet header fields to determine if a packet is part of an established session or not. ACL rules still apply to packets that are part of an ongoing session. "Packets will be dropped unless a specific rule has been set up to allow the packet to pass (where the attributes of the packet match the attributes in the rule and the action associated with the rule is to pass traffic). This is the default action that occurs on an interface if no ACL rule is found. If a packet arrives that does not meet any rule, it is expected to be dropped."

356 The evaluator also verifies that the TSS describes the auditable events for IPsec peer session establishment that are required by the PP-Module.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_GEN.1: “The types of events that cause audit records to be generated include: startup and shutdown of the audit mechanism cryptography related events to include IPSec session establishment with peer, identification and authentication related events, and administrative events (the specific events and the contents of each audit record are listed in the table within the FAU_GEN.1 SFR, “Auditable Events Table”). ”

5.1.1.2 Guidance Documentation

357 The evaluator shall verify that the operational guidance describes how to configure the TSF to result in applicable network traffic logging. Note that this activity may be addressed in conjunction with the guidance Evaluation Activities for FPF_RUL_EXT.1.

Findings:	Addressed by guidance activities for FPF_RUL_EXT.1

5.1.1.3 Tests

358 The following test is expected to execute outside the context of the other requirements. While testing the TOE’s compliance against the SFRs, either specific tests are developed and run in the context of this SFR, or as is typically done, the audit capability is turned on while testing the TOE’s behavior in complying with the other SFRs in the Base-PP and the PP-Module.

359 Test 1: The evaluator shall attempt to flood the TOE with network packets such that the TOE will be unable to process all the packets. This may require the evaluator to configure the TOE to limit the bandwidth the TOE is capable to handling (e.g., use of a 10 MB interface). The evaluator shall then review the audit logs to verify that the TOE correctly records that it is unable to process all of the received packets and verify that the TOE logging behavior is consistent with the TSS.

High-Level Test Description
Configure the TOE’s interface to limit the bandwidth on the incoming interface. Send a constant stream of packets to overwhelm the interface such that the TOE will be unable to process all packets.
Findings: PASS

360 Test 2: The evaluator shall use a remote VPN client to establish an IPsec session with the TOE and observe that the event is logged in accordance with the expectations of the PP-Module.

Note:	The TOE did not claim any optional SFRs for VPN client such as FTA_TSE.1, FTA_SSL.3/VPN, FTA_VCM_EXT.1. All audit events for FCS_IPSEC_EXT and FIA_X509_EXT were tested as part of NDcPP v2.2e of which this TOE claims conformance.

5.2 Cryptographic Key Management (FCS_CKM)

5.2.1 FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication)

5.2.1.1 TSS

361 The evaluator shall check to ensure that the TSS describes how the key-pairs are generated. In order to show that the TSF implementation complies with FIPS PUB 186-4, the evaluator shall ensure that the TSS contains the following information:

362 • The TSS shall list all sections of Appendix B to which the TOE complies.

363 • For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE;

364 • For each applicable section of Appendix B, any omission of functionality related to "shall" or “should” statements shall be described;

365 Any TOE-specific extensions, processing that is not included in the Appendices, or alternative implementations allowed by the Appendices that may impact the security requirements the TOE is to enforce shall be described.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FCS_CKM.1/IKE: “Asymmetric cryptographic keys used for IKE peer authentication are generated according to FIPS PUB 186-4, Appendix B.3 for RSA schemes and Appendix B.4 for ECDSA schemes.”

5.2.1.2 Guidance Documentation

366 The evaluator shall check that the operational guidance describes how the key generation functionality is invoked, and describes the inputs and outputs associated with the process for each signature scheme supported. The evaluator shall also check that guidance is provided regarding the format and location of the output of the key generation process.

Findings:	[AGD] Section 3.3.6.1 describes the following, “RSA and ECDSA keys are generated in pairs, one public key and one private key: (config)# crypto key generate rsa modulus 2048 -or- (config)# crypto key generate ec keysize <256 | 384> exportable The keys generated by this command are saved in the private configuration in NVRAM (which is never displayed to the user or backed up to another device) the next time the configuration is written to NVRAM. Note: Only one set of keys can be configured using the crypto key generate command at a time. Repeating the command overwrites the old keys. Note: If the configuration is not saved to NVRAM with a “copy run start”, the generated keys are lost on the next reload of the router.

5.2.1.3 Tests

***For FFC Schemes using "safe-prime" groups*:**

367 Testing for FFC Schemes using safe-prime groups is done as part of testing in FCS_CKM.2.

For all other selections:

368 The evaluator shall perform the corresponding tests for FCS_CKM.1 specified in the NDcPP SD, based on the selections chosen for this SFR. If IKE key generation is implemented by a different algorithm than the NDcPP key generation function, the evaluator shall ensure this testing is performed using the correct implementation.

Findings:	RSA key generation is covered by the following CAVP certificate all of which claim RSA KeyGen 186-4 for 2048-bit and 3072-bit RSA keys: C1802. These claims are consistent with FCS_CKM.1 in the [ST] section 5.3.2. ECDSA key generation is covered by the following CAVP certificate all of which claim ECDSA KeyGen 186-4 for NIST curves P-256 and P-384: C1802. These claims are consistent with FCS_CKM.1 in the [ST] section 5.3.2. FFC EC key generation is covered by the following CAVP certificate all of which claim KAS ECC Component for NIST curves P-256 and P-384: C1802. These claims are consistent with FCS_CKM.1 and FCS_CKM.2 in the [ST] section 5.2.2. Diffie-Hellman group 14 is covered by protocol testing in FCS_IPSEC_EXT.1.11 and FCS_SSHS_EXT.1.7 Test 2.

5.3 Identification and Authentication (FIA)

5.3.1 FIA_PSK_EXT.1 Pre-Shared Key Composition

5.3.1.1 TSS

369 The evaluator shall examine the TSS to ensure that it identifies all protocols that allow both text-based and bit-based pre-shared keys, and states that text-based pre-shared keys of 22 characters are supported. For each protocol identified by the requirement, the evaluator shall confirm that the TSS states the conditioning that takes place to transform the text-based pre- shared key from the key sequence entered by the user (e.g., ASCII representation) to the bit string used by the protocol, and that this conditioning is consistent with the last selection in the FIA_PSK_EXT.1.3 requirement.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FIA_PSK_EXT.1: "Through the implementation of the CLI, the TOE supports use of IKEv2 pre-shared keys for authentication of IPsec tunnels. Preshared keys can be entered as ASCII character strings, or HEX values. The TOE supports keys that are from 22 characters in length up to 127 bytes in length. The data that is input is conditioned by the cryptographic module prior to use via SHA-1." The evaluator confirmed that the stated input data conditioning is consistent with the last selection made in the FIA_PSK_EXT.1.3 element of section 5.3.3.9 of the [ST].

5.3.1.2 Guidance Documentation

370 The evaluator shall examine the operational guidance to determine that it provides guidance to administrators on the composition of strong text-based pre-shared keys,

and (if the selection indicates keys of various lengths can be entered) that it provides information on the merits of shorter or longer pre-shared keys. The guidance must specify the allowable characters for pre- shared keys, and that list must be a super-set of the list contained in FIA_PSK_EXT.1.2.

Findings:	[AGD] Section 3.3.8.2 provides the following guidance to the administrators on the composition of strong text-based pre-shared keys, “Pre-shared keys on the TOE must be at least 22 characters in length and can be composed of any combination of upper-case letters (A-Z), lower case letters (a-z), numbers (0-9), and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”).The TOE supports pre-shared keys up to 127 bytes in length. ”

371 The evaluator shall confirm the operational guidance contains instructions for either entering bit-based pre-shared keys for each protocol identified in the requirement, or generating a bit-based pre-shared key (or both). The evaluator shall also examine the TSS to ensure it describes the process by which the bitbased pre-shared keys are generated (if the TOE supports this functionality), and confirm that this process uses the RBG specified in FCS_RBG_EXT.1 in the Base-PP.

Findings:	[AGD] Section 3.3.8.2 describes that the TOE is not able to generate bit-based pre-shared keys, but can accept bit-based pre-shared keys in hex format, generated off system with the following command: pre-shared-key hex [hex key]

5.3.1.3 Tests

372 The evaluator shall also perform the following tests for each protocol (or instantiation of a protocol, if performed by a different implementation on the TOE). Note that one or more of these tests can be performed with a single test case.

373 Test 1: The evaluator shall compose a pre-shared key of 22 characters that contains a combination of the allowed characters in accordance with the operational guidance, and demonstrates that a successful protocol negotiation can be performed with the key.

High-Level Test Description
As the TOE allows pre-shared keys up to 127 characters, configure the TOE and VPN Peer to use a key that contains all the possible characters and attempt to establish an IPsec tunnel.
Findings: PASS

374 Test 2 [conditional]: If the TOE supports pre-shared keys of multiple lengths, the evaluator shall repeat Test 1 using the minimum length; the maximum length; and an invalid length. The minimum and maximum length tests should be successful, and the invalid length must be rejected by the TOE.

High-Level Test Description
Configure the TOE and VPN peer to use a valid pre-shared key of minimum length (22 characters) and attempt to establish a connection. Connection successfully established. Configure the TOE and VPN peer to use a valid pre-shared key of maximum length (127 characters) and attempt to establish a connection. Connection successfully established.

High-Level Test Description
Attempt to configure the TOE to use an invalid pre-shared key length (128 characters). TOE does not accept the invalid pre-shared key.
Findings: PASS

375 Test 3 [conditional]: If the TOE does not generate bit-based pre-shared keys, the evaluator shall obtain a bit-based pre-shared key of the appropriate length and enter it according to the instructions in the operational guidance. The evaluator shall then demonstrate that a successful protocol negotiation can be performed with the key.

High-Level Test Description
Configure the TOE and VPN peer to use a hex based pre-shared key and attempt to establish a connection. Connection successfully established.
Findings: PASS

376 Test 4 [conditional]: If the TOE does generate bit-based pre-shared keys, the evaluator shall generate a bit-based pre-shared key of the appropriate length and use it according to the instructions in the operational guidance. The evaluator shall then demonstrate that a successful protocol negotiation can be performed with the key.

Note:	The TOE does not claim generation of bit-based pre-shared keys.

5.4 Packet Filtering (FPF)

5.4.1 FPF_RUL_EXT.1 Rules for Packet Filtering

FPF_RUL_EXT.1.1

5.4.1.1 TSS

377 The evaluator shall verify that the TSS provides a description of the TOE's initialization/startup process, which clearly indicates where processing of network packets begins to take place, and provides a discussion that supports the assertion that packets cannot flow during this process.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "These rules are operational as soon as interfaces are operational following startup of the TOE. There is no state during initialization/ startup that the access lists are not enforced on an interface. The initialization process first initializes the operating system, and then the networking daemons including the access list enforcement, prior to any daemons or user applications that potentially send network traffic. No incoming network traffic can be received before the access list functionality is operational."

378 The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This

could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "No incoming network traffic can be received before the access list functionality is operational." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FAU_GEN.1: "When the incoming traffic to the TOE exceeds what the interface can handle, the packets are dropped at the input queue itself…." [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_FLS.1/SelfTest: "Whenever a failure occurs within the TOE that results in the TOE ceasing operation, the TOE securely disables its interfaces to prevent the unintentional flow of any information to or from the TOE. The TOE shuts down by reloading and will continue to reload as long as the failures persist. This functionally prevents any failure of power-on self-tests, failure of integrity check of the TSF executable image, failure of noise source health tests from causing an unauthorized information flow. There are no failures that circumvent this protection."

5.4.1.2 Guidance Documentation

379 The operational guidance associated with this requirement is assessed in the subsequent test Evaluation Activities.

5.4.1.3 Tests

380 Test 1: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would otherwise be denied by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization.

High-Level Test Description
Ensure that all traffic rules have been cleared except for the rule to deny all traffic.
Start a wireshark capture on the outside workstation.
Using the inside workstation, send a steady stream of TCP packets to the outside network workstation using scamp fuzzing tool.
Restart the TOE and wait for it to be completely initialized.
Verify on the wireshark capture that no traffic from the inside workstation has penetrated the TOE.
Findings: PASS

381 Test 2: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would be permitted by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization and is only permitted once initialization is complete.

382 Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test Evaluation Activities.

High-Level Test Description
Ensure that all traffic rules have been cleared out except for a single rule that permits any traffic to be sent to the outside network. Start a wireshark capture on the outside workstation. Using the inside workstation, send a steady stream of packets to the outside workstation. Restart the TOE. Wait for the TOE to be completely initialized. Verify on the wireshark capture that no traffic from the inside workstation has penetrated the TOE until after initialization.
Findings: PASS

FPF_RUL_EXT.1.2

There are no Evaluation Activities specified for this element. Definition of Packet Filtering policy, association of operations with Packet Filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

FPF_RUL_EXT.1.3

There are no Evaluation Activities specified for this element. Definition of Packet Filtering policy, association of operations with Packet Filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

FPF_RUL_EXT.1.4

5.4.1.4 TSS

383 The evaluator shall verify that the TSS describes a Packet Filtering policy that can use the following fields for each identified protocol, and that the RFCs identified for each protocol are supported:

- IPv4 (RFC 791)
  - o Source address
  - o Destination Address
  - o Protocol
- IPv6 (RFC 2460)
  - o Source Address
  - o Destination Address
  - o Next Header (Protocol)
- TCP (RFC 793)
  - o Source Port
  - o Destination Port
- UDP (RFC 768)
  - o Source Port
  - o Destination Port

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1:

	"By implementing rules that defines the permitted flow of traffic between interfaces of the TOE for unauthenticated traffic. These rules control whether a packet is transferred from one interface to another based on: 1. presumed address of source 2. presumed address of destination 3. transport layer protocol (or next header in IPv6) 4. Service used (UDP or TCP ports, both source and destination) 5. Network interface on which the connection request occurs "These rules are supported for the following protocols: RFC 791(IPv4); RFC 2460 (IPv6); RFC 793 (TCP); RFC 768 (UDP)."

384 The evaluator shall verify that the TSS describes how conformance with the identified RFCs has been determined by the TOE developer (e.g., third party interoperability testing, protocol compliance testing).

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "TOE compliance with these protocols is verified via regular quality assurance, regression, and interoperability testing."

385 The evaluator shall verify that each rule can identify the following actions: permit, discard, and log.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "When network packets are received on a TOE interface, the TOE verifies whether the network traffic is allowed or not and performs one of the following actions, pass/not pass information, as well as optional logging."

386 The evaluator shall verify that the TSS identifies all interface types subject to the Packet Filtering policy and explains how rules are associated with distinct network interfaces. Where interfaces can be grouped into a common interface type (e.g., where the same internal logical path is used, perhaps where a common device driver is used), they can be treated collectively as a distinct network interface.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: "An authorized administrator can define the traffic that needs to be protected by configuring access lists (permit, deny, log) and applying these access lists to interfaces using access and crypto map sets. "

5.4.1.5 Guidance Documentation

387 The evaluators shall verify that the operational guidance identifies the following protocols as being supported and the following attributes as being configurable within Packet filtering rules for the associated protocols:

- IPv4 (RFC 791)
    - o Source address

- Destination Address
- Protocol

- IPv6 (RFC 2460)
  - Source Address
  - Destination Address
  - Next Header (Protocol)
- TCP (RFC 793)
  - Source Port
  - Destination Port
- UDP (RFC 768)
  - Source Port
  - Destination Port

Findings:	[AGD] Section 3.3.4 identifies all the above protocols as being supported and the above attributes as configurable.

388 The evaluator shall verify that the operational guidance indicates that each rule can identify the following actions: permit, discard, and log.

Findings:	[AGD] Section 3.3.5 identifies that packet filtering rules are configured using access lists and each rule in the access lists can be set to Permit or Deny (Discard). To log any rule that matches an access list append the “log-input” keyword at the end of the acl statement.

389 The evaluator shall verify that the operational guidance explains how rules are associated with distinct network interfaces.

Findings:	[AGD] Section 3.3.4 provides instructions on how to associate the access lists with distinct network interfaces.

390 The guidance may describe the other protocols contained within the ST (e.g., IPsec, IKE, potentially HTTPS, SSH, and TLS) that are processed by the TOE. The evaluator shall ensure that it is made clear what protocols were not considered as part of the TOE evaluation.

Findings:	[AGD] Section 3.3.4 identifies the following protocols as part of the TOE evaluation, all other protocols are considered not part of the TOE evaluation: • IPv4(RFC791) • IPv6 (RFC 2460) • TCP (RFC 793) • UDP (RFC 768) • IKEv2 (RFC 5996) • IPsec ESP (RFCs 4301, 4303) • SSH (RFCs 4251, 4252, 4253, 4254,6668)

5.4.1.6 Tests

391 The evaluator shall perform the following tests:

392 Test 1: The evaluator shall use the instructions in the operational guidance to test that packet filter rules can be created that permit, discard, and log packets for each of the following attributes:

- IPv4
  - Source address
  - Destination Address
  - Protocol
- IPv6

- Source Address
- Destination Address
- Next Header (Protocol)

- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

Note:	The construction and effectiveness of the rules shall be tested as part of FPF_RUL_EXT.1.6 in accordance with the note given in the MOD_VPNGW_v1.1 SD.

393 Test 2: The evaluator shall repeat Test 1 above for each distinct network interface type supported by the TOE to ensure that Packet filtering rules can be defined for each all supported types.

394 Note that these test activities should be performed in conjunction with those of FPF_RUL_EXT.1.6 where the effectiveness of the rules is tested; here the evaluator is just ensuring the guidance is sufficient and the TOE supports the administrator creating a ruleset based on the above attributes. The test activities for FPF_RUL_EXT.1.6 define the protocol/attribute combinations required to be tested. If those combinations are configured manually, that will fulfill the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

Note	The construction and effectiveness of the rules shall be tested as part of FPF_RUL_EXT.1.6 in accordance with the note given in the MOD_VPNGW_v1.1 SD.

FPF_RUL_EXT.1.5

5.4.1.7 TSS

395 The evaluator shall verify that the TSS describes the algorithm applied to incoming packets, including the processing of default rules, determination of whether a packet is part of an established session, and application of administrator defined and ordered ruleset.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1: “The TOE is capable of inspecting network packet header fields to determine if a packet is part of an established session or not. ACL rules still apply to packets that are part of an ongoing session. “Packets will be dropped unless a specific rule has been set up to allow the packet to pass (where the attributes of the packet match the attributes in the rule and the action associated with the rule is to pass traffic). This is the default action that occurs on an interface if no ACL rule is found. If a packet arrives that does not meet any rule, it is

	expected to be dropped. Rules are enforced on a first match basis from the top down. As soon as a match is found the action associated with the rule is applied."

5.4.1.8 Guidance Documentation

396 The evaluator shall verify that the operational guidance describes how the order of Packet filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

Findings:	[AGD] Section 3.3.4 states, "Traffic matching is done based on a top-down approach in the access list. The first entry that a packet matches will be the one applied to it."

5.4.1.9 Tests

397 The evaluator shall perform the following tests:

398 Test 1: The evaluator shall devise two equal Packet Filtering rules with alternate operations – permit and discard. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation.

High-Level Test Description
After clearing all rules on the TOE, construct a rule that permits specific traffic and one that denies the exact same traffic. Start packet tracing on both the outside and inside networks. Send traffic that meets the rule and show that it is logged and accepted. Then reorder the rules such that the deny rule is ordered first. Send traffic that meets the rule and show that it is logged and dropped. Do this for both IPv4 and IPv6 traffic.
Findings: PASS

399 Test 2: The evaluator shall repeat the procedure above, except that the two rules should be devised where one is a subset of the other (e.g., a specific address vs. a network segment). Again, the evaluator should test both orders to ensure that the first is enforced regardless of the specificity of the rule.

High-Level Test Description
After clearing all rules on the TOE, construct a rule (i) that denies all traffic to a specific address (called A) and (ii) one that permits traffic to an entire network segment that encompasses both the inside network target address (T) and the specific address (A) such that address A != T. Start packet tracing on both the outside and inside networks. Send traffic from the outside network that meets rule (i) and show that it is logged and dropped. Then reorder the rules such that the rule (ii) is ordered first. Send traffic that meets the rule and show that it is logged and accepted. Do this for both IPv4 and IPv6 traffic.
Findings: PASS

FPF_RUL_EXT.1.6

[Modified by TD 0597]

5.4.1.10 TSS

400 The evaluator shall verify that the TSS describes the process for applying Packet Filtering rules and also that the behavior (either by default, or as configured by the administrator) is to discard packets when there is no rule match. The evaluator shall verify the TSS describes when the IPv4/IPv6 protocols supported by the TOE differ from the full list provided in the RFC Values for IPv4 and IPv6 table.

Findings: [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPF_RUL_EXT.1:

"Packets will be dropped unless a specific rule has been set up to allow the packet to pass (where the attributes of the packet match the attributes in the rule and the action associated with the rule is to pass traffic). This is the default action that occurs on an interface if no ACL rule is found. If a packet arrives that does not meet any rule, it is expected to be dropped. Rules are enforced on a first match basis from the top down. As soon as a match is found the action associated with the rule is applied."

"These rules are supported for the following protocols: RFC 791(IPv4); RFC 2460 (IPv6); RFC 793 (TCP); RFC 768 (UDP)."

The TSS does not indicate the IPv4/IPv6 protocols supported by the TOE differ from the claimed RFCs.

5.4.1.11 Guidance Documentation

401 The evaluator shall verify that the operational guidance describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the operational guidance provides the appropriate instructions to configure the behavior to discard packets with no matching rules. The evaluator shall verify that the operational guidance describes the range of IPv4/IPv6 protocols supported by the TOE.

Findings: Per [AGD] Section 3.3.5, If no explicit 'permit' exists within the crypto map, but there is no explicit or implicit deny, then the packet is allowed to bypass the tunnel in plaintext.
In [AGD] Section 3.3.4 it states that [MOD_VPN] requires the TOE to be configured with a default drop all rule and provides instructions on configuring the TOE to drop all packets by default.

"The following attributes, at a minimum, are configurable within Packet filtering rules for the associated protocols:

- IPv4
  - Source address
  - Destination Address
  - Protocol
- IPv6
  - Source address
  - Destination Address
  - Next Header (Protocol)
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port

o Destination Port”

The operational guidance describes the following range of IPv4/IPv6 protocols supported by the TOE:
“The TOE supports the following packet filtering firewall rule set of the following protocols:
- IPv4 (RFC 791)
- IPv6 (RFC 2460)
- TCP (RFC 793)
- UDP (RFC 768)
- IKEv2 (RFC 5996)
- IPsec ESP (RFCs 4301, 4303)
- SSH (RFCs 4251, 4252, 4253, 4254, 6668)”

5.4.1.12 Tests

402 The evaluator shall perform the following tests:

403 Test 1: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
After clearing all rules on the TOE, construct a rule for each of the listed conditions. Generate traffic that will match the given rule and show that the packet is permitted.
Findings: PASS

404 Test 2: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
After clearing all rules on the TOE, construct a rule for each of the listed conditions. Generate traffic that will match the given rule and show that the packet is denied.
Findings: PASS

405 Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address,

specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Construct an access-list to allow all IP traffic from SOURCE_SUBNET1 to DEST_SUBNET1, in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Construct an access-list to deny all IP traffic from SOURCE_SUBNET2 to DEST_SUBNET2, in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Send traffic from source ip to destination ip that is outside the scope of all source and destination addresses configured above (SOURCE_SUBNET3 and DEST_SUBNET3). Ensure the traffic is denied.
Findings: PASS

406 Test 4: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
After clearing all rules on the TOE, construct a rule for each of the listed conditions. Generate traffic that will match the given rule and show that the packet is permitted.
Findings: PASS

407 Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported

protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
After clearing all rules on the TOE, construct a rule for each of the listed conditions. Generate traffic that will match the given rule and show that the packet is denied.
Findings: PASS

408 Test 6: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are dropped (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Construct an access-list to allow all IP traffic from SOURCE_SUBNET1 to DEST_SUBNET1, in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Construct an access-list to deny all IP traffic from SOURCE_SUBNET2 to DEST_SUBNET2, in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Send traffic from source ip to destination ip that is outside the scope of all source and destination addresses configured above (SOURCE_SUBNET3 and DEST_SUBNET3). Ensure the traffic is denied.
Findings: PASS

409 Test 7: The evaluator shall configure the TOE to permit and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination TCP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged.

High-Level Test Description
Create an access list with the following 3 rules: • Allow TCP traffic with specific source port and destination port. • Allow TCP traffic with specific source port and any destination port. • Allow TCP traffic with any source port and specific destination port.

High-Level Test Description
Send crafted TCP traffic that, matches one of 3 rules, from the inside workstation to the outside workstation and confirm that the TOE permits and logs the traffic.
Findings: PASS

410 Test 8: The evaluator shall configure the TOE to discard and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination TCP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged.

High-Level Test Description
Create an access list with the following 4 rules: • Deny TCP traffic with specific source port and destination port. • Deny TCP traffic with specific source port and any destination port. • Deny TCP traffic with any source port and specific destination port. • Allow all IP traffic that does not match the preceding rules. Send crafted TCP traffic that, matches one of 3 rules, from the inside workstation to the outside workstation and confirm that the TOE discards and logs the traffic.
Findings: PASS

411 Test 9: The evaluator shall configure the TOE to permit and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Here the evaluator ensures that the UDP port 500 (IKE) is included in the set of tests.

High-Level Test Description
Create an access list with the following 3 rules: • Allow UDP traffic with specific source port and destination port. • Allow UDP traffic with specific source port and any destination port. • Allow UDP traffic with any source port and specific destination port. Send crafted UDP traffic that, matches one of 3 rules, from the inside workstation to the outside workstation and confirm that the TOE permits and logs the traffic.
Findings: PASS

412 Test 10: The evaluator shall configure the TOE to discard and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Again, the evaluator ensures that UDP port 500 is included in the set of tests.

High-Level Test Description
Create an access list with the following 4 rules: • Deny UDP traffic with specific source port and destination port. • Deny UDP traffic with specific source port and any destination port. • Deny UDP traffic with any source port and specific destination port. • Allow all IP traffic that does not match the preceding rules. Send crafted UDP traffic that, matches one of 3 rules, from the inside workstation to the outside workstation and confirm that the TOE discards and logs the traffic.
Findings: PASS

5.5 Protection of the TSF (FPT)

5.5.1 FPT_FLS.1/SelfTest Fail Secure (Self-Test Failures)

5.5.1.1 TSS

413 The evaluator shall ensure the TSS describes how the TOE ensures a shutdown upon a self-test failure, a failed integrity check of the TSF executable image, or a failed health test of the noise source. If there are instances when a shut-down does not occur, (e.g., a failure is deemed non- security relevant), the evaluator shall ensure that those cases are identified and a rationale is provided that supports the classification and justifies why the TOE's ability to enforce its security policies is not affected in any such instance.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_FLS.1/SelfTest: "Whenever a failure occurs within the TOE that results in the TOE ceasing operation, the TOE securely disables its interfaces to prevent the unintentional flow of any information to or from the TOE. The TOE shuts down by reloading and will continue to reload as long as the failures persist. This functionally prevents any failure of power-on self-tests, failure of integrity check of the TSF executable image, failure of noise source health tests from causing an unauthorized information flow. There are no failures that circumvent this protection."

5.5.1.2 Guidance Documentation

414 The evaluator shall verify that the operational guidance provides information on the self-test failures that can cause the TOE to shut down and how to diagnose the specific failure that has occurred, including possible remediation steps if available.

Findings:	[AGD] Sections 3.2.4 and 3.2.5 identify that if any self-test fails the TOE transitions to an error state, and in the error state, all secure data transmission is halted and the TOE outputs an error indicating the failure. The [AGD] advises the administrator to refer to the System Message Overview online documentation for troubleshooting and remediation steps available.

5.5.1.3 Tests

415 There are no test Evaluation Activities for this SFR.

5.5.2 FPT_TST_EXT.3 TSF Self-Test with Defined Methods

5.5.2.1 TSS

416 The evaluator verifies that the TSS describes the method used to perform self-testing on the TSF executable code, and that this method is consistent with what is described in the SFR.

Findings:	[ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FPT_TST_EXT.3: "For testing of the TSF, the TOE automatically runs checks and tests at startup and during resets and periodically during normal operation to ensure the TOE is operating correctly, including checks of image integrity and all cryptographic functionality." "The integrity of stored TSF executable code when it is loaded for execution can be verified through the use of RSA and Elliptic Curve Digital Signature algorithms." "The Software Integrity Test is run automatically whenever the IOS system images is loaded and confirms that the image file that's about to be loaded has maintained its integrity." The method used to perform self-testing of the TSF executable code is consistent with selections made in the FPT_TST_EXT.3 component.

5.5.2.2 Guidance Documentation

417 There are no operational guidance Evaluation Activities for this SFR.

5.5.2.3 Tests

418 There are no test Evaluation Activities for this SFR.

5.6 Trusted Path/Channels (FTP)

5.6.1 FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications)

5.6.1.1 TSS

419 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Findings:	See prior FTP_ITC.1 TSS findings.

5.6.1.2 Guidance Documentation

420 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Findings:	IPsec is the only allow protocol to secure trusted channels between TOE and authorized IT entities. [AGD] Section 3.3.8 Configuration of IPsec – contains instructions for establishing IPsec with all authorized IT entities. If an IPsec session

	with a peer is unexpectedly interrupted, the connection will be broken. The IPsec session will be re-established (a new SA set up) once the peer is back online.

5.6.1.3 Tests

421 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications. Additional evaluation testing for IPsec is covered in FCS_IPSEC_EXT.1.

Note:	All trusted channels in the evaluated configuration use IPsec. Test steps and findings are found in the associated Base-PP (NDcPP2.2E) documentation.

5.7 Security Management (FMT)

5.7.1 FMT_SMF.1/VPN Specification of Management Functions (VPN)

5.7.1.1 TSS

422 The evaluator shall examine the TSS to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the TSS identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	The evaluator confirmed the management functions specified in the FMT_SMF.1/VPN element are present within the FMT_SMF.1 section of Table 19 in section 6.1 of the [ST]. [ST] Section 6.1 TOE Security Functional Requirement Measures > Table 19 > FMT_SMF.1: “The management functionality of the TOE is provided through the TOE CLI.”

5.7.1.2 Guidance Documentation

423 The evaluator shall examine the operational guidance to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the operational guidance identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	Management functions specified in FMT_SMF.1/VPN are identified as the following: Definition of packet filtering rules; Association of packet filtering rules to network interfaces; Ordering of packet filtering rules by priority; The evaluator determined that all management functions specified in FMT_SMF.1/VPN were addressed in the guidance documentation Assurance Activities for FPF_RUL_EXT.1. The logical interface used to perform these functions was previously confirmed in FMT_SMF.1 as the TOE CLI which can be accessed from the Local Console or through SSH.

	The Local Console is described as “This includes any IT Environment Console that is directly connected to the TOE via the Serial Console Port and is used by the TOE administrator to support TOE administration.”

5.7.1.3 Tests

424 The evaluator tests management functions as part of testing the SFRs identified in sections 2.2, 3, and 4. No separate testing for FMT_SMF.1/VPN is required unless one of the management functions in FMT_SMF.1.1/VPN has not already been exercised under any other SFR.

Findings:	The evaluator tested management functions as part of testing the SFRs identified in sections 2.2, 3, and 4.

6 Evaluation Activities for Security Assurance Requirements

6.1 ASE: Security Target

425 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.

426 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:	See above sections.

6.2 ADV: Development

427 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.

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

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

430 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.

431 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 SSH CLI, local serial port), there were no areas that were deficient.

432 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.

433 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.

6.3 AGD: Guidance

434 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.

435 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 documentation is available for public download from Cisco's web site (https://www.cisco.com/c/en/us/solutions/industries/government/global-government-certifications/common-criteria.html)

436 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.

437 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:	[AGD] section 3.2.3 provides instructions for configuring FIPS mode which addresses this requirement.

438 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:	[AGD] section 3.3 specifies the interfaces used by the TOE in the evaluated configuration.

439 5.3.1.5 Evaluation Activity:

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 TD 0536]

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:	[AGD] section 3.2.3 provides instructions for configuring FIPS mode which addresses this requirement. [AGD] sections 2 provide instructions for the download and verification of the TOE updates. [AGD] section 1.6 provides a list of excluded functions.

440 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:	[AGD] section 1.5 provides instructions for configuration of the Operational Environment.

441 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:	[AGD] section 1.5 provides instructions for configuration of the Operational Environment. The evaluator verified that this addresses all claimed platforms.

442 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.

443 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 help instil a culture of secure manageability within a larger operational environment.

444 5.3.2.5 Evaluation Activity:

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

The preparative procedures must:

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

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

Findings:	The [AGD] section 3.2 specifies the secure installation of the TOE to provide a protected interface. No passwords have been identified that have default values.

7 Vulnerability Assessment

445 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.

446 [Modified by TD 0547] 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).

447 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: Vendor security advisories: https://tools.cisco.com/security/center/publicationListing.x CISA Known Exploited Vulnerabilities Catalog https://www.cisa.gov/known-exploited-vulnerabilities-catalog NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search a) Common Vulnerabilities and Exposures: http://cve.mitre.org/cve/ https://www.cvedetails.com/vulnerability-search.php b) US-CERT: http://www.kb.cert.org/vuls/html/search Offensive Security Exploit Database: https://www.exploit-db.com/ Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities Google Type 1 Hypothesis searches were conducted on September 19th, 2022, and

included the following search terms:
-ISR900 C921-4P
-ISR900 C921J-4P
-ISR900 C926-4P
-ISR900 C927-4P
-ISR900 C931-4P
-Intel Atom C3558 processor
-IOS 15.9 operating system
-IC2Mrel5
-OpenSSH - 1:5.5p1
-ACT2Lite

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

Type-2 hypotheses identified for the NDcPP are found in Section 3.

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

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