www.GossamerSec.com

Assurance Activity Report for Cisco Catalyst 9200CX/9500X/9600X Series Switches 17.15

Version 1.0
08/04/25

Prepared by:
Gossamer Security Solutions
Accredited Security Testing Laboratory – Common Criteria Testing
Columbia, MD 21045

Prepared for:
National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

Document: AAR-VID11586

Revision History

Revision	Date	Authors	Summary
Version 0.1	07/16/25	Miller	Initial draft
Version 0.2	07/31/25	Miller	Addressed ECR comments
Version 1.0	08/04/25	Miller	Updated ECR comment responses

The TOE Evaluation was Sponsored by:
Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95134

Evaluation Personnel:
- Julia Miller

Common Criteria Versions:
- Common Criteria for Information Technology Security Evaluation Part 1: Introduction, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1, Revision 5, April 2017

Common Evaluation Methodology Versions:
- Common Methodology for Information Technology Security Evaluation, Evaluation Methodology, Version 3.1, Revision 5, April 2017

Table of Contents

1. Introduction 6
1.1 Equivalence 6
1.1.1 Evaluated Platform Equivalence 6
1.1.2 CAVP Equivalence 7
1.2 References....8
2. Protection Profile SFR Assurance Activities 10
2.1 Security audit (FAU) 10
2.1.1 Audit Data Generation (NDcPP30e:FAU_GEN.1) 10
2.1.2 User identity association (NDcPP30e:FAU_GEN.2)....12
2.1.3 Protected Audit Event Storage (NDcPP30e:FAU_STG_EXT.1) 13
2.2 Cryptographic support (FCS) 18
2.2.1 Cryptographic Key Generation - per TD0923 (NDcPP30e:FCS_CKM.1) 18
2.2.2 Cryptographic Key Establishment (NDcPP30e:FCS_CKM.2)....23
2.2.3 Cryptographic Key Destruction (NDcPP30e:FCS_CKM.4)....26
2.2.4 Cryptographic Operation (AES Data Encryption/Decryption) (NDcPP30e:FCS_COP.1/DataEncryption) 28
2.2.5 Cryptographic Operation (Hash Algorithm) (NDcPP30e:FCS_COP.1/Hash)....32
2.2.6 Cryptographic Operation (Keyed Hash Algorithm) (NDcPP30e:FCS_COP.1/KeyedHash) 35
2.2.7 Cryptographic Operation (Signature Generation and Verification) - per TD0923 (NDcPP30e:FCS_COP.1/SigGen)....36
2.2.8 IPsec Protocol - per TD0868 (NDcPP30e:FCS_IPSEC_EXT.1) 38
2.2.9 Random Bit Generation (NDcPP30e:FCS_RBG_EXT.1)....54
2.2.10 SSH Protocol - per TD0909 (SSH10:FCS_SSH_EXT.1) 56
2.2.11 SSH Protocol - Server - per TD0682 (SSH10:FCS_SSHS_EXT.1) 63
2.3 Identification and authentication (FIA) 64
2.3.1 Authentication Failure Management (NDcPP30e:FIA_AFL.1)....64
2.3.2 Password Management (NDcPP30e:FIA_PMG_EXT.1) 67
2.3.3 Protected Authentication Feedback (NDcPP30e:FIA_UAU.7) 69
2.3.4 User Identification and Authentication - per TD0900 (NDcPP30e:FIA_UIA_EXT.1)....70

2.3.5 X.509 Certificate Validation (NDcPP30e:FIA_X509_EXT.1/Rev)....73
2.3.6 X.509 Certificate Authentication (NDcPP30e:FIA_X509_EXT.2)....77
2.3.7 X.509 Certificate Requests (NDcPP30e:FIA_X509_EXT.3)....79
2.4 Security management (FMT)....80
2.4.1 Management of security functions behaviour (NDcPP30e:FMT_MOF.1/ManualUpdate)....80
2.4.2 Management of TSF Data (NDcPP30e:FMT_MTD.1/CoreData)....82
2.4.3 Management of TSF Data (NDcPP30e:FMT_MTD.1/CryptoKeys)....83
2.4.4 Specification of Management Functions - per TD0880 (NDcPP30e:FMT_SMF.1)....84
2.4.5 Restrictions on Security Roles (NDcPP30e:FMT_SMR.2)....87
2.5 Protection of the TSF (FPT)....88
2.5.1 Protection of Administrator Passwords (NDcPP30e:FPT_APW_EXT.1)....88
2.5.2 Protection of TSF Data (for reading of all symmetric keys) (NDcPP30e:FPT_SKP_EXT.1)....89
2.5.3 Reliable Time Stamps (NDcPP30e:FPT_STM_EXT.1)....90
2.5.4 TSF testing - per TD0836 (NDcPP30e:FPT_TST_EXT.1)....92
2.5.5 Trusted update (NDcPP30e:FPT_TUD_EXT.1)....95
2.6 TOE access (FTA)....98
2.6.1 TSF-initiated Termination (NDcPP30e:FTA_SSL.3)....98
2.6.2 User-initiated Termination (NDcPP30e:FTA_SSL.4)....100
2.6.3 TSF-initiated Session Locking (NDcPP30e:FTA_SSL_EXT.1)....101
2.6.4 Default TOE Access Banners (NDcPP30e:FTA_TAB.1)....102
2.7 Trusted path/channels (FTP)....102
2.7.1 Inter-TSF trusted channel (NDcPP30e:FTP_ITC.1)....103
2.7.2 Trusted Path (NDcPP30e:FTP_TRP.1/Admin)....105
3. Protection Profile SAR Assurance Activities....108
3.1 Security Target (ASE)....108
3.2 Development (ADV)....108
3.2.1 Basic functional specification (ADV_FSP.1)....108
3.3 Guidance documents (AGD)....109
3.3.1 Operational user guidance (AGD_OPE.1)....109
3.3.2 Preparative procedures (AGD_PRE.1)....111

3.4 Life-cycle support (ALC)......112
3.4.1 Labelling of the TOE (ALC_CMC.1)......112
3.4.2 TOE CM coverage (ALC_CMS.1)......113
3.4.3 Flaw reporting procedures (ALC_FLR.2)......114
3.5 Tests (ATE)......115
3.5.1 Independent testing - conformance (ATE_IND.1)......115
3.6 Vulnerability assessment (AVA)......117
3.6.1 Vulnerability survey (AVA_VAN.1)......117
3.6.2 Additional Flaw Hypothesis (AVA_VAN.1)......119

1. Introduction

This document presents evaluations results of the Cisco Catalyst 9200CX/9500X/9600X Series Switches 17.15 NDcPP30e/SSH10 evaluation. This document contains a description of the assurance activities and associated results as performed by the evaluators.

1.1 Equivalence

This section explains why the test subset was adequate to address all product installations.

1.1.1 Evaluated Platform Equivalence

The TOE is the Cisco Catalyst 9200CX/9300LM/9300X/9500X Series Switches all running Internetworking Operating System (IOS)-XE 17.15.

- Catalyst 9200CX models: C9200CX-12T-2X2G, **C9200CX-12P-2X2G**, C9200CX-8P-2X2G, C9200CX-8UXG-2X
- Catalyst 9500X models: **C9500X-28C8D**, C9500X-60L4D
- Catalyst 9600X models:
  - Chassis: C9606R
  - With the following Supervisor models: C9600X-SUP2
  - With the following Line Card models: C9600-LC-40YL4CD, C9600X-LC-32CD, C9600X-LC-56YL4C

The evaluator performed full testing on the following platforms (**bolded** in the lists above) and microarchitectures.

- C9200CX-12P-2X2G – Xilinx ZU3EG (ARM Cortex-A53)
- C9500X-28C8D – Intel Xeon D-1564N (Broadwell)

There are two distinct software images for each hardware family. Software images starting with “cat9k_lite_iosxe” are used in the 9200CX family of devices, and images starting with “cat9k_iosxe” are used on the 9500X and 9600X family of devices. Both images provide the same functionality and services. The only distinction is that each image is compiled for the respective CPU architecture of each device family.

All Security Functionality) is implemented in the cryptographic library, which is the IOS Common Cryptographic Module (IC2M)).

While there are different models in each family, they differ primarily in physical form factor, number and types of connections and slots, and relative performance. These differing characteristics primarily affect only non-TSF relevant functionality (such as throughput, processing speed, number and type of network connections supported, number of concurrent connections supported, and amount of storage). All TOE security functions are implemented in software, and the TOE security behavior is the same on all the devices for each of the SFRs defined by the

Security Target. These SFRs are instantiated by the same version of the TOE software and in the same way on every platform. Therefore, the evaluation lab has determined that the models chosen for testing covers all claimed models.

1.1.2 CAVP Equivalence

The TOE is the Cisco Catalyst 9200CX/9500X/9600X all running Internetworking Operating System (IOS)-XE 17.15.

- Catalyst 9200CX models: C9200CX-12T-2X2G, **C9200CX-12P-2X2G**, C9200CX-8P-2X2G, C9200CX-8UXG-2X
- Catalyst 9500X models: **C9500X-28C8D**, C9500X-60L4D
- Catalyst 9600X models:
    - Chassis: C9606R
    - With the following Supervisor models: C9600X-SUP2
    - With the following Line Card models: C9600-LC-40YL4CD, C9600X-LC-32CD, C9600X-LC-56YL4C

The evaluator performed full testing on the following platforms (**bolded** in the lists above) and microarchitecture.

- C9200CX-12P-2X2G - Xilinx ZU3EG (ARM Cortex-A53)
- C9500X-28C8D - Intel Xeon D-1564N (Broadwell)

The TOE provides the cryptography to support all security functions. All algorithms claimed have Cryptographic Algorithm Validation Program (CAVP certificates running on the processor specified in the table below.

Hardware Model	Processor
Catalyst 9200CX Series	Xilinx ZU3EG (ARM Cortex-A53)
Catalyst C9500X Series	Intel Xeon D-1564N (Broadwell)
Catalyst 9600X Series	Intel Xeon D-1573N (Broadwell)

All cryptography is implemented using the IOS Common Cryptographic Module (IC2M) version Rel5a cryptographic module.

All the algorithms claimed have CAVP certificates as shown in the table below.

SFR	Selection	Algorithm	Implementation	Standard	Certificate Number
FCS_CKM.1 – Cryptographic Key Generation	3072	RSA	IC2M	FIPS PUB 186-4	A1462

SFR	Selection	Algorithm	Implementation	Standard	Certificate Number
FCS_CKM.1 – Cryptographic Key Generation	P-384	ECDSA	IC2M	FIPS PUB 186-4	A1462
FCS_CKM.2 – Cryptographic Key Establishment	P-384	KAS-ECC	IC2M	NIST SP 800-56A Rev 3	A1462
FCS_COP.1/DataEncryption – AES Data Encryption/Decryption	AES-CBC-256 AES-GCM-256	AES	IC2M	ISO/IEC 18033-3 (AES) ISO/IEC 10116 (CBC) ISO/IEC 19772 (GCM)	A1462
FCS_COP.1/SigGen – Cryptographic Operation (Signature Generation and Verification)	3072	RSA	IC2M	FIPS PUB 186-4	A1462
	P-384	ECDSA	IC2M	FIPS PUB 186-4	A1462
FCS_COP.1/Hash – Cryptographic Operation (Hash Algorithm)	SHA-256 SHA-512	SHS	IC2M	ISO/IEC 10118-3:2004	A1462
FCS_COP.1/KeyedHash – Cryptographic Operation (Keyed Hash Algorithm)	HMAC-SHA-256	HMAC	IC2M	ISO/IEC 9797-2:2011	A1462
FCS_RBG_EXT.1– Random Bit Generation	CTR_DRBG (AES) 256 bits	DRBG	IC2M	ISO/IEC 18031:2011	A1462

Table 1. CAVP Certificates

The evaluator performed full algorithm testing on the 9200x, 9500x, and 9600x hardware.

1.2 References

The following evidence was used to complete the Assurance Activities:

- Cisco Catalyst 9200CX/9500X/9600X Series Switches 17.15 Security Target, Version 0.5, July 31, 2025 (**ST**)

- Cisco Catalyst 9200CX/9500X/9600X Series Switches 17.15 Common Criteria Configuration Guide, Version 0.4, July 31, 2025 (**Admin Guide**)

2. Protection Profile SFR Assurance Activities

This section of the AAR identifies each of the assurance activities in the claimed Protection Profiles and describes the findings in each case.

2.1 Security audit (FAU)

2.1.1 Audit Data Generation (NDcPP30e:FAU_GEN.1)

2.1.1.1 NDcPP30e:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.1.2 NDcPP30e:FAU_GEN.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For the administrative task of generating/import of, changing, or deleting of cryptographic keys as defined in FAU_GEN.1.1c, the TSS shall identify what information is logged to identify the relevant key.

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.

Section 6 (FAU_GEN.1) of the ST states each of the events specified in the audit record is in enough detail to identify the user for which the event is associated, when the event occurred, the outcome of the event, and the type of event that occurred such as generating keys, including the key identifier.

The TOE is not distributed.

Component Guidance Assurance Activities: 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).

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.

Section "Auditing" in the **Admin Guide** provides a table identifying all auditable events required by FAU_GEN.1 and provides examples of each audit record for each SFR (where applicable). The content of these audit examples match the content of audits generated during testing as recorded in the DTR.

From a review of the ST, the Guidance and through testing the evaluator also determined that the guidance contains all of the administrative actions and their associated audit events that are relevant to the PP and to use of the TOE. These administrative actions are consistent with the security requirements implemented in the TOE and were found to have appropriate management capabilities identified in the guidance documentation.

Component Testing Assurance Activities: 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 identify and authentication (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.

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.

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

The evaluator created a list of the required audit events. The evaluator then collected the audit events when running the other security functional tests described by the protection profiles. For example, the required event for FPT_STM.1 is discontinuous change to time. The evaluator collected audit records when modifying the clock using administrative commands. The evaluator then recorded the relevant audit events in the proprietary Detailed Test Report (DTR). The security management events are handled in a similar manner. When the administrator was required to set a value for testing, the audit record associated with the administrator action was collected and recorded in the DTR.

2.1.2 User identity association (NDcPP30e:FAU_GEN.2)

2.1.2.1 NDcPP30e:FAU_GEN.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

See NDcPP30e:FAU_GEN.1.

Component Guidance Assurance Activities: The TSS and Guidance Documentation requirements for FAU_GEN.2 are already covered by the TSS and Guidance Documentation requirements for FAU_GEN.1.

See NDcPP30e:FAU_GEN.1.

Component Testing Assurance Activities: This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.

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.

See NDcPP30e:FAU_GEN.1.

2.1.3 Protected Audit Event Storage (NDcPP30e:FAU_STG_EXT.1)

2.1.3.1 NDcPP30e:FAU_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.2 NDcPP30e:FAU_STG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.3 NDcPP30e:FAU_STG_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.4 NDcPP30e:FAU_STG_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.5 NDcPP30e:FAU_STG_EXT.1.5

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.6 NDcPP30e:FAU_STG_EXT.1.6

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

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.

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit data to an external IT entity can be done in real-time, periodically, or both. In the case where the TOE is capable of performing transmission periodically, the evaluator shall verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible acceptable frequency for the transfer of audit data.

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).

The evaluator shall examine the TSS to ensure it describes the amount of audit data that can be stored locally and how these records are protected against unauthorized modification or deletion.

The evaluator shall examine the TSS to ensure it describes the method implemented for local logging, including format (e.g. buffer, log file, database) and whether the logs are persistent or non-persistent.

The evaluator shall examine the TSS to ensure it describes the conditions that must be met for authorized deletion of audit records.

The evaluator shall examine the TSS to ensure 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.

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.

Section 6 (FAU_STG_EXT.1) of the ST states the TOE is a standalone device configured to export syslog records to a specified, external syslog server in real-time. The TOE protects communications with an external syslog server using 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 to the syslog server.

Fow audit records stored internally to the TOE the audit records are stored in a non-persistent 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 2,147,483,647 bytes of available disk space.

Only Authorized Administrators can clear the local logs, and local audit records are stored in a directory that does not allow Administrators to modify the contents.

The TOE is not distributed.

Component Guidance Assurance Activities: 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.

The evaluator shall also examine the guidance documentation to ensure 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.

The evaluator shall examine the guidance documentation to ensure it describes any configuration required for protection of the locally stored audit data against unauthorized modification or deletion.

If the storage size is configurable, the evaluator shall review the Guidance Documentation to ensure it contains instructions on specifying the required parameters.

If more than one selection is made for FAU_STG_EXT.1.5, the evaluator shall review the Guidance Documentation to ensure it contains instructions on specifying which action is performed when the local storage space is full.

Section "IPsec" of the Admin Guide provides instructions for how to establish the trusted channel to the audit server, including generating a crypto key pair for IPsec, creating trustpoints, configuring the IKEv2 profile, configuring IPsec transform sets, configuring SA lifetimes, configuring crypto maps, access control lists, and the reference identifier.

Section "Enable Remote Syslog Server" of the Admin Guide provides instructions for using the 'logging host' command to enable the TOE to transmit audit data to a specified external audit server. The section also states that when an audit event is generated, it is simultaneously sent to the external server and the local store.

Section "Configure Local Logging Buffer Size" of the Admin Guide provides instructions for how to configure the size of the local logging buffer. This section notes that it is recommended to not make the buffer size too large because the TOE could run out of memory for other tasks and it is recommended to set it to at least 150000000. Additionally, if the local storage space for audit data is full the TOE will overwrite the oldest audit record to make room for the new audit record.

Component Testing Assurance Activities: Testing of secure transmission of the audit data externally (FTP_ITC.1) and, where applicable, intercomponent (FPT_ITT.1 or FTP_ITC.1) shall be performed according to the assurance activities for the particular protocol(s).

The evaluator shall perform the following additional test 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.

b. Test 2: For distributed TOEs, Test 1 defined above shall be applicable to all TOE components that forward audit data to an external audit server.

c. Test 3: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall then make note of whether the TSS claims persistent or non-persistent logging and perform one of the following actions:

i. If persistent logging is selected, the evaluator shall perform a power cycle of the TOE and ensure that following power on operations the log events generated are still maintained within the local audit storage.

ii. If non-persistent logging is selected, the evaluator shall perform a power cycle of the TOE and ensure that following power on operations the log events generated are no longer present within the local audit storage.

d. Test 4: 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.1.5. Depending on the configuration this means that the evaluator shall check the content of the audit data when the audit data is just filled to the maximum and then verifies that

i. 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.1.5).

ii. 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.1.5)

iii. The TOE behaves as specified (for the option 'other action' in FAU_STG_EXT.1.5).

e. Test 5: For distributed TOEs, for the local storage according to FAU_STG_EXT.1.4 ,Test 1 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, Test 2 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

f. Test 6 [Conditional]: In case manual export or ability to view locally is selected in FAU_STG_EXT.1.6, during interruption the evaluator shall perform a TSF-mediated action and verify the event is recorded in the audit trail.

Note: The intent of the test is to ensure that the local audit TSF (as specified by FAU_STG_EXT.1.3) operates independently from the ability to transmit the generated audit data to an external audit server (as specified in FAU_STG_EXT.1.1). There are no specific requirements on the interruption of the connection between the TOE and the external audit server (as for FTP_ITC.1). (TD0886 applied)

Test 1: The successful establishment of the IPsec syslog connection for both TOE devices was demonstrated in FTP_ITC.1. In each case, the TOE initiated the connection without administrator intervention. The use of IPsec ensured that no audits were viewed in cleartext. The audits collected as part of FAU_GEN.1 throughout testing were gathered from the remote syslog server running rsyslog version 8.16.0, thus demonstrating that audits were successfully received by the remote syslog server.

Test 2: The option "overwrite previous audit records" is selected in FAU_STG_EXT.1.3. The logs are stored in a local buffer which has a configurable size. The evaluator issued the "show logging" command and viewed the contents of the TOE's log buffer which was at full capacity. The evaluator then performed other commands to generate audit activity and issued the "show logging" command again. The evaluator observed that the new audit records were generated and verified that the oldest audit records had been overwritten.

Test 3: Not applicable. The TOE does not claim FAU_STG_EXT.1/LocSpace.

Test 4: Not applicable. The TOE is not distributed.

Test 5: Not applicable. The TOE is not distributed.

Test 6: The option “ability to view locally” is selected for FAU_STG_EXT.1.6. The evaluator issued the “show logging” command to display the local audit, before disabling the remote syslog logging. The evaluator logged out and logged in and issued the “show logging” command again, displaying the logging in and out event, locally.

2.2 Cryptographic support (FCS)

2.2.1 Cryptographic Key Generation - per TD0923 (NDcPP30e:FCS_CKM.1)

2.2.1.1 NDcPP30e:FCS_CKM.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.1 (FCS_CKM.1) of the ST identifies ECDSA and RSA schemes that meet FIPS PUB 186-4, ECC schemes that meet FIPS PUB 186-4 for asymmetric key generation, and EC-DH schemes that meet NIST SP 800-56A Revision 3 for key establishment. The TOE uses ECDSA keys of size 384 bits in support of device authentication for SSH remote administration. The TOE uses RSA keys of size 3072 bits in support of transmission of generated audit data to an external IT entity.

Component Guidance Assurance Activities: 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.

Section "FIPS Mode" of the Admin Guide provides instructions for configuring the TOE for FIPS mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for generating an EC 384 key for SSH, configuring the SSH server key exchange algorithm (ecdh-sha2-nistp384), configuring the host key and user public key algorithms and configuring SSH rekey settings.

Section "Generating a Crypto Key Pair for IPsec" of the Admin Guide provides instructions for generating an RSA 3072 bit key for use with IPsec.

Section "Create Trustpoints for IPsec" of the Admin Guide provides instructions for mapping the RSA key generated for IPsec to a configured trustpoint and for generating a CSR.

Section "IKEv2" of the Admin Guide provides instructions for configuring DH group 20 in the ikev2 proposal and also for defining an ikev2 policy and profile. The evaluated configuration allows authentication of the IPsec peer using X.509v3 certificates.

Component Testing Assurance Activities: 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 or FIPS PUB 186-5 RSA Schemes

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. This test must be repeated for each supported RSA modulo and generation method.

Key Pair generation specifies 5 ways (or methods) to generate the primes p and q. These include:

a) Random Provable Primes (p and q shall be provable primes).

b) Random Probable primes (p and q shall be probable primes).

c) Provable Primes with Conditions (p1, p2, q1, q2, p and q shall all be provable primes)

d) Provable/probable primes with Conditions (p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes)

e) Probable primes with Conditions (p1, p2, q1, q2, p and q shall all be probable primes)

The Random Provable primes, and all the Primes with Conditions can be tested in the same manner because each of these begin with a starting random number and calculate the p and q values from this value. The test instructs the TSF to generate intermediate values and the p, q, n, and d values. The evaluator then validates the correctness of the values generated by the TSF.

To test the key generation method for the Random Provable primes method or 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. If the TSF provides the input to the key generation function, such input must be recorded and verified. For each RSA key length (modulo) claimed, the evaluator shall generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing Key Pair values generated by the TSF with those generated using the same set of input values using a known good implementation.

The Random Probable primes must be tested in a different way because this test generates different random numbers, not related to each other, until the number satisfies the “probably prime” requirements. This validation method requires two tests for Random Probable primes. These include the Known Answer Test and the Miller-Rabin probabilistic primality test.

To test the key generation method for the Random Probable primes, the known answer test must be used. The evaluator shall compare the results of a known good implementation with the TSF results for a set (of a corresponding size depending on modulus) of supplied values containing private prime factor p, private prime factor q and confirm all public keys, e, match. Then the evaluator shall use the TSF to generate prime p, q pairs for each modulus size and perform the Miller-Rabin tests.

(TD0921 applied)

Key Generation for Elliptic Curve Cryptography (ECC)

Key Generation for ECDSA Schemes

Key pairs for the ECDSA consist of pairs (d, Q), where the private key, d, is an integer, and the public key, Q, is an elliptic curve point.

The evaluator shall verify the implementation of ECC Key Generation by the TOE using the following components tests:

- ECC Key Pair Generation

- ECC Public Key Validation (PKV)

ECC Key Pair Generation Test

There are four methods by which these pairs may be generated:

-FIPS 186-4 Section B.4.1 Key Pair Generation Using Extra Random Bits

Using this method, 64 more bits are requested from the RBG than are needed for d so that bias produced by the mod function is negligible.

-FIPS 186-4 Section B.4.2 Key Pair Generation by Testing Candidates

Using this method, a random number is obtained and tested to determine that it will produce a value of d in the correct range. If d is out-of-range, another random number is obtained (i.e., the process is iterated until an acceptable value of d is obtained.

-FIPS 186-5 Section A.2.1 ECDSA Key Pair Generation using Extra Random Bits

Using this method, more bits are requested from the DRBG than are needed for d so that the bias produced by the mod function is negligible.

-FIPS 186-5 Section A2.2 ECDSA Key Pair Generation by Rejection Sampling

Using this method, a random number is obtained and tested to determine that it will produce a value of d in the correct range. If d is out of range, an ERROR is returned.

The evaluator shall test the ECC Key Pair Generation by having the TSF produce 10 key pairs (d, Q) for each implemented key generation method using each supported curve (i.e., P-256, P-384, and P-521). The steps are the same for each key generation method and curve. The private key, d, shall be generated using the output of an approved DRBG converted to an integer via modular reduction or the discard method. The known private key is then used by the TSF to compute the public key, Q'. To evaluator then validates the correctness by comparing the value Q' computed by the TSF to the public key, Q generated by a known good implementation.

(TD0921 applied)

ECC Public Key Verification (PKV) Test

The evaluator shall generate 12 key pairs (d, Q) for each selected curve, with 6 valid public keys, Q, using a known good implementation and 6 modified, Q', and determine whether the TSF can accurately detect these modifications. Q' should be otherwise valid but include at least one of the following errors: a) point X or Y not on the curve, b) point X or Y is too large for the field for the given curve. The evaluator encouraged to make sure that the modification does not inadvertently result in another valid public key (e.g., modifying Y and accidentally hitting a different point on the curve).

(TD0921 applied)

Key Generation for FIPS PUB 186-5

FIPS 186-5 Key Generation Test

For the Ed25519 curve, 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 186-5 Key Verification Test

For the Ed25519 curve, 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)

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.

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

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process

- Generator g constructed through an unverifiable process.

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.

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

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.

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

for each FFC parameter set and key pair.

FFC Schemes using 'safe-prime' groups

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.2 Cryptographic Key Establishment (NDcPP30e:FCS_CKM.2)

2.2.2.1 NDcPP30e:FCS_CKM.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

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 as shown in the table below. 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.

```
Scheme          |          SFR          |        Service

-------------------------------------------------------------------------------------------

RSA                |  FCS_TLSS_EXT.1  | Administration

-------------------------------------------------------------------------------------------
```

```
ECDH            |  FCS_IPSEC_EXT.1 | Authentication Server

--------------------------------------------------------------------------------------------
```

Section 6 (FCS_CKM.2) of the ST contains multiple tables mapping the key establishment schemes to each service, which can be summarized as follows.

Scheme	Standard	SFR	Service
EC-DH	NIST SP 800-56A Revision 3	FCS_IPSEC_EXT.1	Transmit generated audit data to an external IT entity
		FCS_SSHS_EXT.1	SSH Remote Administration

Component Guidance Assurance Activities: The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

See FCS_CKM.1 where this activity has been performed.

Component Testing Assurance Activities: Key Establishment Schemes

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

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests for ECC and FIPS186-type. 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

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.

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.

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.

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.

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

Validity Test

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.

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 shall 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).

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

RSA-based key establishment

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.

FFC Schemes using 'safe-prime' groups

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.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.3 Cryptographic Key Destruction (NDcPP30e:FCS_CKM.4)

2.2.3.1 NDcPP30e:FCS_CKM.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine 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 shall confirm 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 for2). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator shall check that this is consistent with the operation of the TOE.

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).

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.

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.

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.

Where the ST specifies the use of 'a value that does not contain any CSP' to overwrite keys, the evaluator shall examine 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.

Section 6 (FCS_CKM.4) of the ST states that the TOE meets all requirements specified in FIPS 140-2 for destruction of keys and Critical Security Parameters (CSPs) when no longer required for use. Section 6.1 (Key Zeroization) of the ST provides a table that identifies all keys, provides a description of each key including where it is stored, and identifies how it is cleared. The ST does not identify any configuration needed to ensure keys are cleared.

Component Guidance Assurance Activities: 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.

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

Section "Zeroize Private Key" of the Admin Guide provides instruction for zeroing private keys stored in NVRAM that are generated for SSH or IPsec using the 'crypto key zeroize' command. Other keys stored in SDRAM are zeroized when no longer in use, zeroized with a new value of the key, or zeroized on power-cycle. The Admin Guide does not identify any configurations or circumstances that do not strictly conform to the key destruction requirements or any situation where key destruction may be delayed at the physical layer.

Component Testing Assurance Activities: None Defined

2.2.4 Cryptographic Operation (AES Data Encryption/Decryption) (NDcPP30e:FCS_COP.1/DataEncryption)

2.2.4.1 NDcPP30e:FCS_COP.1.1/DataEncryption

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FCS_COP.1/DataEncryption) of the ST states the TOE provides symmetric encryption and decryption capabilities using AES in CBC mode and GCM mode using 256 bit keys as described in ISO/IEC 18033-3, ISO/IEC 10116, and ISO/IEC 19772. AES is implemented in the SSH and IPsec protocols.

Component Guidance Assurance Activities: 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.

Section "FIPS Mode" of the Admin Guide provides instructions for configuring the TOE for FIPS mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported encryption algorithms used in SSH: aes256-gcm@openssh.com and aes256-cbc.

Section "IKEv2" of the Admin Guide provides instructions for configuring the supported encryption algorithm (aes-gcm-256) in the ikev2 proposal.

Section "IPsec Transform Sets and SA Lifetimes" of the Admin Guide provides instructions for configuring the IPsec ESP encryption algorithm: esp-gcm 256.

Component Testing Assurance Activities: AES-CBC Known Answer Tests

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.

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.

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.

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.

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.

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].

To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of keys 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.

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].

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

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.

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

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]
```

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.

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

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.

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.

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.

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

The Counter (CTR) mode is a confidentiality mode that features the application of the forward cipher to a set of input blocks, called counters, to produce a sequence of output blocks that are exclusive-ORed with the plaintext to produce the ciphertext, and vice versa. Due to the fact that Counter Mode does not specify the counter that is used, it is not possible to implement an automated test for this mode. The generation and management of the counter is tested if the TSF is validated against the requirements of the Functional Package for Secure Shell referenced in Section 2.2 of the cPP. 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):

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.

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.

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.

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].

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

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

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]

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.

There is no need to test the decryption engine.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.5 Cryptographic Operation (Hash Algorithm) (NDcPP30e:FCS_COP.1/Hash)

2.2.5.1 NDcPP30e:FCS_COP.1.1/Hash

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FCS_COP.1/Hash) in the ST states that the TOE provides cryptographic hashing services using SHA-256, and SHA-512 as specified in ISO/IEC 10118-3:2004 (with key sizes and message digest sizes of 256, and 512 bits respectively).

SHA-512 is used within SSH and IPsec key exchange. SHA-512 is also used for verification of software image integrity.

The TOE provides keyed-hashing message authentication services using HMAC-SHA-256 that operates on 512-bit blocks with key size and message digest size of 256 bits bits as specified in ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”.

Section 6 (FCS_SSHS_EXT.1) in the ST indicates that SSH key exchange implementation supports the following algorithms:

- hmac-sha2-256
- ecdsa-sha2-nistp384 (for public key and key exchange)

Section 6 (FPT_APW_EXT.1) in the ST states that all passwords are stored using a SHA-2 hash.

Component Guidance Assurance Activities: The evaluator checks the AGD documents to determine that any configuration that is required to configure the required hash sizes is present.

Section "FIPS Mode" of the Admin Guide provides instructions for configuring the TOE for FIPS Mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported key exchange algorithm: ecdh-sha2-nistp384, the supported MAC algorithms: hmac-sha2-256, the supported host key algorithms: ecdsa-sha2-nistp384, and the supported user/client public key algorithm: ecdsa-sha2-nistp384.

Section "IKEv2" of the Admin Guide provides instructions for configuring the IPsec supported PRF algorithm: prf sha384.

Component Testing Assurance Activities: 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.

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

The evaluator shall 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 evaluator shall 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

The evaluator shall 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

The evaluator shall 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

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

This test is for byte-oriented implementations only. The evaluator shall 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 evaluator shall then ensure that the correct result is produced when the messages are provided to the TSF.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.6 Cryptographic Operation (Keyed Hash Algorithm) (NDcPP30e:FCS_COP.1/KeyedHash)

2.2.6.1 NDcPP30e:FCS_COP.1.1/KeyedHash

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FCS_COP.1/KeyedHash) of the ST states the TOE provides keyed-hashing message authentication services using HMAC-SHA-256 that operates on 512-bit blocks of data, with key size and message digest size of 256 bits as specified in ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

Section 6 (FCS_SSHS_EXT1) of the ST states that the TSF's SSH transport implementation supports the following MAC algorithms: hmac-sha2-256.

Component Guidance Assurance Activities: 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.

Section "FIPS Mode" of the Admin Guide provides instructions for configuring the TOE for FIPS Mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported MAC algorithms: hmac-sha2-256.

Component Testing Assurance Activities: 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.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.7 Cryptographic Operation (Signature Generation and Verification) - per TD0923 (NDcPP30e:FCS_COP.1/SigGen)

2.2.7.1 NDcPP30e:FCS_COP.1.1/SigGen

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6 (FCS_COP.1/SigGen) of the ST states the TOE provides cryptographic signature services using RSA Digital Signature Algorithm with key size of 3072 and ECDSA Digital Signature algorithm with curve P-384 as specified in FIPS PUB 186-4.

Component Guidance Assurance Activities: 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.

Section "FIPS" of the Admin Guide provides instructions for configuring the TOE for FIPS Mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for generating an EC 384 key for use with SSH remote administration.

Section "Generating a Crypto Key Pair for IPsec" of the Admin Guide provides instructions for generating an RSA 3072 bit key for use with IPsec.

Section "Create Trustpoints for IPsec" of the Admin Guide provides instructions for mapping the RSA key generated for IPsec to a configured trustpoint, importing the CA certificate and generating a certificate signing request.

Component Testing Assurance Activities: ECDSA Signature Algorithm Tests

The evaluator shall verify the implementation of ECDSA Signature generation by the TOE using the Signature Generation Test. This test validates the TSF generation of the (r, s) pair that represent the digital signature. The digital signature shall be verified using the same domain parameters and hash function that were used during signature generation. An approved hash function or an XOF shall be used for this purpose.

Signature Generation Test

The purpose of this test is to verify the ability of the TSF to produce correct signatures.

To test signature generation, the evaluator supplies 10 pseudorandom messages to the TSF and a key pair, (d, Q), generated by a known good implementation. Exercising each applicable curve (i.e., P-256, P 384, or P-521) and hash algorithm or extendable-output function combination, the TSF generates signatures for each supplied message and returns the corresponding signatures. Using a known-good implementation, the evaluator validates the signatures by using the associated public key, Q, to verify the signature.

Signature Verification Test

The purpose of this test is to verify the ability of the TSF to accept valid signatures and reject invalid signatures.

For each curve/hash algorithm or extendable-output function combination supported by the TSF, the evaluator shall use a known good implementation to generate a key pair, (d, Q), and use known good implementation with the private key, d, to sign 15 pseudorandom messages of 1024 bits. The evaluator shall alter some of the messages or signatures so that signature verification should fail.

To test signature verification, the evaluator supplies the public key, Q, and 15 pseudorandom messages, including altered messaged, to the TSF for verification of signatures. The evaluator shall then verify that the TSF validates correct signatures on the original messages and flags or rejects the altered messages.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature generation by the TOE using the Signature Generation Test. This test verifies the ability of the TSF to produce correct signatures.

There are 2 different RSA Signature algorithms that can be implemented. These include:

a. RSASSA-PKCS1-v1.5

b. RSASSA-PSS

To test signature generation, the evaluator generates or obtains 10 messages for each modulus size/hash or extendable-output function combination supported by the TOE. Using a key generated by a known good implementation, the TSF generates and returns the corresponding signatures to a known good implementation that validates the signatures by using the associated public key to verify the signature.

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

The evaluator shall verify the implementation of RSA Signature verification by the TOE using the Signature Verification Test. This test verifies the ability of the TSF to recognize valid and invalid signatures.

There are 2 different RSA Signature algorithms that can be implemented. These include:

a. RSASSA-PKCS1-v1.5

b. RSASSA-PSS

For each modulus size/hash or extendable-output function combination supported by the TOE, the evaluator shall use a known good implementation to generate a modulus and three associated key pairs, (d, e). Each private key d is used to sign six pseudorandom messages each of 1024 bits. Some of the public keys, e, messages, IR format, or signatures must be altered so that signature verification should fail. The modifications must cover distinct “key modified”, “message modified”, “signature modified”, “IR moved”, and “trailer moved” tests. The modulus, hash or extendable-output algorithm, public key e values, messages, and signatures are forwarded to the TSF. The TSF then attempts to verify the signatures and returns the results. The evaluator then compares the received results with the stored results from a known good implementation.

The evaluator shall verify that the TSF validates correct signatures on the original messages and flags or rejects the altered messages.

(TD0921 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

2.2.8 IPsec Protocol - per TD0868 (NDcPP30e:FCS_IPSEC_EXT.1)

2.2.8.1 NDcPP30e:FCS_IPSEC_EXT.1.1

TSS Assurance Activities: 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.

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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that when a packet matches a permit entry in a particular access list, and the corresponding crypto map entry is tagged connections are established, if necessary. If the crypto map entry is tagged as ipsec-isakmp, IPsec is triggered. If there is no SA that the IPsec can use to protect this traffic to the peer, IPsec uses IKE to negotiate with the remote peer to set up the necessary IPsec SAs on behalf of the data flow. The negotiation uses information specified in the crypto map entry as well as the data flow information from the specific access list entry.

Once established, the set of SAs (outbound to peer) is then applied to the triggering packet and to subsequent applicable packets as those packets exit the Switch. "Applicable" packets are packets that match the same access list criteria that the original packet matched. For example, all applicable packets could be encrypted before being forwarded to the remote peer. The corresponding inbound SAs are used when processing the incoming traffic from that peer.

Access lists associated with IPsec crypto map entries also represent the traffic that the Switch needs protected by IPsec. Inbound traffic is processed against crypto map entries. If an unprotected packet matches a permit entry in a particular access list associated with an IPsec crypto map entry, that packet is dropped because it was not sent as an IPsec-protected packet. 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 in the crypto map, but that is not disallowed by other ACLs on the interface is allowed to BYPASS the tunnel. Traffic that does not match a permit ACL and is also blocked by other non-crypto ACLs on the interface would be DISCARDED. Rules applied to an access list can be applied to either inbound or outbound traffic.

Guidance Assurance Activities: 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.

Section "IPsec Crypto Map and Access Control List" of the Admin Guide provides instructions for configuring access lists and applying these access lists to interfaces using crypto map sets. 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 and packets will be encrypted/decrypted (Protect). This section also provides examples of how to configure and apply the access lists such that specific traffic will be dropped (Discard) or will flow through unencrypted (Bypass).

Testing Assurance Activities: The evaluator shall use the guidance documentation to configure the TOE to perform 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 shall perform 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 shall observe 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.

b. Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. As with Test 1, the evaluator shall ensure 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.

Test 1: The TOE uses access-lists in order to control the flow of traffic. If traffic is destined for an IP address associated with the IPsec tunnel, then the traffic will be protected. Otherwise, the access-lists are screened in order to determine whether the traffic is allowed through unprotected or dropped. The evaluator sent a series of packets to the TOE. The evaluator sent packets to the IPsec tunnel interface that matched the rules to be encrypted (Protect) and observed that the traffic was encrypted by IPsec. The evaluator then sent the packets that did not match IPsec traffic selectors but which matched permit rules on the access list applied to the interface and observed that the traffic was sent in plaintext (Bypass). The evaluator then sent packets that matched rules to be denied and observed that the packets were dropped (Discard). The evaluator also sent packets that did not match any of the specific rules and observed that they were denied by default.

Test 2: To test the priority behavior of the SPD rules, the evaluator configured the access list on the interface to discard packets on the subnet first followed by a bypass rule for the specific host. The evaluator sent packets from the defined host and viewed that they were denied as that rule took priority. The evaluator then flipped the priorities of the access list on the interface so the specific bypass rule was first. The evaluator sent packets from the defined host and viewed that they were accepted in plaintext as that rule took priority.

2.2.8.2 NDcPP30e:FCS_IPSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

The evaluator shall use the guidance documentation to configure the TOE to perform 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 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 shall 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 create'' final entry that discards packets that do not match any previous entries). The evaluator shall send the packet and observes that the packet was dropped.

This test was performed as part of NDcPP30e:FCS_IPSEC_EXT.1.1-t1, which demonstrates the TOE protecting a packet, discarding a packet, allowing the packet to bypass the tunnel, and dropping the packet if a packet field is modified and does not match any rule.

2.2.8.3 NDcPP30e:FCS_IPSEC_EXT.1.3

TSS Assurance Activities: The evaluator shall check 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).

Section 6 (FCS_IPSEC_EXT.1) of the ST states the TOE supports both transport tunnel mode for IPsec communications between the TOE and an external audit server.

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

Section "IPsec Transform Sets and SA Lifetimes" of the Admin Guide provides instructions on how to configure the connection in transport or tunnel mode.

Testing Assurance Activities: The evaluator shall perform the following test(s) based on the selections chosen:

a. Test 1: If tunnel mode is selected, the evaluator shall use 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 shall configure 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 shall observe (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

b. Test 2: If transport mode is selected, the evaluator shall use 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 shall configure 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 shall then initiate a connection from the TOE to connect to the VPN peer. The evaluator shall observe (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

Test 1 and Test 2: The TOE supports both tunnel and transport mode. The evaluator configured a test peer to require only tunnel mode or only transport mode. In each case, the evaluator then attempted to establish an IPsec connection between the test peer and the TOE and observed via logs and packet captures that the connection was successful in each case.

2.2.8.4 NDcPP30e:FCS_IPSEC_EXT.1.4

TSS Assurance Activities: The evaluator shall examine the TSS to verify that the algorithms are implemented. In addition, the evaluator shall ensure that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1(4)/KeyedHash Cryptographic Operations (for keyed-hash message authentication) and if the SHA-based HMAC function truncated output is utilized it must also be described.

Section 6 (FCS_IPSEC_EXT.1) of the ST states The TSF’s implementation of the IPsec standard (in accordance with the RFCs noted in the SFR) uses the Internet Key Exchange version 2 (IKEv2) protocol and the Encapsulating Security Payload (ESP) protocol to provide authentication and encryption supporting the following algorithms:

- AES-GCM-256

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Section " IPsec Transform Set and Profile " of the Admin Guide provides instructions for configuring the supported encryption algorithm (aes-gcm-256) for ESP.

Testing Assurance Activities: 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.

The evaluator configured the TOE to use AES-GCM-256 and verified via logs and packet captures that the connection was successfully established for each algorithm.

2.2.8.5 NDcPP30e:FCS_IPSEC_EXT.1.5

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

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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states the TOE supports IKEv2 session establishment. This is consistent with the requirement with only claims IKEv2.

Guidance Assurance Activities: 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 for the following test (if selected).

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.

Section "IKEv2" of the Admin Guide provides instructions on how to configure the IPsec with IKEv2 functionality for the TOE. The TOE does not support NAT traversal.

Testing Assurance Activities: 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 shall then show that main mode exchanges are supported.

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 7296, Section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

Test 1: Not applicable. IKEv1 is not selected.

Test 2: Not applicable. NAT traversal is not selected within the IKEv2 selection.

2.2.8.6 NDcPP30e:FCS_IPSEC_EXT.1.6

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that the TSF's implementation of the IPsec standard (in accordance with the RFCs noted in the SFR) uses the Internet Key Exchange version 2 (IKEv2) protocol and the Encapsulating Security Payload (ESP) protocol to provide authentication and encryption supporting AES-GCM-256.

Guidance Assurance Activities: The evaluator shall ensure that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Section "IKEv2" of the Admin Guide provides instructions for configuring the encryption algorithm, aes-gcm-256, in the IKEv2 proposal.

Testing Assurance Activities: 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 shall confirm the algorithm was that used in the negotiation.

The evaluator configured IKEv2 profiles (AES-GCM-256) on the TOE. The evaluator then confirmed that the TOE could establish a session with each algorithm and that the tunnel successfully established with the selected algorithm.

2.2.8.7 NDcPP30e:FCS_IPSEC_EXT.1.7

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that the TOE supports configuration of session lifetimes for both IKEv2 SAs and IKEv2 Child SAs using the command "lifetime". The time values for IKEv2 SAs can be configured between 2 minutes and 24 hours. The time values for IKEv2 Child SAs can be configured between 2 minutes and 720 hours. The IKEv2 Child SA lifetimes can also be configured by an Administrator based on number of bytes.

Guidance Assurance Activities: 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.

Section "IKEv2" of the Admin Guide provides instructions for configuring the lifetime in seconds of the IKEv2 profile. This includes the ability to configure the values from 120 to 86400 seconds, which is equivalent to the values (2 minutes to 24 hours) specified in the requirement.

Testing Assurance Activities: 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.'

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 a 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.

b. 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 Phase1 SA lifetime that exceeds the Phase 1SA lifetime on the TOE. The evaluator shall establish a SA between the TOE and the test peer, maintain the Phase 1 SA for 24 hours, and determine that a new Phase 1 SA is negotiated on or before 24 hours has elapsed. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.

Test 1: Not applicable. The TOE does not support data size based rekey limits for the IKEv2 SA.

Test 2: The TOE’s Phase 1 and phase 2 lifetimes are configurable.As they are configurable, the evaluator configured the TOE to have a 1 hour IKE and 30 minute ESP limits. The evaluator established an IPsec connection between the TOE and the test server and then waited for over 1 hour before terminating the test. The evaluator observed through logs and packet captures that the connection was successful and that the TOE rekeyed well before the configured time limits were reached, ensuring that the keys were renegotiated successfully and there was no data loss during the rekey. This ensured that the TOE could rekey before the maximum 24 hour Phase 1 and 8 hour Phase 2 requirements.

2.2.8.8 NDcPP30e:FCS_IPSEC_EXT.1.8

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that the TOE supports configuration of session lifetimes for both IKEv2 SAs and IKEv2 Child SAs using the command "lifetime". The time values for IKEv2 SAs can be configured between 2 minutes and 24 hours. The time values for IKEv2 Child SAs can be configured between 2 minutes and 720 hours. The IKEv2 Child SA lifetimes can also be configured by an Administrator based on number of bytes.

Guidance Assurance Activities: 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 8 hours. 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.

Section "IPsec Transform Sets and SA Lifetimes" of the Admin Guide provides instructions for configuring the IPsec security association lifetime, which can be configured based on time in seconds. This includes the ability to configure the values from 120 to 2592000 seconds, which is equivalent to the values 2 minutes to 720 hours as specified in the requirement. Instructions are also included for configuring the IPsec security association lifetime by volume in kilobytes with a range of 2560 to 4294967295.

Testing Assurance Activities: When testing this functionality, the evaluator shall 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.'

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 a 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.

b. 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 lifetime of the Phase 2 SA lifetime on the TOE. The evaluator shall establish a SA between the TOE and the test peer, maintain the Phase 1 SA for 8 hours, and determine that once a new Phase 2 SA is negotiated when or before 8 hours has elapsed. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

Test 1: The evaluator configured a max lifetime of 2560 KB on the TOE and established a connection with a test peer. The Phase 2 SA timed out after the packet size was exceeded and the TOE rekeyed the negotiation successfully.

Test 2: This test was performed as part of the FCS_IPSEC_EXT.1.7 test 2 where the evaluator observed that the TOE rekeyed before the configured time limits for the Phase 1 SA and the Phase 2 SA.

2.2.8.9 NDcPP30e:FCS_IPSEC_EXT.1.9

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states the TSF generates the secret value 'x' used in the IKEv2 Diffie-Hellman key exchange ('x' in g^x mod p) using the NIST approved DRBG specified in FCS_RBG_EXT.1 and having possible lengths of 384 bits. The TOE generates nonces used in IKEv2 exchanges, of at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash. When a random number is needed for a nonce, 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 likewise generated using the HMAC DRBG.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.8.10 NDcPP30e:FCS_IPSEC_EXT.1.10

TSS Assurance Activities: 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.

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.

Section 6.1 (FCS_IPSEC_EXT.1) of the ST states the TSF generates the secret value 'x' used in the IKEv2 Diffie-Hellman key exchange ('x' in g^x mod p) using the NIST approved DRBG specified in FCS_RBG_EXT.1 and having possible lengths of 384 bits. The TOE generates nonces used in IKEv2 exchanges, of at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash. When a random number is needed for a nonce, 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 likewise generated using the HMAC DRBG.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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.

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.

See TSS assurance activity above.

2.2.8.11 NDcPP30e:FCS_IPSEC_EXT.1.11

TSS Assurance Activities: 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 shall check to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Section 6 (FCS_IPSEC_EXT.1) of the ST states the TOE supports Diffie-Hellman Group 20.

Guidance Assurance Activities: The evaluator shall ensure that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Section "IKEv2" of the Admin Guide provides information on how to select DH Group 20) for IKE.

Testing Assurance Activities: 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.

The evaluator made a successful IPsec connection to an IPsec peer using the claimed DH group 20. The evaluator was able to capture DH group 20 using a packet capture to ensure the correct DH group was used.

2.2.8.12 NDcPP30e:FCS_IPSEC_EXT.1.12

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that the resulting potential strength of the symmetric key will be 256 bits of security. As part of this negotiation, the TOE verifies that the negotiated IKE Child SA symmetric algorithm key strength is at most as large as the negotiated IKE SA key strength as configured on the TOE and peer via an explicit check.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall follow 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.

b. Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish a 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.

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.

d. Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish a 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.

Test 1: The evaluator made an IPsec connection to an IPsec peer using each of the claimed hash functions identified in the requirements. The evaluator verified via packet capture and logs that the connection was successful with each of the claimed functions.

Test 2: This test is met by guidance in the Admin Guide which instructs the user not to configure an ESP encryption algorithm with greater strength than the algorithm configured for the IKEv2 SA. The evaluator configured the TOE as instructed and then attempted to establish a connection with a test server configured with an ESP algorithm with greater strength than the IKEv2 SA algorithm and observed that the attempt failed.

Test 3: The evaluator attempted to establish a connection with an unsupported algorithm/hash combination. The connection attempt failed.

Test 4: This test was performed as part of Test 3 above.

2.2.8.13 NDcPP30e:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: The evaluator shall ensure 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(2)/SigGen Cryptographic Operations (for cryptographic signature).

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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states that the TOE supports authentication of IPsec peers using RSA X.509 certificates. This is consistent with the algorithms as specified in FCS_COP.1/SigGen.

Pre-shared keys are not selected.

Guidance Assurance Activities: The evaluator shall ensure the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

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.

The evaluator shall 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'.

Section "Create Trustpoints for IPsec" of the Admin Guide provides instructions on how to set up the TOE to use certificates with RSA signatures and public keys. This section also explains 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'.

Section "IKEv2" of the Admin Guide explains how to configure IPsec to use X.509v3 certificates for authentication of IPsec peers.

Pre-shared keys are not selected.

Testing Assurance Activities: For efficiency sake, the testing is combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

This testing is combined and performed as part of testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1 where IPsec connections were successfully established using RSA certificates.

2.2.8.14 NDcPP30e:FCS_IPSEC_EXT.1.14

TSS Assurance Activities: 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.

Section 6 (FCS_IPSEC_EXT.1) of the ST states for peer authentication using RSA X.509 certificates, the TOE validates the presented identifier provided supporting the following fields and types: SAN: IP address, SAN: Fully Qualified Domain Name (FQDN).

Certificate maps provide the ability for a certificate to be matched with a given set of criteria. The Administrator is instructed in the CC Configuration Guide to specify one or more certificate fields together with their matching criteria and the value to match. In the evaluated configuration, the field name must speficy the SAN (alt-subject-name) field. Match criteria should be "eq" for equal. SAN example: alt-subject-name eq <peer.cisco.com>.

The TOE will reject the IKE connection in any of these situations: 1) If the data ID Payload for any of those ID Types does not match the peer's certificate exactly; 2) If an ID Payload is not provided by the peer; 3) If multiple ID Types are provided in the ID Payload.

Guidance Assurance Activities: 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.

Section "Configure Reference Identifier" of the Admin Guide provides instructions for configuration of the peer reference identifier through use of a certificate map. 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. The certificate map is associated with the IPsec trustpoint. This section further states that in the evaluated configuration, the field name must specify the SAN field of the peer's certificate. This includes specifying either the FQDN of the peer or the IP address of the peer in the SAN.

Testing Assurance Activities: In 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.

The evaluator shall perform the following tests:

a. 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.

b. 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.

c. Test 3: (conditional) For each CN/identifier type combination selected, the evaluator shall:

i. Create a valid certificate with the CN so it contains the valid identifier followed by ''. 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.

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

d. Test 4: (conditional) For each SAN/identifier type combination selected, the evaluator shall:

i. 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.

ii. 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.

e. 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.

f. 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:

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

ii. Append '' to a non-CN field of an otherwise authorized DN.

Test 1: Not applicable. The TOE does not support CN identifiers.

Test 2: For the first part of this test, the evaluator alternately configured a test peer to use an authentication certificate with the correct SAN: IP address and SAN: FQDN. The evaluator then attempted to establish an IPsec connection between the TOE and the test peer and confirmed that the connection was successful. CN identifiers are not supported by the TOE so the second part of the test is not applicable.

Test 3: Not applicable. The TOE does not support CN identifier types.

Test 4: For this test, the evaluator alternately configured the TOE to look for each of the supported SAN reference identifiers (SAN: IP address and SAN: FQDN). The evaluator then configured the test peer to use a certificate that would present an incorrect SAN reference identifier and a correct CN reference identifier as CN checking is not prioritized over SAN (CN is not supported). In each case, the evaluator attempted to establish an IPsec connection to the TOE and then expected the TOE to reject the connection.

Test 5: Not applicable. DN identifier types are not supported.

Test 6: Not applicable. The TOE does not support DN and CN identifier types.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.9 Random Bit Generation (NDcPP30e:FCS_RBG_EXT.1)

2.2.9.1 NDcPP30e:FCS_RBG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.9.2 NDcPP30e:FCS_RBG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: Documentation shall be produced - and the evaluator shall perform the activities - in accordance with Appendix D of [NDcPP].

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.

The Entropy description is provided in a separate (non-ST) document that has been delivered to NIAP for approval. Note that the entropy analysis has been accepted by NIAP/NSA.

Section 6 (FCS_RBG_EXT.1) of the ST states The TOE implements a NIST-approved CTR-DRBG, as specified in ISO/IEC 18031:2011 seeded by an entropy source that accumulates entropy from a software-based noise source.

The DRBG 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.

Component Guidance Assurance Activities: Documentation shall be produced - and the evaluator shall perform the activities - in accordance with Appendix D of [NDcPP].

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

The Entropy description is provided in a separate (non-ST) document that has been delivered to NIAP for approval. Note that the entropy analysis has been accepted by NIAP/NSA.

Section "FIPS Mode" of the Admin Guide provides instructions for configuring the TOE for FIPS mode of operation. FIPS Mode of operation must be enabled in the evaluated configuration and allows the TOE to use only approved cryptography, including the proper DRBG methods. Otherwise, there is no further configuration required for configuring RNG functionality.

Component Testing Assurance Activities: The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration.

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 shall verify 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).

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 shall verify 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.

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.

The TOE has been CAVP tested. Refer to the section entitled "CAVP Certificates" earlier in this document.

2.2.10 SSH Protocol - per TD0909 (SSH10:FCS_SSH_EXT.1)

2.2.10.1 SSH10:FCS_SSH_EXT.1.1

TSS Assurance Activities: The evaluator shall ensure that the selections indicated in the ST are consistent with selections in this and subsequent components. Otherwise, this SFR is evaluated by activities for other SFRs

This SFR is evaluated by activities for other SFRs

Guidance Assurance Activities: There are no guidance evaluation activities for this component. This SFR is evaluated by activities for other SFRs.

There are no guidance evaluation activities for this component.

Testing Assurance Activities: There are no test evaluation activities for this component. This SFR is evaluated by activities for other SFRs

There are no test evaluation activities for this component.

2.2.10.2 SSH10:FCS_SSH_EXT.1.2

TSS Assurance Activities: The evaluator shall check to ensure that the authentication methods listed in the TSS are identical to those listed in this SFR component; and, ensure if password-based authentication methods have been selected in the ST then these are also described; and, ensure that if keyboard-interactive is selected, it describes the multifactor authentication mechanisms provided by the TOE.

Section 6 (FCS_SSH_EXT.1) of the ST states the TOE supports password-based authentication and ecdsa-sha2-nistp384for client public key authentication. When the SSH client presents a public key, the TSF verifies it matches the one configured for the Administrator account. If the presented public key does not match the one configured for the Administrator account, access is denied.

Keyboard-interactive is not selected.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure the configuration options, if any, for authentication mechanisms provided by the TOE are described.

Section "SSH Remote Administration Protocol" of the Admin Guide gives instructions how to configure password and publickey authentication for SSH usingecdh-sha2-nistp384.

Testing Assurance Activities: Test 1: [conditional] If the TOE is acting as SSH Server:

a. The evaluator shall use a suitable SSH Client to connect to the TOE, enable debug messages in the SSH Client, and examine the debug messages to determine that only the configured authentication methods for the TOE were offered by the server.

b. [conditional] If the SSH server supports X509 based Client authentication options:

a. The evaluator shall initiate an SSH session from a client where the username is associated with the X509 certificate. The evaluator shall verify the session is successfully established.

b. Next the evaluator shall use the same X509 certificate as above but include a username not associated with the certificate. The evaluator shall verify that the session does not establish.

c. Finally, the evaluator shall use the correct username (from step a above) but use a different X509 certificate which is not associated with the username. The evaluator shall verify that the session does not establish.

Test 2: [conditional] If the TOE is acting as SSH Client, the evaluator shall test for a successful configuration setting of each authentication method as follows:

a. The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established.

b. Next, the evaluator shall use bad authentication data (e.g. incorrectly generated certificate or incorrect password) and ensure that the connection is rejected.

Steps a-b shall be repeated for each independently configurable authentication method supported by the server.

Test 3: [conditional] If the TOE is acting as SSH Client, the evaluator shall verify that the connection fails upon configuration mismatch as follows:

a. The evaluator shall configure the Client with an authentication method not supported by the Server.

b. The evaluator shall verify that the connection fails.

If the Client supports only one authentication method, the evaluator can test this failure of connection by configuring the Server with an authentication method not supported by the Client. In order to facilitate this test, it is acceptable for the evaluator to configure an authentication method that is outside of the selections in the SFR.

Test 1: The TOE supports only ecdh-sha2-nistp384 for client public key authentication. The evaluator enabled debug messages on the TOE. Then, generated an ECDSA P384 key pair on the test server and then configured an admin user on the TOE with the public key. The evaluator then attempted to login to the TOE using this rsa-sha2-256 public key and observed that the login was successful and confirmed the audit logs.

Test 2: Not applicable. TOE is not acting as SSH client.

Test 3: Not applicable. TOE is not acting as SSH client.

2.2.10.3 SSH10:FCS_SSH_EXT.1.3

TSS Assurance Activities: The evaluator shall check that the TSS describes how 'large packets' are detected and handled.

Section 6 (FCS_SSH_EXT.1) of the ST states that SSHv2 connections will be dropped if the TOE receives a packet larger than 65,806 bytes. Large packets are detected by the SSHv2 implementation and dropped internal to the SSH process.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: Test 1: The evaluator shall demonstrate that the TOE accepts the maximum allowed packet size.

Test 2: This test is performed to verify that the TOE drops packets that are larger than size specified in the component.

a. The evaluator shall establish a successful SSH connection with the peer.

b. Next the evaluator shall craft a packet that is slightly larger than the maximum size specified in this component and send it through the established SSH connection to the TOE. The packet should not be greater than the maximum packet size + 16 bytes. If the packet is larger, the evaluator shall justify the need to send a larger packet.

c. The evaluator shall verify that the packet was dropped by the TOE. The method of verification will vary by the TOE. Examples_include reviewing the TOE audit log for a dropped packet audit or observing the TOE terminates the connection.

(TD0732 applied)

Test 1: The evaluator created and send a pack to the TOE of the maximum allowed packet size. The TOE accepted the packet and the connection was successful.

Test 2: The evaluator created and sent a packet to the TOE that was larger than the maximum packet size. The TOE rejected the packet and the connection was closed.

2.2.10.4 SSH10:FCS_SSH_EXT.1.4

TSS Assurance Activities: The evaluator will check the description of the implementation of SSH in the TSS to ensure the encryption algorithms supported are specified. The evaluator will check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

Section 6 (FCS_SSH_EXT.1) of the ST states that the TSF's SSH transport implementation supports the following encryption algorithms: aes256-cbc and aes256-gcm@openssh.com. This is consistent with the algorithms specified in the requirement.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported encryption algorithms used in SSH: aes256-gcm@openssh.com, aes256-cbc.

Testing Assurance Activities: The evaluator shall perform the following tests.

If the TOE can be both a client and a server, these tests must be performed for both roles.

Test 1: The evaluator must ensure that only claimed algorithms and cryptographic primitives are used to establish an SSH connection. To verify this, the evaluator shall establish an SSH connection with a remote endpoint. 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 only the algorithms defined in the ST for the TOE for SSH connections. The evaluator shall perform one successful negotiation of an SSH connection and verify that the negotiated algorithms were included in the advertised set. If the evaluator detects that not all algorithms defined in the ST for SSH are advertised by the TOE or the TOE advertises additional algorithms not defined in the ST for SSH, the test shall be regarded as failed.

The data collected from the connection above shall be used for verification of the advertised hashing and shared secret establishment algorithms in FCS_SSH_EXT.1.5 and FCS_SSH_EXT.1.6 respectively.

Test 2: For the connection established in Test 1, the evaluator shall terminate the connection and observe that the TOE terminates the connection.

Test 3: The evaluator shall configure the remote endpoint to only allow a mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the attempt fails.

Test 1: The evaluator established an SSH connection with the TOE. Then collected the extracted the data collected to verify that the appropriate mechanisms were advertised. The evaluator used the packet capture of the connection to collect a list of SSH encryption algorithms, hash algorithms, key exchange algorithms, and host key algorithms that were offered by the TOE.

Test 2: Tested in test 1. The evaluator established an SSH connection with the TOE then issued the exit command at the end of the connection attempted. The TOE terminated the connection.

Test 3: The evaluator attempted to connect to the TOE using a disallowed cipher, 3des-cbc, and observed that the connection failed.

2.2.10.5 SSH10:FCS_SSH_EXT.1.5

TSS Assurance Activities: The evaluator will check the description of the implementation of SSH in the TSS to ensure the hashing algorithms supported are specified. The evaluator will check the TSS to ensure that the hashing algorithms specified are identical to those listed for this component.

Section 6.1 (FCS_SSH_EXT.1) of the ST states that the TSF's SSH transport implementation supports hmac-sha2-256. When aes256-gcm@openssh.com is used as the encryption algorithm the MAC algorithm is implicit.

This is consistent with the algorithms specified in the requirement.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported MAC algorithms: hmac-sha2-256 . When (and only when) the encryption algorithm selected is aes256-gcm@openssh.com supplying the MAC algorithm is skipped. When aes256-gcm@openssh.com is used as the encryption algorithm, the MAC algorithm is implicit.

Testing Assurance Activities: Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.

Test 2: The evaluator shall configure an SSH peer to allow only a hashing algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.

Test 1: The evaluator established an SSH connection with the TOE. Then collected the extracted the data collected to verify that the appropriate mechanisms were adversited.

Test 2: The evaluator attempted to establish an SSH connection with the TOE using the HMAC-MD5 algorithm. The connection attempt failed.

2.2.10.6 SSH10:FCS_SSH_EXT.1.6

TSS Assurance Activities: The evaluator will check the description of the implementation of SSH in the TSS to ensure the shared secret establishment algorithms supported are specified. The evaluator will check the TSS to ensure that the shared secret establishment algorithms specified are identical to those listed for this component.

Section 6 (FCS_SSH_EXT.1) of the ST states that the TSF's SSH key exchange implementation supports the following key exchange algorithm: ecdh-sha2-nistp384 (RFC 5656). This is consistent with the requirement.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the supported key exchange algorithm: ecdh-sha2-nistp384 (RFC 5656),.

Testing Assurance Activities: Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.

Test 2: The evaluator shall configure an SSH peer to allow only a key exchange method that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.

Test 1: The evaluator performed this as part of FCS_SSH_EXT.1.4_t1 where the evaluator verified that the appropriate Key Exchange algorithms were advertised by the TOE

Test 2: The evaluator attempted to establish an SSH connection with the TOE using a key exchange method that is not included in the ST: diffie-hellman-group1-sha1. The connection failed.

2.2.10.7 SSH10:FCS_SSH_EXT.1.7

TSS Assurance Activities: The evaluator will check the description of the implementation of SSH in the TSS to ensure the KDFs supported are specified. The evaluator will check the TSS to ensure that the KDFs specified are identical to those listed for this component

Section 6 (FCS_SSH_EXT.1) of ST states the TOE derives cryptographic session keys via shared secret using SSH KDF as defined in RFC 5656 (Section 4).This aligns with the selections made in the SFR.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.10.8 SSH10:FCS_SSH_EXT.1.8

TSS Assurance Activities: The evaluator shall check the TSS to ensure that if the TOE enforces connection rekey or termination limits lower than the maximum values that these lower limits are identified.

In cases where hardware limitation will prevent reaching data transfer threshold in less than one hour, the evaluator shall check the TSS to ensure it contains:

a. An argument describing this hardware-based limitation and

b. Identification of the hardware components that form the basis of such argument.

For example, if specific Ethernet Controller or Wi-Fi radio chip is the root cause of such limitation, these subsystems shall be identified.

Section 6 (FCS_SSHS_EXT.1) of the ST states the TSF's SSH implementation will perform a rekey after no longer than one hour or more than one gigabyte of data has been transmitted with the same session key. Both thresholds are checked. Rekeying is performed upon reaching whichever threshold is met first. The Administrator can configure lower rekey values if desired. The minimum time value is 10 minutes. The minimum volume value is 100 kilobytes.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that if the connection rekey or termination limits are configurable, it contains instructions to the administrator on how to configure the relevant connection rekey or termination limits for the TOE.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring time-based and volume-based rekey values. SSH connections with the same session keys cannot be used longer than one hour, and with no more than one gigabyte of transmitted data. Values can be configured to be lower if desired. The minimum time value is 10 minutes. The minimum volume value is 100 kilobytes.

Testing Assurance Activities: The test harness needs to be configured so that its connection rekey or termination limits are greater than the limits supported by the TOE -- it is expected that the test harness should not be initiating the connection rekey or termination.

Test 1: Establish an SSH connection. Wait until the identified connection rekey limit is met. Observed that a connection rekey or termination is initiated. This may require traffic to periodically be sent, or connection keep alive to be set, to ensure that the connection is not closed due to an idle timeout.

Test 2: Establish an SSH connection. Transmit data from the TOE until the identified connection rekey or termination limit is met. Observe that a connection rekey or termination is initiated.

Test 3: Establish an SSH connection. Send data to the TOE until the identified connection rekey limit or termination is met. Observe that a connection rekey or termination is initiated.

Test 1: The evaluator configured the rekey time of 10 minutes. The evaluator attempted to connect to the TOE using an SSH client and confirmed that a rekey happened when the configured threshold was reached.

Test 2: The evaluator configured the rekey data limit to 100 KB. The evaluator then repeatedly issued a command on the TOE's CLI that outputs a large amount of text and confirmed the rekey happened at the 100KB threshold.

Test 3: The evaluator configured the rekey data limit to 100 KB. The evaluator attempted to connect to the TOE using a SSH client sending data via an SCP copy and confirmed that a rekey happened at the 100KB threshold.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.2.11 SSH Protocol - Server - per TD0682 (SSH10:FCS_SSHS_EXT.1)

2.2.11.1 SSH10:FCS_SSHS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

Section "SSH Remote Administration Protocol" of the Admin Guide gives instructions for configuring authentication of itself to its peer (SSH Client) using: ecdsa-sha2-nistp384.

Component Testing Assurance Activities: The evaluator shall perform the following tests:

Test 1: The evaluator shall use a suitable SSH Client to connect to the TOE and examine the list of server host key algorithms in the SSH_MSG_KEXINIT packet sent from the server to the client to determine that only the configured server authentication methods for the TOE were offered by the server.

Test 2: The evaluator shall test for a successful configuration setting of each server authentication method as follows. The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established. Repeat this process for each independently configurable server authentication method supported by the server.

Test 3: The evaluator shall configure the peer to only allow an authentication mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the TOE sends a disconnect message.

(TD0682 applied)

Test 1: This test has been performed in FCS_SSH_EXT.1.4-t1 where the evaluator verified that the correct list of server host key algorithms was offered by the TOE.

Test 2: The evaluator connected to the TOE using an SSH client alternately using the claimed server authentication method. The evaluator confirmed that the supported authentication methods resulted in successful connections.

Test 3: The evaluator attempted to establish an SSH connection with the TOE using an authentication algorithm method that is not included in the ST: ssh-dss. The connection failed.

2.3 Identification and Authentication (FIA)

2.3.1 Authentication Failure Management (NDcPP30e:FIA_AFL.1)

2.3.1.1 NDcPP30e:FIA_AFL.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.1.2 NDcPP30e:FIA_AFL.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

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).

Section 6 (FIA_AFL.1) of the ST states the TOE provides the Administrator the ability to specify the maximum number of unsuccessful authentication attempts before an offending account will be blocked. The TOE also provides the ability to specify the time period to block offending accounts.

To avoid a potential situation where password failures made by Administrators leads to no Administrator access until the defined blocking time period has elapsed, the CC Configuration Guide instructs the Administrator to configure the TOE for SSH public key authentication which is not subjected to password-based brute force attacks. During the block out period, the TOE provides the ability for the Administrator account to login remotely using SSH public key authentication.

To block password-based brute force attacks, the TOE uses an internal AAA function to detect and track failed login attempts. When an account attempting to log into an administrative interface reaches the set maximum number of failed authentication attempts, the account will not be granted access until the time period has elapsed or until the Administrator manually unblocks the account.

Component Guidance Assurance Activities: 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.

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.

Section "Configure Authentication Failure" of the Admin Guide provides instructions for specifying the value for the maximum number of failed login attempts and the time period to lock the offending account.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the switch for SSH public key authentication. This is necessary to avoid a potential situation where password failures by remote Administrators lead to no Administrator access until the account is manually unlocked. During the defined lockout period, the TOE provides the ability for the Administrator account to login remotely using SSH public key authentication.

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

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

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

If the administrator action selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

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

Test 1 & 2: The evaluator first set the maximum number of invalid logins to 3 with the command "aaa authentication rejected 3 in 60 ban 60". The evaluator then attempted to login to the user TestUser with an invalid password 3 times. The evaluator then attempted to login to the user TestUser before the 60 second time period with a valid password, which did not result in successful access. The evaluator then attempted to login to the user TestUser after the 60 second time period with a valid password, resulting in successful access.

Next, the evaluator next set the maximum number of invalid logins to 5 with the command "aaa authentication rejected 5 in 120 ban 120". The evaluator then attempted to login to the user TestUser with an invalid password 5 times. The evaluator then attempted to login to the user TestUser before the 120 second time period with a valid password, which did not result in successful access. The evaluator then attempted to login to the user TestUser after the 120 second time period with a valid password, resulting in successful access.

Finally, the evaluator then reset the maximum number of invalid logins to 3 with the command "aaa authentication rejected 3 in 600 ban 600". The evaluator then attempted to login to the user TestUser with an invalid password 3 times. The evaluator then attempted to login to the user TestUser before the 600 second time period with a valid password, which did not result in successful access. The evaluator then logged in as the admin account and demonstrated that the TestUser account was displayed as blocked. The evaluator then unlocked the TestUser account with the "clear aaa local user blocked username TestUser" command. The TestUser account no longer was displayed as blocked. The evaluator then attempted to login to the user TestUser with a valid password before the 600 second time period, resulting in successful access.

2.3.2 Password Management (NDcPP30e:FIA_PMG_EXT.1)

2.3.2.1 NDcPP30e:FIA_PMG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall check that the TSS:

a. lists the supported special character(s) for the composition of administrator passwords.

b. to ensure that the minimum_password_length parameter is configurable by a Security Administrator.

c. lists the range of values supported for the minimum_password_length parameter. The listed range shall include the value of 15.

Section 6 (FIA_PMG_EXT.1) of the ST states the TOE supports the local definition of users with corresponding passwords. The passwords can be composed of any combination of upper and lower case letters, numbers, and special characters that include: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")" and other special characters listed in the table below. Minimum password length is settable by the Authorized Administrator, and can be configured for minimum password lengths of 1 and maximum of 127 characters. A minimum password length of 8 is recommended.

Special Character	Name
	Space
;	Semicolon
:	Colon
“	Double Quote
‘	Single Quote
|	Vertical Bar
+	Plus
-	Minus
=	Equal Sign

.	Period
,	Comma
/	Slash
\	Backslash
<	Less Than
>	Greater Than
_	Underscore
`	Grave Accent (backtick)
~	Tilde
{	Left Brace
}	Right Brace

Component Guidance Assurance Activities: 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.

Section "Define Password Policy" of the Admin Guide provides instructions for defining a common criteria policy that can be applied to each local account and that will ensure that passwords contain a minimum of 1 character.

Section “Add Administrator Account” provides guidance to security administrators on the composition of strong passwords:

- The password should not contain the associated username and the username should not be reversed.
- The characters in the password should not be repeated more than three times consecutively.
- The password should not be cisco, ocsic, admin, nimda, or any variant obtained by changing the capitalization of letters therein, or by substituting "1" "|" or "!" for i, and/or substituting "0" for "o", and/or substituting "$" for "s".
- The password should be composed of uppercase letters, lowercase letters, digits, and the special characters listed in table 4. The password length should be at least 8.

Component Testing Assurance Activities: 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.

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.

Test 1 & 2: The evaluator configured a minimum password length of 8 characters, configured a user with a valid 8 character password, and attempted the following password changes:

1. Attempted to set a password that was 7 characters (change rejected by TOE),
2. Attempted to set a password that was 128 characters (change rejected by TOE),
3. Attempted to set a password demonstrating all claimed special characters (change accepted by TOE),
4. Attempted to set a password demonstrating all valid numerical characters (change accepted by TOE),
5. Attempted to set a password demonstrating all valid uppercase characters (change accepted by TOE), and
6. Attempted to set a password demonstrating all valid lowercase characters (change accepted by TOE).

2.3.3 Protected Authentication Feedback (NDcPP30e:FIA_UAU.7)

2.3.3.1 NDcPP30e:FIA_UAU.7.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: 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.

There are no preparatory steps needed to ensure authentication data is not revealed. Section 6 (FIA_UAU.7) of the ST states that when a user enters their password at the local console, the TOE does not echo any characters as the password is entered. For remote session authentication, the TOE does not echo any characters as they are entered.

Component Testing Assurance Activities: 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.

Test 1: This test was performed as part of the tests for NDcPP30e:FIA_UIA_EXT.1 where the evaluator observed that passwords are obscured on the console logins.

2.3.4 User Identification and Authentication - per TD0900 (NDcPP30e:FIA_UIA_EXT.1)

2.3.4.1 NDcPP30e:FIA_UIA_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.4.2 NDcPP30e:FIA_UIA_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.4.3 NDcPP30e:FIA_UIA_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.4.4 NDcPP30e:FIA_UIA_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that it describes the logon process for remote authentication mechanism (e.g. SSH public key, Web GUI password, etc.) and optional local authentication mechanisms supported by the TOE. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a 'successful logon'.

The evaluator shall examine the TSS to determine that it describes which actions are allowed before administrator identification and authentication. The description shall cover authentication and identification for local and remote TOE administration.

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, 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.

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

Section 6 (FIA_UIA_EXT.1) of the ST states the TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed. Prior to being granted access, a login warning banner is displayed.

Administrative access to the TOE is facilitated through a local password-based authentication mechanism and remote SSH password and public key authentication mechanisms on the TOE through which all Administrator actions are mediated. Once a potential (unauthenticated) administrative user attempts to access the TOE through an interactive administrative interface, the TOE prompts the user for a user name and password or SSH public key authentication. The TOE then either grants administrative access (if credentials are valid, and the account has not been locked) or indicates the login attempt was unsuccessful. The TOE does not provide a reason for failure in the cases of a login failure. A successful login is indicated by a hash sign ("#") next to the device hostname. No access is allowed to the administrative functionality of the TOE until the administrator is successfully identified and authenticated.

The TOE is not distributed.

Component Guidance Assurance Activities: 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.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for configuring the remote SSH interface. The details of the configuration, preparatory steps, and claimed functionality are described in the assurance activities for the SSH protocol. As identified in the FIA requirements, the password and account lockout mechanisms are described in the guidance. Section "Access CLI Over SSH" describes how to login via SSH.

Section "Preparative Procedures and Operational Guidance for the TOE" provides guidance for managing the device locally.

Component Testing Assurance Activities: 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 all combinations of supported credentials and login methods, 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.

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.

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.

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

The TOE offers the following user interfaces where authentication is provided:

- Local Console using local authentication with password
- SSH CLI using local authentication with password
- SSH CLI using local authentication with public key

Test 1: Using each interface, the evaluator performed an unsuccessful and successful login of each type using bad and good credentials, respectively.

Test 2: Using each interface, the evaluator was able to observe the TOE displayed a banner to the user before login and that there were no other services available nor any configuration options offered to administrators to control services available prior to authentication.

Test 3: This test was performed as part of Test 1. Using each interface, the evaluator found that, prior to login, no functions were available to the administrator with the exception of acknowledging the banner.

Test 4: Not applicable. The TOE is not a distributed TOE.

2.3.5 X.509 Certificate Validation (NDcPP30e:FIA_X509_EXT.1/Rev)

2.3.5.1 NDcPP30e:FIA_X509_EXT.1.1/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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 expected that either OCSP or CRL revocation checking is performed when a certificate is presented to the TOE (e.g. during authentication). 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).

b. 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.

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

d. 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 shall then attempt 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.

e. Test 4a: [conditional] If OCSP is selected, the evaluator shall configure an authorized responder or use a man-in-the-middle tool to use a delegated OCSP signing authority to respond to the TOE's OCSP request. The resulting positive OCSP response (certStatus: good (0)) shall be signed by an otherwise valid and trusted certificate with the extendedKeyUsage extension that does not contain the OCSPSigning (OID 1.3.6.1.5.5.7.3.9). The evaluator shall verify that the TSF does not successfully complete the revocation check.

Note: Per RFC 6960 Section 4.2.2.2, the OCSP signature authority is delegated when the CA who issued the certificate in question is NOT used to sign OCSP responses.

f. Test 4b: [conditional] If CRL is selected, the evaluator shall present an otherwise valid CRL signed by a trusted certificate that does not have the cRLsign key usage bit set and verify that validation of the CRL fails.

g. 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.)

h. Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC 5280 Section 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.)

i. 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.)

j. Test 8 (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The following tests are run when a minimum certificate path length of three certificates is implemented:

k. 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.

l. 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.

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

Test 1: The evaluator alternately configured the TOE to have and then not have the trusted root CA used by Strongswan on the test peer to anchor all of its certificates. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE, expecting the connection to succeed only when the trusted root CA was properly configured, forming a valid certificate chain.

Test 2: The evaluator alternately configured Strongswan on a test peer to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect to the IPsec VPN between the test peer and the TOE, expecting the connection to succeed only if there are no expired certificates.

Test 3: The evaluator alternately configured Strongswan on a test peer to send an authentication certificate 1) that is valid, 2) that is revoked, and 3) issued by an intermediate CA that is revoked. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE, expecting the connection to succeed only if there are no revoked certificates.

Test 4: The evaluator alternately configured Strongswan on a test peer to send an authentication certificate 1) that is valid, 2) issued by an intermediate CA referring to a CRL revocation server where the signer lacks cRLSign, and 3) issued by an intermediate CA whose issuer CA refers to a CRL revocation server where the signer lacks cRLSign. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE, expecting the connection to succeed only if all retreived CRLs are signed using certificates with cRLSign.

Test 5: For this test, the evaluator alternately configured Strongswan on a test peer to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE, expecting the connection to succeed only if the certificate is not modified/corrupted.

Test 6: This test has been performed in FIA_X509_EXT.1.1/Rev-t5.

Test 7: This test has been performed in FIA_X509_EXT.1.1/Rev-t5.

Test 8: Not applicable as while the TOE claims ECDSA support for FCS_COP.1/SigGen, that is only relevant to SSH. ECDSA certs are not supported for IPsec as noted in FCS_IPSEC_EXT.1.13.

2.3.5.2 NDcPP30e:FIA_X509_EXT.1.2/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: 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.

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). 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 shall confirm 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).

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 shall confirm 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).

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).

Test 1: For this test, the evaluator alternately configured a test peer to send an authentication certificate issued by a Sub CA with no basicConstraints and with basicConstraints but the CA Flag set to false. In each case, the evaluator attempted to connect the IPsec VPN between the test peer and the TOE and observed that the connection was rejected in each case.

Test 2: This was performed as part of Test 1.

Component TSS Assurance Activities: 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).

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.

Section 6 (FIA_X509_EXT.1/Rev) of the ST states the TOE uses X.509v3 certificates to support authentication for IPsec connections. The TSF determines the validity of certificates at the time of authentication by ensuring that the certificate and the certificate path are valid in accordance with RFC 5280. The certificate path is validated by ensuring that all the CA certificates have the basicConstraints extension and the CA flag is set to TRUE and the certificate path must terminate with a trusted CA certificate. CRL revocation checking is supported by the TOE. Revocation checking is performed on the leaf and intermediate certificate(s) when authenticating a certificate chain provided by the remote peer. There are no functional differences if a full certificate chain or only a leaf certificate is presented.

Component Guidance Assurance Activities: 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.

Section "IPsec" of the Admin Guide states that the TOE uses X.509v3 certificates to support authentication for IPsec connections. The TSF determines the validity of certificates by ensuring that the certificate and the certificate path are valid in accordance with RFC 5280. The certificate path is validated by ensuring that all the CA certificates have the basicConstraints extension and the CA flag is set to TRUE and the certificate path must terminate with a trusted CA certificate. OCSP is not supported; therefore the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) is trivially satisfied by the TOE. Revocation checking is performed on the leaf and intermediate certificate(s) when authenticating a certificate chain provided by the remote peer.

Component Testing Assurance Activities: None Defined

2.3.6 X.509 Certificate Authentication (NDcPP30e:FIA_X509_EXT.2)

2.3.6.1 NDcPP30e:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.6.2 NDcPP30e:FIA_X509_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

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.

Section 6 (FIA_X509_EXT.2) of the ST states the TOE determines which certificate to use based upon the trustpoint configured. The instructions for configuring trustpoints is provided in CC Configuration Guide. In the event that a network connection cannot be established to verify the revocation status of certificate for an external peer, the certificate will be rejected and the connection will not be established.

Component Guidance Assurance Activities: 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.

Section “IPsec” in the Admin Guide provides instructions for how to establish the trusted channel to the audit server including generating a crypto key pair for IPsec, creating trustpoints, configuring the IKEv2 profile, configuring IPsec transform sets, configuring SA lifetimes, configuring crypto maps, access control lists and the reference identifier.

Section “Create Trustpoints for IPsec” in the Admin Guide states that IPsec must be configured to use X.509v3 certificates supporting a minimum path length of three (root CA -> intermediate CA -> end-entity). This section provides the steps for creating a root CA and a subordinate CA using certificates from the administrator’s organization's PKI. Before proceeding, the administrator is instructed to have the root CA certificate and subordinate CA ready for import from their CA administrator.   Steps are then provided to create the root CA and subordinate CA trustpoints and install the associated certificates, generate a CSR and then import the signed certificate.

Section "Create Trustpoints for IPsec" in the Admin Guide further states that if the TOE is unable to obtain a CRL, the TOE will reject the peer's certificate and a "CRL fetch for trustpoint <trustpoint name> failed" message will appear in the message log.   The Administrator will need to enable the remote syslog server as described in section "Enable Remote Syslog Server", once the revocation server is back online.

Component Testing Assurance Activities: The evaluator shall perform the following test for each trusted channel:

a. Test 1: 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.

The evaluator alternately configured a test peer to send an authentication certificate with valid/accessible revocation servers and an authentication certificate with revocation information referring to an inaccessible revocation server. In each case, the evaluator then attempted to make a connection between the test peer and the TOE, expecting the connection to be successful when the revocation server is accessible and when the revocation server is not accessible, only if that behavior is claimed for the TOE. The evaluator observed the certificate validation checking behavior in each case and confirmed that it was consistent with the actions selected in FIA_X509_EXT.2.2 in the ST.

2.3.7 X.509 Certificate Requests (NDcPP30e:FIA_X509_EXT.3)

2.3.7.1 NDcPP30e:FIA_X509_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.7.2 NDcPP30e:FIA_X509_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

The ST does not select 'device-specific information'.

Component Guidance Assurance Activities: The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certification 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.

Section "Create Trustpoints for IPsec" of the Admin Guide provides instructions for creating the root CA and subordinate CA trustpoints, which includes establishing the subject-name to at least the Common Name, Organization, Organizational Unit, and Country. The certificate signing request is then generated from the subordinate CA trustpoint and includes the Common Name (CN), Organization (O), Organizational Unit (OU), and Country (C) as specified in the requirement.

Component Testing Assurance Activities: 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 request 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.

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 response message, and demonstrate that the function succeeds.

Test 1: The evaluator followed optional guidance to log into the TOE and then generated the CSR/key pair. The evaluator captured the generated request and ensured that it contains the information claimed in the ST.

Test 2: The evaluator attempted to import a certificate without a valid certification path and the import failed. The evaluator then attempted to import a certificate with a valid certification path and the import succeeded.

2.4 Security management (FMT)

2.4.1 Management of security functions behaviour (NDcPP30e:FMT_MOF.1/ManualUpdate)

2.4.1.1 NDcPP30e:FMT_MOF.1.1/ManualUpdate

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For distributed TOEs see Section 2.4.1.1. There are no specific requirements for non-distributed TOEs.

The TOE is not distributed.

Component Guidance Assurance Activities: 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).

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).

Section "Update TOE Software" of the Admin Guide provides instructions on how to perform a manual update of the TOE. It explains the image on the Cisco website and as part of the install, the signature will be validated. If the signature is not correct, the device will not boot.

The TOE is not distributed.

Component Testing Assurance Activities: The evaluator shall perform the following tests:

a. Test 1: 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 should fail.

b. Test 2: 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.

Test 1: As can be seen in the FIA_UIA_EXT.1 test evidence, no functions are offered to users prior to a successful login. Any user that can login is considered an administrator and can perform TOE updates.

Test 2: FPT_TUD_EXT.1 demonstrates the successful updating of the TOE by a trusted administrator.

2.4.2 Management of TSF Data (NDcPP30e:FMT_MTD.1/CoreData)

2.4.2.1 NDcPP30e:FMT_MTD.1.1/CoreData

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For each administrative function identified in the guidance documentation that is accessible through an interface prior to administrator log-in are identified, the evaluator shall confirm that the TSS details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.

If the 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.

Section 6 (FMT_MTD.1/CoreData) of the ST states 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. Only Security Administrators can access the TOE's trust store. Each of the predefined and administratively configured roles has create (set), query, modify, or delete access to the TOE data, though with some privilege levels, the access is limited.

The TOE performs role-based authorization, using TOE platform authorization mechanisms, to grant access to the privileged and semi-privileged roles. For the purposes of this evaluation, the privileged level is equivalent to full administrative access to the CLI, which is the default access for IOS-XE privilege level 15; and the semi-privileged level 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 customizable.

Component Guidance Assurance Activities: 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.

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.

Section "Create Trustpoints for IPsec" of the Admin Guide provides instructions on how to use X.509 certificates with the TOE. It explains how to request a certificate, how to communicate with a certificate authority, and how to load a certificate.

Guidance for all other management functions have been identified under the guidance activity for the SFR which they apply.

Component Testing Assurance Activities: 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.

No additional testing was required as all management functions were demonstrated throughout the course of testing other SFRs.

2.4.3 Management of TSF Data (NDcPP30e:FMT_MTD.1/CryptoKeys)

2.4.3.1 NDcPP30e:FMT_MTD.1.1/CryptoKeys

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For distributed TOEs see Section 2.4.1.1.

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 names the operations that are performed.

Section 6 (FMT_MTD.1/CryptoKeys) of the ST states that only Security Administrators can access the TOE's trust store. This section describes that the Authorized Administrator generates RSA key pairs to be used in the IPsec protocol and ECC key pairs to be used in the SSH protocol. This section also describes that the TOE Administrators can control (generate/delete) the following keys, RSA Key Pairs and SSH EC Key Pairs by following the instruction in the AGD.

The TOE is not distributed.

Component Guidance Assurance Activities: For distributed TOEs see Section 2.4.1.2.

For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the operations are performed on the keys the Security Administrator is able to manage.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions for generating an EC 384 key for SSH, configuring the SSH server key exchange algorithm (ecdh-sha2-nistp384), configuring the host key and user public key algorithms, and configuring SSH rekey settings.

Section "Generating a Crypto Key Pair for IPsec" of the Admin Guide provides instructions for generating an RSA 3072 bit key for use with IPsec.

Section "Create Trustpoints for IPsec" of the Admin Guide includes instructions for mapping RSA keys generated for IPsec to a configured trustpoint, importing the CA certificate and generating a certificate signing request.

The TOE is not distributed.

Component Testing Assurance Activities: The evaluator shall perform the following tests:

a. Test 1: 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.

b. Test 2: The evaluator shall try to perform at least one of the related actions with prior authentication as Security Administrator. This attempt should be successful.

Test 1: As can be seen in the FIA_UIA_EXT.1 test evidence, no functions are offered to users prior to a successful login. Any user that can login is considered an administrator and can perform TOE updates.

Test 2: FIA_X509_EXT.3 test 1 demonstrates the successful generating of a CSR/key pair by an authorized administrator.

2.4.4 Specification of Management Functions - per TD0880 (NDcPP30e:FMT_SMF.1)

2.4.4.1 NDcPP30e:FMT_SMF.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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(1)/ManualUpdate, FMT_MOF.1(4)/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1(2)/Services, and FMT_MOF.1(3)/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.

(containing also requirements on Guidance Documentation and Tests)

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).

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.

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.

(If configure local audit is selected) The evaluator shall examine the TSS and Guidance Documentation to ensure that a description of the logging implementation is described in enough detail to determine how log files are maintained on the TOE.

Section 6 (FMT_SMF.1) of the ST provides a description of the management functions of the TOE. The TOE provides all capabilities necessary to securely manage the TOE and the services provided by the TOE. The management functionality of the TOE is provided through the TOE CLI. The Authorized Administrator can perform all management functions by accessing the TOE directly via connected console cable or remote administration via SSHv2 secure connection. The specific management capabilities available from the TOE include:

- Ability to administer the TOE remotely;
- Ability to configure the access banner;
- Ability to configure the remote session inactivity time before session termination;
- Ability to update the TOE, and to verify the updates using a digital signature capability prior to installing those updates;
- Ability to configure local audit behavior (e.g. changes to storage locations for audit; changes to behavior when local audit storage space is full, changes to local audit storage size);
- Ability to manage the cryptographic keys;

- Ability to configure the cryptographic functionality;
- Ability to configure thresholds for SSH rekeying;
- Ability to configure lifetime for IPsec SAs;
- Ability to re-enable an Administrator account;
- Ability to set the time which is used for time-stamps;
- Ability to configure the reference identifier for the peer;
- Ability to manage the TOE's trust store and designate X.509v3 certificates as trust anchors;
- Ability to import X.509v3 certificates to the TOE's trust store;
- Ability to generate Certificate Signing Request (CSR) and process CA certificate response;
- Ability to administer the TOE locally;
- Ability to configure the local session inactivity time before session termination or locking,
- Ability to configure the authentication failure parameters for FIA_AFL.1;
- Ability to manage the trusted public keys database;

Guidance for all management functions have been identified under the guidance activity for the SFR which they apply.

Component Guidance Assurance Activities: See TSS Assurance Activities

See TSS Assurance Activities.

Component Testing Assurance Activities: The evaluator shall test 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.

All TOE security functions are identified and have been tested as documented throughout this AAR. A summary is below:

- Ability to administer the TOE remotely – tested numerous times by the administrator using SSH to access the TOE for testing
- Ability to configure the access banner – tested as part of FTA_TAB.1
- Ability to configure the remote session inactivity time before session termination – tested as part of FTA_SSL.3
- Ability to update the TOE, and to verify the updates using a digital signature capability prior to installing those updates – tested as part of FPT_TUD_EXT.1
- Ability to configure local audit behavior (e.g. changes to storage locations for audit; changes to behavior when local audit storage space is full, changes to local audit storage size) – tested as part of FAU_STG_EXT.1.
- Ability to manage the cryptographic keys – tested as part of FCS_SSHS_EXT.1 and FCS_IPSEC_EXT.1

- Ability to configure the cryptographic functionality - tested as part of FCS_SSHS_EXT.1 and FCS_IPSEC_EXT.1
- Ability to configure thresholds for SSH rekeying - tested as part of FCS_SSHS_EXT.1
- Ability to configure lifetime for IPsec SAs - tested as part of FCS_IPSEC_EXT.1
- Ability to re-enable an Administrator account - tested as part of FIA_AFL.1
- Ability to set the time which is used for time-stamps - tested as part of FPT_STM.1
- Ability to configure the reference identifier for the peer - tested as part of FCS_IPSEC_EXT.1
- Ability to manage the TOE's trust store and designate X.509v3 certificates as trust anchors - tested as part of FIA_X509_EXT.1
- Ability to import X.509v3 certificates to the TOE's trust store - tested as part of FIA_X509_EXT.1
- Ability to generate Certificate Signing Request (CSR) and process CA certificate response - tested as part of FIA_X509_EXT.3
- Ability to administer the TOE locally - tested numerous times by the administrator using the console to access the TOE for testing
- Ability to configure the local session inactivity time before session termination or locking - tested as part of FTA_SSL_EXT.1
- Ability to configure the authentication failure parameters for FIA_AFL.1 - tested as part of FIA_AFL.1
- Ability to manage the trusted public keys database - tested as part of FIA_X509_EXT.1

2.4.5 Restrictions on Security Roles (NDcPP30e:FMT_SMR.2)

2.4.5.1 NDcPP30e:FMT_SMR.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.5.2 NDcPP30e:FMT_SMR.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.4.5.3 NDcPP30e:FMT_SMR.2.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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 (e.g. if local administrators and remote administrators have different privileges or if several types of administrators with different privileges are supported by the TOE).

Section 6 (FMT_SMR.2) of the ST states the TOE maintains privileged and semi-privileged Administrator roles. The TOE performs role-based authorization, using TOE platform authorization mechanisms, to grant access to TOE functions. 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-XE privilege level (PL) 15. Semi-privileged roles are assigned a PL of 0-14. PL 0 and 1 are defined by default and are customizable, while PL 2-14 are undefined by default and are also customizable. Note: Levels 0-14 are a subset of PL 15 and the levels are not hierarchical.

The TOE can and shall be configured to authenticate all access to the command line interface using a username and password. The TOE supports both local administration via directly connected console cable and remote administration via SSHv2 secure connection.

Component Guidance Assurance Activities: 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.

See FIA_UIA_EXT.1 which identifies the instructions of the Admin Guide for administering the TOE both locally and remotely.

Component Testing Assurance Activities: 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, if the TSF shall be validated against the Functional Package for Secure Shell referenced in Section 2.2 of the cPP; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

Testing throughout the course of the evaluation was performed using both the SSH and local hardware interfaces.

2.5 Protection of the TSF (FPT)

2.5.1 Protection of Administrator Passwords (NDcPP30e:FPT_APW_EXT.1)

2.5.1.1 NDcPP30e:FPT_APW_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.1.2 NDcPP30e:FPT_APW_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FPT_APW_EXT.1) of the ST states the TOE is designed specifically to not disclose any passwords stored in the TOE. All passwords are stored using a SHA-2 hash. 'Show' commands display only the hashed password.

The CC Configuration Guide instructs the Administrator to use the algorithm-type scrypt subcommand when passwords are created or updated. The scrypt is password type 9 and uses a SHA-2 hash.

Section 6 (FPT_SKP_EXT.1) of the ST states the TOE is designed specifically to not disclose any keys stored in the TOE. The TOE stores all private keys in a secure directory that cannot be viewed or accessed, even by the Administrator. The TOE stores symmetric keys only in volatile memory.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5.2 Protection of TSF Data (for reading of all symmetric keys) (NDcPP30e:FPT_SKP_EXT.1)

2.5.2.1 NDcPP30e:FPT_SKP_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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 any interface designed specifically for that purpose, by any enabled role, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Section 6 (FPT_SKP_EXT.1) of the ST states the TOE is designed specifically to not disclose any keys stored in the TOE. The TOE stores all private keys in a secure directory that cannot be viewed or accessed, even by the Administrator. The TOE stores symmetric keys only in volatile memory.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5.3 Reliable Time Stamps (NDcPP30e:FPT_STM_EXT.1)

2.5.3.1 NDcPP30e:FPT_STM_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.3.2 NDcPP30e:FPT_STM_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

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.

Section 6.1 (FPT_STM_EXT.1) of the ST states the TSF implements a clock function to provide a source of date and time. The clock function is reliant on the system clock provided by the underlying hardware. All Switch models have a real-time clock (RTC) with battery to maintain time across reboots and power loss.

The TOE relies upon date and time information for the following security functions:

- To monitor local and remote interactive administrative sessions for inactivity (FTA_SSL_EXT.1, FTA_SSL.3);
- Validating X.509 certificates to determine if a certificate has expired (FIA_X509_EXT.1/Rev);

- To determine when IKEv2 SA lifetimes have expired and to initiate a rekey (FCS_IPSEC_EXT.1);
- To determine when IPsec Child SA lifetimes have expired and to initiate a rekey (FCS_IPSEC_EXT.1);
- To determine when SSH session keys have expired and to initiate a rekey (FCS_SSHS_EXT.1);
- To provide accurate timestamps in audit records (FAU_GEN.1.2).

Additionally, 'obtain time from underlying virtualization system' is not selected for the TOE.

Component Guidance Assurance Activities: The evaluator shall examine 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.

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.

Section "Configure Time and Date" of the Admin Guide provides information on how to set the time on the TOE.

The TOE does not support the use of an NTP server.

The TOE does not support obtaining time from the underlying VS.

Component Testing Assurance Activities: 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 shall use the guidance documentation to set the time. The evaluator shall then use an available interface to observe that the time was set correctly.

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 shall 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.

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.

If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

Test 1: The evaluator followed the guidance instructions to configure the time on the TOE. The evaluator read the time from the TOE using a date command and also found audit records confirming that the time was successfully changed.

Test 2: Not applicable. NTP is not supported.

Test 3: Not applicable. The TOE does not support obtaining time from the underlying VS.

2.5.4 TSF testing - per TD0836 (NDcPP30e:FPT_TST_EXT.1)

2.5.4.1 NDcPP30e:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.4.2 NDcPP30e:FPT_TST_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to ensure that it details each of the self-tests that are identified by the SFR; 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. If more than one failure response is listed in FPT_TST_EXT.1.2, the evaluator shall examine the TSS to ensure it clarifies which response is associated with which type of failure.

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. The evaluator shall also examine the TSS to ensure it describes how the TOE reacts if one or more TOE components fail self-testing (e.g. halting and displaying an error message; failover behaviour).

(TD0836 applied)

The TOE runs a suite of self-tests during initial start-up to verify correct operation of the cryptographic module. All ports are blocked from moving to forwarding state during the POST. If all components of all modules pass the POST, the system is placed in FIPS PASS state and ports are allowed to forward data traffic. If any of the tests fail, the system halts and a message is displayed to the local console. These tests include:

- 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. If the encrypted texts match, the test passes; otherwise, the test fails. 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. If the decrypted texts match, the test passes; otherwise, the test fails.
- 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. If the encrypted values match, the test passes; otherwise, the test fails. The encrypted data is then decrypted using the private key. This value is compared to the original plaintext value. If the decrypted values match, the test passes; otherwise, the test fails.
- 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. If the random bits match, the test passes; otherwise, the test fails.
- 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. If the MAC values match, the test passes; otherwise, the test fails.
- Software Integrity Test: The Software Integrity Test uses HMAC-SHA256 verification to confirm the cryptographic module has maintained its integrity. The Software Integrity Test is run automatically when the module is loaded.
- SHA-256/384/512 Known Answer Test: For each of the values listed, the SHA implementation is fed known data and a key. These values are used to generate a hash. This hash is compared to a known value. If the hash values match, the test passes; otherwise, the test fails.

If any component reports failure for the POST, the system crashes. Appropriate information is displayed on the screen and saved in the crashinfo file.

All ports are blocked during the POST. If all components pass the POST, the system is placed in FIPS PASS state and ports can forward data traffic.

If an error occurs during the self-test, a SELF_TEST_FAILURE system log is generated.

Example Error Message: "%CRYPTO-0-SELF_TEST_FAILURE: Crypto algorithms self-test failed (SHA hashing)".

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.

At the request of the authorized administrator, the self-tests can be executed on-demand. Refer to the AGD for related instructions.

The TOE is not distributed.

Component Guidance Assurance Activities: 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.

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.

Section "Cryptographic Self-Tests" of the Admin Guide states the TOE runs a suite of self-tests during initial start-up to verify correct operation of cryptographic modules. If any component reports failure for the POST, the system crashes and appropriate information is displayed on the local console. All ports are blocked from moving to forwarding state during the POST. If all components of all modules pass the POST, the system is placed in FIPS PASS state and ports are allowed to forward data traffic. If any of the tests fail, a message is displayed to the local console and the TOE component will automatically reboot. If the Administrator observes a cryptographic self-test failure, they must contact Cisco Technical Support.

Instructions are also provided for the administrator to run an on-demand version of the self-test.

The TOE is not distributed.

Component Testing Assurance Activities: 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 fulfill any of the SFRs.

Although formal compliance is not mandated, the self-tests performed should aim for a level of confidence comparable to:

a. FIPS 140-2, Section 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, Section 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.

The evaluator shall verify that the self tests described above are carried out according to the SFR and in agreement with the descriptions in the TSS.

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.

(TD0836 applied)

During a reboot of the TOE, the evaluator confirmed that the TOE performed self tests to verify the firmware integrity and the cryptographic functions. The evaluator observed the output of these tests indicate that they were successful. The firmware integrity test passed and all other tests were successfully completed with no errors.

2.5.5 Trusted update (NDcPP30e:FPT_TUD_EXT.1)

2.5.5.1 NDcPP30e:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.5.2 NDcPP30e:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.5.3 NDcPP30e:FPT_TUD_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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 shall describe how and when the inactive version becomes active. The evaluator shall verify this description.

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. 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 of the update, and the actions that take place for both successful and unsuccessful signature verification.

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.

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.

Section 6 (FPT_TUD_EXT.1) of the ST states an Authorized Administrator can query the software version running on the TOE and can initiate updates to (replacements of) software images. 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 the updates. The updates can be downloaded from https://software.cisco.com/. Trusted updates can be installed on the TOE in a single stage or as a multi-stage process with a delayed activation. The inactive version will become active when the Administrator responds 'y' at the re-boot prompt. The updates can be downloaded from software.cisco.com.

The TOE will authenticate the image using a digital signature verification check to ensure it has not been modified since distribution using the following process: Prior to being made publicly available, the software image is hashed using a SHA512 algorithm and then digitally signed. The digital signature is embedded to the image (hence the image is signed). The TOE uses a Cisco public key to validate the digital signature to obtain the SHA512 hash. The TOE then computes its own hash of the image using the same SHA512 algorithm and verifies the computed hash against the embedded hash. If they match the image has not been modified or tampered since distributed from Cisco meaning the software is authenticated and the image is ready to be activated automatically in the single stage upgrade or by the administrator in the multistage upgrade. If they do not match the image will not install.

Component Guidance Assurance Activities: 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.

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.

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.

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.

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 shall also ensure that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

Section "Update TOE Software" of the Admin Guide provides instructions for how to download an image on the TOE. The TOE will automatically verify the integrity of the stored image when loaded for execution. The Admin Guide provides detailed instructions for each step in the process. The SWITCH uses a Cisco public key to validate the digital signature to obtain an embedded SHA512 hash that was generated prior to the image being distributed from Cisco. The SWITCH then computes its own hash of the image using the same SHA512 algorithm. The SWITCH verifies the computed hash against the embedded hash. If they match the image is authenticated and has not been modified or tampered and the install will proceed. If they do not match the image will not boot or execute.

The TOE is not distributed.

Component Testing Assurance Activities: 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.

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:

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

ii. An image that has not been signed.

iii. 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)

iv. 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.

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

For distributed TOEs the evaluator shall perform Test 1 and Test 2 for all TOE components.

Test 1: The evaluator verified the current TOE version and then followed guidance procedures to install a new image to the TOE. The evaluator then verified the TOE version again and confirmed that it was updated successfully.

Test 2: For each device, the evaluator used legitimate updates that were modified in three ways:

- Corrupted Image/Valid Signature - A few bytes in the update file are modified via a hex editor
- Valid Image/No Signature - Update is missing a signature
- Valid Image/Invalid Signature - Update's signature is corrupted.

Attempts to update with each of these modified images failed as expected.

Test 3: Not applicable. The TOE does not verify the integrity of updates using published hashes.

2.6 TOE ACCESS (FTA)

2.6.1 TSF-INITIATED TERMINATION (NDcPP30E:FTA_SSL.3)

2.6.1.1 NDcPP30e:FTA_SSL.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6 (FTA_SSL.3) of the ST states the allowable inactivity timeout range is <0-35791> minutes. A value of 0 means there is no inactivity timeout enforced and therefore a value of 0 must not be used in the evaluated configuration.. An Authorized Administrator can configure maximum inactivity times individually for both local and remote administrative sessions using the "exec-timeout" command applied to the console and virtual terminal (vty) lines.

The configuration of the vty lines sets the configuration for the remote console access.

The line console settings are not immediately activated for the current session. The current line console session must be exited. When the user logs back in, the inactivity timer will be activated for the new session. The local interactive session terminates and does not lock. If a local user session is inactive for a configured period, the session will be terminated and will require re-identification and authentication to login. If a remote user session is inactive for a configured period, the session will be terminated and will require re-identification and authentication to establish a new session.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Section " 4.2.17. SSH Remote Administration Protocol" of the Admin Guide, starting at step 13, discusses remote SSH session termination. It provides instructions for setting termination values for inactive SSH sessions. The range is from 0 to 35791 minutes. A value of 0 means there is no inactivity timeout enforced therefore a value of 0 must not be used in the evaluated configuration.

Component Testing Assurance Activities: For each method of remote administration, the evaluator shall perform the following test:

a. Test 1: The evaluator shall follow the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator shall establish a remote interactive session with the TOE. The evaluator shall then observes that the session is terminated after the configured time period.

The evaluator set a session timeout of 1 minute and found that the SSH session disconnected after 1 minute, as expected. The evaluator then repeated the test with a time of 2 minutes and found that the SSH session disconnected after 2 minutes, as expected.

2.6.2 User-initiated Termination (NDcPP30e:FTA_SSL.4)

2.6.2.1 NDcPP30e:FTA_SSL.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that it details how the remote administrative session (and if applicable the local administrative session) are terminated.

Section 6 (FTA_SSL.4) of the ST states an Authorized Administrator can exit out of both local and remote administrative sessions by issuing the 'exit' or 'logout' command.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation states how to terminate a remote interactive session (and if applicable the local administrative session).

Section "SSH Remote Administration Protocol" of the Admin Guide states that to terminate a remote or local session to the switch, use the 'exit' or 'logout' command at the User or Privilege EXEC prompt to terminate the session.

Component Testing Assurance Activities: The evaluator shall perform the following tests:

a. Test 1: If the TOE supports local administration, the evaluator shall initiate an interactive local session with the TOE. The evaluator shall then follow the guidance documentation to exit or log off the session and observes that the session has been terminated.

b. Test 2: For each method of remote administration, the evaluator shall initiate an interactive remote session with the TOE. The evaluator shall then follow the guidance documentation to exit or log off the session and observes that the session has been terminated.

Test 1: The evaluator logged in to the local console and then typed in the command "exit". The evaluator observed that the session ended and a login prompt was presented.

Test 2: The evaluator repeated this test using an SSH connection and observed that the session ended and the SSH connection was terminated.

2.6.3 TSF-initiated Session Locking (NDcPP30e:FTA_SSL_EXT.1)

2.6.3.1 NDcPP30e:FTA_SSL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FTA_SSL_EXT.1) of the ST states an Authorized Administrator can configure maximum inactivity times individually for both local and remote administrative sessions using the "exec-timeout" command applied to the console and virtual terminal (vty) lines. The allowable inactivity timeout range is <0-35791> minutes. A value of 0 means there is no inactivity timeout enforced.

The configuration of the vty lines sets the configuration for the remote console access.

The line console settings are not immediately activated for the current session. The current line console session must be exited. When the user logs back in, the inactivity timer will be activated for the new session. The local interactive session terminates and does not lock. If a local user session is inactive for a configured period, the session will be terminated and will require re-identification and authentication to login. If a remote user session is inactive for a configured period, the session will be terminated and will require re-identification and authentication to establish a new session.

Component Guidance Assurance Activities: 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.

Section "Session Termination" of the Admin Guide discusses remote session termination. It provides instructions for setting termination values for inactive sessions.

Component Testing Assurance Activities: 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 reauthentication is needed when trying to unlock the session.

The evaluator set a session timeout of 1 minute and found that the local console session disconnected after 1 minute, as expected. The evaluator then repeated this test with a time of 2 minutes.

2.6.4 Default TOE Access Banners (NDcPP30e:FTA_TAB.1)

2.6.4.1 NDcPP30e:FTA_TAB.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall check the TSS to ensure that it details each administrative method of access (local and/or 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).

Section 6 (FTA_TAB.1) of the ST states the Administrator can configure an access banner that describes restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE. The banner will display on the local console port and SSH interfaces prior to allowing any administrative access.

Component Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it describes how to configure the banner message.

Section "Access Banner" of the Admin Guide explains how to configure the login banner.

Component Testing Assurance Activities: The evaluator shall also perform the following test:

a. Test 1: The evaluator shall follow 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.

The evaluator configured a banner and verified that the banner was displayed appropriately for console and SSH CLI logins.

2.7 Trusted path/channels (FTP)

2.7.1 Inter-TSF trusted channel (NDcPP30e:FTP_ITC.1)

2.7.1.1 NDcPP30e:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.1.2 NDcPP30e:FTP_ITC.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.1.3 NDcPP30e:FTP_ITC.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6.1 (FTP_ITC.1) of the ST states the TOE uses secure protocols to provide trusted communications between itself and authorized IT entities as specified in the table below:

IT Entity	TOE Acting as Client or Server	Secure Communication Mechanism / Protocol	Non-TSF Endpoint Identification
Syslog Server	Client	IPsec	X.509 Certificate

Component Guidance Assurance Activities: 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.

Section "IPsec" of the Admin Guide provide instructions on how to establish an IPsec connection. See FCS_IPSEC_EXT.1 for specific configuration details.

Section "IPsec Session Interruption and Recovery" of the Admin Guide explains that if an IPsec session with a peer is unexpectedly interrupted, the connection will be broken and the Administrator will find a connection time out error message in the audit log. The administrator can use the 'show crypto session detail' command to confirm the connection is broken. The number of active SAs will show zero. When a connection is broken no administrative interaction is required. The IPsec session will be reestablished (a new SA set up) once the peer is back online.

Component Testing Assurance Activities: 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.

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.

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.

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

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.

Further assurance activities are associated with the specific protocols.

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.

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.

The TOE utilizes IPsec for trusted channels protecting communication with an external audit server (syslog server).

A successful TOE IPsec connection supporting communication to an external audit server was established. Examining the packet capture from that test, we see that the IPsec connection between the TOE component and the external syslog server was established, the TOE initiated the connection, and application data transferred is encrypted (i.e., not plaintext).

The evaluator began a packet capture of traffic between the TOE and external audit server. With the connection established, the evaluator physically disconnected the network between the TOE and the remote audit server. The evaluator left the network disconnected several minutes then reconnected the wiring. Because the TOE automatically reconnects broken IPsec connections, the evaluator waited for the syslog server to begin receiving audit data again and stopped the packet capture shortly after traffic began flowing after the disruption. The evaluator observed that no data was transmitted unprotected.

The evaluator also used the TOE to initiate an IPsec protected communication pathway to an external authentication server. Examination of the packet capture obtained during this activity showed that the connection was protected by IPsec, the TOE initiated the connection, and all application data was transferred encrypted (i.e., not plaintext). The evaluator also performed the same physical disruption test during this test and observed that no data was transmitted unprotected.

Upon completion of these activities, the resulting transcripts and packet captures were inspected. This data showed the following:

Test 1: The TOE support for IPsec protected syslog was demonstrated.

Test 2: The TOE initiated the IPsec connection for syslog trusted channels.

Test 3: Syslog communication was not sent in plaintext.

Test 4: A physical disruption in the network resulted in an IPsec session interruption and no data was transmitted unprotected upon resumption.

2.7.2 Trusted Path (NDcPP30e:FTP_TRP.1/Admin)

2.7.2.1 NDcPP30e:FTP_TRP.1.1/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.2.2 NDcPP30e:FTP_TRP.1.2/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.2.3 NDcPP30e:FTP_TRP.1.3/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 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.

Section 6 (FTP_TRP.1/Admin) of the ST states all remote administrative communications take place over a secure encrypted SSHv2 session. The SSHv2 session is encrypted using AES encryption. The remote users (Authorized Administrators) can initiate SSHv2 communications with the TOE.

This matches the SFR claims of FCS_SSH_EXT.1 and FCS_SSHS_EXT.1.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation contains instructions for establishing the remote administrative sessions for each supported method.

Section "SSH Remote Administration Protocol" of the Admin Guide provides instructions on how to establish an SSH connection for remote administration. See FCS_SSH_EXT.1 for specific configuration details.

Component Testing Assurance Activities: 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.

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

Further assurance activities are associated with the specific protocols.

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.

The TOE offers remote administration via SSHv2 or SSH tunneled through IPsec to provide the trusted path (with protection from disclosure and modification) for all remote administration sessions.

The evaluator found in FCS_SSH_EXT.1 and FCS_SSHS_EXT.1 that the SSH connection is not sent in plaintext.

The evaluator found in FTP_ITC.1-t4 that the IPsec connection is not sent in plaintext.

The TOE is not distributed.

3. Protection Profile SAR Assurance Activities

The following sections address assurance activities specifically defined in the claimed Protection Profile that correspond with Security Assurance Requirements.

3.1 Security Target (ASE)

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

The assurance activity conclusions in Section 2 of this document address the requirements for the ST. Additionally, each specific work unit is performed in the proprietary Evaluation Technical Report (ETR).

3.2 Development (ADV)

3.2.1 Basic functional specification (ADV_FSP.1)

Assurance Activities: The EAs for this assurance component focus on understanding the interfaces (e.g., application programming interfaces, command line interfaces, graphical user interfaces, network interfaces) described in the AGD documentation, and possibly identified in the TOE Summary Specification (TSS) in response to the SFRs. Specific evaluator actions to be performed against this documentation are identified (where relevant) for each SFR in Section 2, and in EAs for AGD, ATE and AVA SARs in other parts of Section 5.

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

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

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

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.

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates). Explicitly labeling TSFI as security relevant or non-security relevant is not necessary. A TSFI is implicitly security relevant if it is used to satisfy an evaluation activity, or if it is identified in the ST or guidance documentation as adhering to the security policies (as presented in the SFRs). The intent is that these interfaces will be adequately tested and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied. According to the description above 'security relevant' corresponds to the combination of 'SFR-enforcing' and 'SFR-supporting' as defined in CC Part 3, paragraph 224 and 225.

The set of TSFI that are provided as evaluation evidence are contained in the Security Target and the guidance documentation.

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

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

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

It should be noted that there may be some SFRs that do not have a TSFI that is explicitly 'mapped' to invoke the desired functionality. For example, generating a random bit string or destroying a cryptographic key that is no longer needed are capabilities that may be specified in SFRs, but are not invoked by an interface.

The required EAs define the design and interface information required to meet ADV_FSP.1. If the evaluator is unable to perform some EA, then the evaluator shall conclude that an adequate functional specification has not been provided.

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. No additional 'functional specification' documentation is necessary to satisfy the Evaluation Activities specified in the SD.

3.3 Guidance documents (AGD)

3.3.1 Operational user guidance (AGD_OPE.1)

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

Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in Appendix B.4.2.1.

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR. For the related evaluation activities, the evaluation evidence documents Security Target, AGD documentation (user guidance) and functional specification documentation (if provided) shall be used as input documents. Each input document is subject to ALC_CMS.1-2 requirements.

In addition, the evaluator performs the EAs specified below.

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR. For the related evaluation activities, the evaluation evidence documents Security Target, AGD documentation (user guidance) and functional specification documentation (if provided) shall be used as input documents. Each input document is subject to ALC_CMS.1-2 requirements.

In addition, the evaluator performs the EAs specified below.

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.

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.

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

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.

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

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

b. The evaluator shall verify that this process includes 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).

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.

Section "FIPS Mode" of the Admin Guide states the TOE must be run in the FIPS mode of operation. The use of the cryptographic engine in any other mode was not evaluated not tested during the CC evaluation of the TOE. Instructions for configuring algorithms are provided with each protocol.

Section "Operational Environment" of the Admin Guide provides information for the Operational Environment to be CC compliant.

Section "Excluded Functionality" of the Admin Guide provides information on what functionality is excluded from the evaluation.

Section "Update TOE Software" of the Admin Guide describes the update process and states that the TOE will automatically verify the integrity of the stored image when loaded for execution. This section provides step by step instructions for installing an image.

3.3.2 Preparative procedures (AGD_PRE.1)

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

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

In addition, the evaluator performs the EAs specified below.

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).

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

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.

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

The evaluator shall examine the preparative procedures to ensure they include instructions on how to manage the TSF as a product and as a component of the larger Operational Environment in a manner that allows to preserve integrity of the TSF.

The intent of this requirement is to ensure there exists adequate preparative procedures (guidance in most cases) to put the TSF in a secure state (i.e., evaluated configuration). AGD_PRE.1 lists general requirements, the specific assurance activities implementing it are performed as part of FMT_SMF.1, FMT_MTD.1 and FMT_MOF.1 series of SFRs.

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 mandate that they shall be changed.

The evaluator had the Admin Guide to use when configuring the TOE. The completeness of the Admin Guide is addressed by its use in the AAs carried out in the evaluation.

3.4 Life-cycle support (ALC)

3.4.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activities: When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

ALC_CMC.1-1 The evaluator shall check that the TOE provided for evaluation is labelled with its reference.

989 The evaluator should ensure that the TOE contains the unique reference which is stated in the ST. This could be achieved through labelled packaging or media, or by a label displayed by the operational TOE. This is to ensure that it would be possible for consumers to identify the TOE (e.g. at the point of purchase or use).

The TOE may provide a method by which it can be easily identified. For example, a software TOE may display its name and version number during the start up routine, or in response to a command line entry. A hardware or firmware TOE may be identified by a part number physically stamped on the TOE.

Alternatively, the unique reference provided for the TOE may be the combination of the unique reference of each component from which the TOE is comprised (e.g. in the case of a composed TOE).

ALC_CMC.1-2 The evaluator shall check that the TOE references used are consistent.

If the TOE is labelled more than once then the labels have to be consistent. For example, it should be possible to relate any labelled guidance documentation supplied as part of the TOE to the evaluated operational TOE. This ensures that consumers can be confident that they have purchased the evaluated version of the TOE, that they have installed this version, and that they have the correct version of the guidance to operate the TOE in accordance with its ST.

The evaluator also verifies that the TOE reference is consistent with the ST.

If this work unit is applied to a composed TOE, the following will apply. The composed IT TOE will not be labelled with its unique (composite) reference, but only the individual components will be labelled with their appropriate TOE reference. It would require further development for the IT TOE to be labelled, i.e. during start-up and/or operation, with the composite reference. If the composed TOE is delivered as the constituent component TOEs, then the TOE items delivered will not contain the composite reference. However, the composed TOE ST will include the unique reference for the composed TOE and will identify the components comprising the composed TOE through which the consumers will be able to determine whether they have the appropriate items.

The evaluator verified that the ST, TOE, and Admin Guide are all labeled with the same hardware versions and software. The information is specific enough to procure the TOE and it includes hardware models and software versions. The evaluator checked the TOE software version and hardware identifiers during testing by examining the actual machines used for testing.

3.4.2 TOE CM coverage (ALC_CMS.1)

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

ALC_CMS.1-1 The evaluator shall check that the configuration list includes the following set of items:

a) the TOE itself;

b) the evaluation evidence required by the SARs in the ST.

ALC_CMS.1-2 The evaluator shall examine the configuration list to determine that it uniquely identifies each configuration item.

1103 The configuration list contains sufficient information to uniquely identify which version of each item has been used (typically a version number). Use of this list will enable the evaluator to check that the correct configuration items, and the correct version of each item, have been used during the evaluation.

See Section 3.3.1 above for an explanation of how all CM items are addressed.

3.4.3 Flaw reporting procedures (ALC_FLR.2)

Assurance Activities: When evaluating the flaw remediation procedures, the evaluator performs the work units as presented in the CEM.

ALC_FLR.2-1 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes the procedures used to track all reported security flaws in each release of the TOE.

ALC_FLR.2.2 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would produce a description of each security flaw in terms of its nature and effects.

ALC_FLR.2-3 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would identify the status of finding a correction to each security flaw.

ALC_FLR.2-4 The evaluator shall check the flaw remediation procedures to determine that the application of these procedures would identify the corrective action for each security flaw.

ALC_FLR.2-5 The evaluator shall examine the flaw remediation procedures documentation to determine that it describes a means of providing the TOE users with the necessary information on each security flaw.

ALC_FLR.2-6 The evaluator shall examine the flaw remediation procedures to determine that they describe procedures for the developer to accept reports of security flaws or requests for corrections to such flaws.

ALC_FLR.2-7 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure every reported flaw is corrected.

ALC_FLR.2-8 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would help to ensure that the TOE users are issued remediation procedures for each security flaw.

ALC_FLR.2-9 The evaluator shall examine the flaw remediation procedures to determine that the application of these procedures would result in safeguards that the potential correction contains no adverse effects.

ALC_FLR.2-10 The evaluator shall examine the flaw remediation guidance to determine that the application of these procedures would result in a means for the TOE user to provide reports of suspected security flaws or requests for corrections to such flaws.

Cisco provided a flaw remediation document that described the flaw reporting, tracking, and correction process. The evaluator has assessed that the conclusion of the process includes identifying the corrective action for each security flaw. Each specific work unit is performed in the proprietary Evaluation Technical Report (ETR).

3.5 Tests (ATE)

3.5.1 Independent testing - conformance (ATE_IND.1)

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

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

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

Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in Appendix B.4.3.1.

The evaluator created a Detailed Test Report (DTR) to address all aspects of this requirement. The DTR discusses the test configuration, test cases, expected results, and test results. The test configuration consisted of the following TOE platforms along with supporting products:

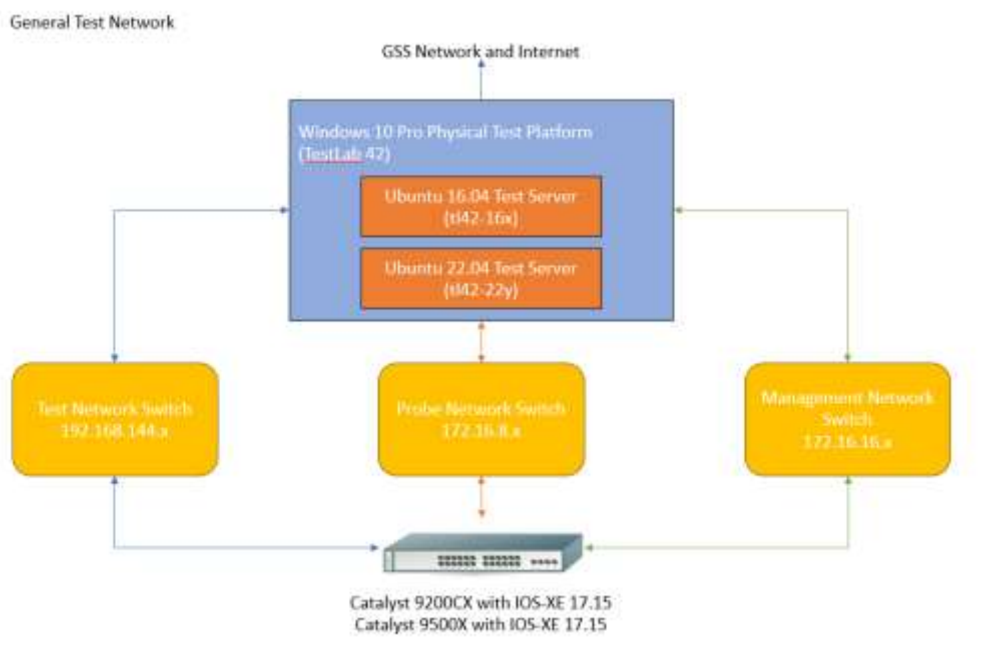
General Test Network
GSS Network and Internet
Windows 10 Pro Physical Test Platform (TestLab 42)
Ubuntu 16.04 Test Server (tl42-16x)
Ubuntu 22.04 Test Server (tl42-22y)
Test Network Switch 192.168.144.x
Probe Network Switch 172.16.8.x
Management Network Switch 172.16.16.x
Catalyst 9200CX with IOS-XE 17.15
Catalyst 9500X with IOS-XE 17.15

Figure 1 Test Setup

TOE Platforms:

- Catalyst 9200CX
  - 192.168.144.192
    - Physical interface MAC – 5cb1.2e12.ca03
    - VLAN interface MAC – 5cb1.2e12.ca77
  - 172.16.8.192
    - Physical interface MAC – 5cb1.2e12.ca01
    - VLAN interface MAC – 5cb1.2e12.ca5c
  - 172.16.16.192
    - Physical interface MAC – 5cb1.2e12.ca46
- Catalyst 9500X
  - 192.168.144.195
    - Physical interface MAC – 6887.c614.f007
    - VLAN interface MAC – 6887.c614.f102
  - 172.16.8.195
    - Physical interface MAC – 6887.c614.f02b
    - VLAN interface MAC – 6887.c614.f102
  - 172.16.16.195
    - Physical interface MAC – 6887.c614.f000

Supporting Products:

- TestLab42 (Windows 10 Pro)
  - 192.168.144.102 – 00-1B-21-38-A6-3D
  - 172.16.16.102 – 00-1B-21-38-A6-3C
- TL42-16x (Ubuntu 16.04)
  - 192.168.144.35 – 00:15:5d:00:c0:12
  - 172.16.8.35 – 00:15:5d:00:c0:13
  - 172.16.16.35 - 00:15:5d:00:c0:14
- TL42-22y (Ubuntu 22.04)
  - 192.168.144.103 – 00:15:5d:00:c0:02
  - 172.16.8.103 – 00:15:5d:00:c0:03
  - 172.16.16.103 - 00:15:5d:00:c0:04

Supporting Software:

The Gossamer Test server utilized both a Windows and Ubuntu environment.

The Windows supporting environment included the following:

- Wireshark version 4.0.5
- Putty version 0.76 (used to connect to device console and Ubuntu environment)

The Ubuntu 16 supporting environment included the following:

- Openssh-client version 7.2p2
- Nmap version 7.01
- tcpdump 4.9.3
- libpcap version 1.7.4
- OpenSSL 1.0.2g
- Rsyslog 8.16.0
- strongSwan U5.3.5

3.6 Vulnerability Assessment (AVA)

3.6.1 Vulnerability survey (AVA_VAN.1)

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

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

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

In addition to the activities specified by the CEM in accordance with Table 2, the evaluator shall perform the following activities.

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.

The developer shall provide documentation identifying the list of software and hardware components[7] that compose the TOE. Hardware components should identify compute-capable hardware components, at a minimum that must include the processor, and where applicable, discrete crypto ASICs, TPMs, etc. 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 implementations, (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.

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

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

b. a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, Table 2]

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

The evaluator shall formulate hypotheses in accordance with process defined in Appendix A. The evaluator shall document 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.

The vulnerability analysis is in the Detailed Test Report (DTR) prepared by the evaluator. The vulnerability analysis includes a public search for vulnerabilities and fuzz testing. None of the public search for vulnerabilities or the fuzz testing uncovered any residual vulnerability.

The evaluator searched the National Vulnerability Database (https://web.nvd.nist.gov/vuln/search), Vulnerability Notes Database (http://www.kb.cert.org/vuls/), Rapid7 Vulnerability Database (https://www.rapid7.com/db/vulnerabilities), Tipping Point Zero Day Initiative (http://www.zerodayinitiative.com/advisories), Known Vulnerability Exploit Catalog (https://www.cisa.gov/known-exploited-vulnerabilities-catalog), Tenable Network Security (http://nessus.org/plugins/index.php?view=search), Offensive Security Exploit Database (https://www.exploit-db.com/) on 7/15/2025 (from 12/1/2023) with the following search terms: "Cisco Catalyst", "IOS-XE 17.15", "Cisco IOS XE 17.15", "ARM Cortex-A53", "Xilinx ZU3EG", "Catalyst 9200CX", "Catalyst 9500X", "Catalyst 9600X", "Intel Xeon D-1564N", "Intel Xeon D-1573N", "IC2M", "IOS Common Cryptographic Module".

3.6.2 Additional Flaw Hypothesis (AVA_VAN.1)

Assurance Activities: The following additional tests shall be performed:1.) [Conditional]: If the TOE is a TLS server and supports ciphersuites that use RSA transport (e.g. supporting TLS_RSA_WITH_* ciphers) the following test shall be performed. Where RSA Key Establishment schemes are claimed and especially when PKCS#1 v1.5* padding is used, the evaluators shall test for implementation flaws allowing Bleichenbacher and Klima et al. style attacks, including Bock et al's ROBOT attacks of 2017 in the flaw analysis. Even though Bleichenbacher's original paper is two decades old, Bock et al. found these attacks to still be effective in weakening the security of RSA key establishment in current products. Bleichenbacher and Klima et al. style attacks are complex and may be difficult to detect, but a number of software testing tools have been created to assist in that process. The iTC strongly recommends that at least one of the tools mentioned in Bock et al's ROBOT attacks of 2017 webpage or paper, as effective to detect padding oracle attacks, be used to test TOE communications channels using RSA based Key Establishment (related sources: http://archiv.infsec.ethz.ch/education/fs08/secsem/bleichenbacher98.pdf, https://eprint.iacr.org/2003/052, https://robotattack.org/). Network Device Equivalency Consideration.

The TOE does not support a TLS server implementation and therefore is not subject to Bleichenbacher attacks.