www.GossamerSec.com

Assurance Activity Report for Cisco FTD 7.4 on Firepower 4100 and 9300 Series with FMC/FMCv

Version 0.2
02/25/25

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-11516

Revision History

Revision	Date	Authors	Summary
Version 0.1	02/12/25	Kalmus	Initial draft
Version 0.2	02/25/25	Kalmus	Addressed ECR Comments

The TOE Evaluation was Sponsored by:

Cisco Systems, Inc.
170 West Tasman Dr.
San Jose, CA 95

Evaluation Personnel:

- Douglas Kalmus
- Linh Le

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

This document presents evaluations results of the Cisco FTD 7.4 on Firepower 4100 and 9300 Series with FMC/FMCv- NDcPP22e/STFFW14e/IPS10/VPNGW13 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 FTD 7.4 on Firepower 4100 and 9300 Series with FMC/FMCv** consisting of the following hardware models running software versions: Firepower Threat Defense (FTD) 7.4, Firepower Management Center (FMC) 7.4 and Firepower Management Center Virtual (FMCv) 7.4, FXOS Version 2.14.

- Firepower 4100 & 9300 Series (4112, 4115, 4125, 4145 and 9300)
- Cisco Firepower Management Center (FMC) (FMC1600, FMC2600, FMC4600, FMC1700 and FMC4700)
- FMCv running on ESXi 7.0 on the Unified Computing System (UCS) UCSC-C220-M5, UCSC-C240-M5, UCSC-C480-M5, UCSC-C220-M6, UCSC-C225-M6, UCSC-C240-M6, UCSC-C220-M7, UCSC-C240-M7and UCS-E1100D-M6

The evaluation team ran the entire test suite on each of the following devices:

- Cisco FMC1600 running FMC v7.4
- Cisco FMCv running on ESXi 7.0 with FMC v7.4 on UCSC-C220-M5
- Cisco Firepower 4145 running Firepower Threat Defense v7.4 & FXOS v2.14

The software image is the same among the models in each of the groups included in the evaluation. There are two images that is used for all Firepower 4100 series devices. This is an image for FTD and for FXOS. All 4100 and 9300 models use the same images for both FTD and FXOS. One image is used for all FMC devices and one image is used for all FMCv devices. The only differences between the images among these devices is related to the different hardware characteristics and does not affect any of the security relevant functionality. Any differing hardware characteristics among the models in each group affect only non-security relevant functionality such as throughput, processing speed, number and type of network connections supported, number of concurrent connections supported, and amount of storage. All security functions provided by the TOE 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.

Full testing was performed on both the FMC and FMCv because each uses a different image. For FMCv on ESXi, the evaluators fully tested on the C220-M5. The same FMCv image is installed on the other models of UCS servers included in the evaluated configuration. The differences in hardware characteristics among the UCS servers similarly only affect non-security relevant functionality as described above. Those servers simply provide resources to the FMCv images and are not making security relevant decisions. The same also applies to all models of the FMC, which all use the same image across all hardware models with the differences in hardware only affecting non-security relevant functionality. FTD and FXOS were also tested individually.

The TOE includes the following models:

Model	Processor ID	Microarchitecture
FP 4112	Intel Xeon Silver 4116	Sky Lake
FP 4115	Intel Xeon Silver 4116	Sky Lake
FP 4125	Intel Xeon Gold 6130T	Sky Lake
FP 4145	Intel Xeon Gold 6152	Sky Lake
FP 9300 SM-40	Intel Xeon Gold 6138T	Sky Lake
FP 9300 SM-48	Intel Xeon Platinum 8160	Sky Lake
FP 9300 SM-56	Intel Xeon Platinum 8176	Sky Lake
Supervisor Blade	Intel i3-3115C	Ivy Bridge
FMC1600	Intel Xeon Silver 4110	Skylake
FMC1700	AMD EPYC 7232P	Zen 2
FMC2600	Intel Xeon Silver 4110	Skylake
FMC2700	AMD EPYC 7282	Zen 2
FMC4600	Intel Xeon Silver 4214	Cascade Lake
FMC4700	AMD EPYC 7352	Zen 2
C220 M5	Intel Xeon Bronze 3104	Skylake
	Intel Xeon Silver 4110	Skylake
	Intel Xeon Gold 6128	Skylake
	Intel Xeon Platinum 8153	Skylake
	Intel Xeon Platinum 8160	Skylake
C240 M5	Intel Xeon Bronze 3104	Skylake
	Intel Xeon Silver 4110	Skylake
	Intel Xeon Gold 6128	Skylake
	Intel Xeon Platinum 8153	Skylake
	Intel Xeon Platinum 8160	Skylake
C480-M5	Intel Xeon Bronze 3104	Skylake
	Intel Xeon Silver 4110	Skylake
	Intel Xeon Gold 6128	Skylake
	Intel Xeon Platinum 8153	Skylake
	Intel Xeon Platinum 8160	Skylake
C220 M6	Intel Xeon Silver 4310	Ice Lake
	Intel Xeon Silver 4314	Ice Lake
	Intel Xeon Silver 4316	Ice Lake
	Intel Xeon Gold 5315Y	Ice Lake
	Intel Xeon Gold 5318N	Ice Lake
	Intel Xeon Gold 6312U	Ice Lake
	Intel Xeon Gold 6342	Ice Lake
	Intel Xeon Platinum 8351N	Ice Lake

Model	Processor ID	Microarchitecture
C225 M6	AMD EPYC 7232P	Zen 2
	AMD EPYC 7252	Zen 2
	AMD EPYC 7262	Zen 2
	AMD EPYC 7272	Zen 2
	AMD EPYC 7282	Zen 2
	AMD EPYC 72F3	Zen 3
	AMD EPYC 7302	Zen 2
	AMD EPYC 7313	Zen 3
	AMD EPYC 7343	Zen 3
	AMD EPYC 7352	Zen 2
	AMD EPYC 7373X	Zen 3
	AMD EPYC 73F3	Zen 3
	AMD EPYC 7402	Zen 2
	AMD EPYC 74F3	Zen 3
	AMD EPYC 7543	Zen 3
C240 M6	Intel Xeon Silver 4310	Ice Lake
	Intel Xeon Silver 4314	Ice Lake
	Intel Xeon Silver 4316	Ice Lake
	Intel Xeon Gold 5315Y	Ice Lake
	Intel Xeon Gold 5318N	Ice Lake
	Intel Xeon Gold 6312U	Ice Lake
	Intel Xeon Gold 6342	Ice Lake
	Intel Xeon Platinum 8351N	Ice Lake
C220 M7	Intel Xeon Bronze 3408U	Sapphire Rapids
	Intel Xeon Silver 4410T	Sapphire Rapids
	Intel Xeon Gold 5411N	Sapphire Rapids
	Intel Xeon Gold 6414U	Sapphire Rapids
	Intel Xeon Platinum 8444H	Sapphire Rapids
	Intel Xeon Platinum 8452Y	Sapphire Rapids
C240 M7	Intel Xeon Bronze 3408U	Sapphire Rapids
	Intel Xeon Silver 4410T	Sapphire Rapids
	Intel Xeon Gold 5411N	Sapphire Rapids
	Intel Xeon Gold 6414U	Sapphire Rapids
	Intel Xeon Platinum 8444H	Sapphire Rapids
	Intel Xeon Platinum 8452Y	Sapphire Rapids
E1100D-M6	Intel Xeon D-1746TER	Ice lake
	Intel Xeon D-2796TE	Ice Lake

Evaluated Model Technical Details

The TOE also uses a cryptographic accelerator to support the IPsec implementation (for specific 4100 series hardware models). The evaluated models use the following cryptographic accelerators.

Model	Processor ID
FP 4112	Cavium NITROX III series die v1.1
FP 9300 (SM-40)	
FP 4115	NITROX-V GC
FP 4125	
FP 4145	
FP 9300 (SM-48 & 56)	

Evaluated Model Crypto Accelerator

1.1.2 CAVP Equivalence

The TOE is the Cisco FTD 7.4 on Firepower 4100 and 9300 Series with FMC/FMCv which includes cryptographic libraries that perform all cryptographic operations.

- CiscoSSL FOM Cryptographic implementation version 7.3a - (FTD & FMC)
- CiscoSSL version 7.3a FOM - Virtual - (FMCv)
- Nitrox III crypto engine (Nitrox III series die) or Nitrox V crypto engine (Nitrox V-GC)

These cryptographic libraries perform all cryptographic operations that provide the cryptographic functions identified in the following table.

The evaluation encompasses several physical devices. These devices utilize several processors. The table in the previous subsection lists the appliances, cryptographic modules and processors that are claimed in the ST. The following table maps the Cryptographic operations, related Security Functional Requirements (SFR) and standards to the CAVP certificates that demonstrate compliance to these SFR and standards.

Functions	Requirement	Standard	Certificate #'s
Encryption/Decryption			
AES CBC (128, 192, and 256 bits)	FCS_COP.1/DataEncryption	FIPS Pub 197 ISO 10116 NIST SP 800-38A ISO 19772	A 4446 A 4595
AES GCM (128, 192, and 256 bits)	FCS_COP.1/DataEncryption	ISO 19772 FIPS Pub 197 NIST SP 800-38A	A 4446 A 4595
Cryptographic hashing			
SHA-1, SHA-256, SHA-384, SHA-512	FCS_COP.1/Hash	FIPS Pub 180-4 ISO/IEC 10118-3:2004	A 4446 A 4595
Keyed-hash message authentication			
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (digest sizes and block sizes of 160, 256, 384 and 512 bits)	FCS_COP.1/KeyedHash	FIPS Pub 198-1 FIPS Pub 180-4 ISO/IEC 9797-2:2011	A 4446 A 4595
Cryptographic signature services			
RSA Digital Signature (rDSA) (2048, 3072 bits)	FCS_COP.1/SigGen	FIPS Pub 186-4 ISO/IEC 9796-2	A 4446 A 4595

Functions	Requirement	Standard	Certificate #'s
ECDSA Digital Signature (P-256, P-384, P-521)	FCS_COP.1/SigGen	FIPS Pub 186-4 ISO/IEC 14888-3	A 4446 A 4595
Random bit generation			
HMAC-DRBG with platform based noise sources with a minimum of 256 bits of non-determinism	FCS_RBG_EXT.1	FIPS SP 800-90A ISO/IEC 18031:2011	A 4446 A 4595
Key generation			
RSA Key Generation (2048-bit, 3072-bit)	FCS_CKM.1	FIPS Pub 186-4 ISO/IEC 9796-2	A 4446 A 4595
ECC Key Generation (P-256, P-384, P-521)	FCS_CKM.1	FIPS PUB 186-4	A 4446 A 4595
FFC Scheme using key sies of 2048-bit or greater DSA KeyPairGen	FCS_CKM.1	FIPS PUB 186-4	A 4446 A 4595
FFC Schemes using 'safe-prime'	FCS_CKM.1	NIST SP 800-56A Revision 3	Tested with known good implementation
Key establishment			
RSA	FCS_CKM.2	RSAES-PKCS1-v1_5	Tested with known good implementation
KAS ECC P-256, P-384, P-521	FCS_CKM.2	NIST SP 800-56A Rev 3	A 4446 A 4595
KAS FFC	FCS_CKM.2	NIST SP 800-56A Rev 3	A 4446 A 4595
FFC Schemes using 'safe-prime' groups	FCS_CKM.2	NIST SP 800-56A Rev 3	Tested with known good implementation

CiscoSSL FOM Version 7.3a & CiscoSSL version 7.3a FOM Virtual

Functions	Requirement	Standard	Certificate #
Encryption/Decryption			
AES CBC (128, 192, and 256 bits)	FCS_COP.1/DataEncryption	FIPS Pub 197 ISO 10116 NIST SP 800-38A ISO 19772	AES 2034
AES GCM (128, 192, and 256 bits)	FCS_COP.1/DataEncryption	ISO 19772 FIPS Pub 197 NIST SP 800-38A	AES 2035
Cryptographic hashing			
SHA-1, SHA-256, SHA-384, SHA-512	FCS_COP.1/Hash	FIPS Pub 180-4 ISO/IEC 10118-3:2004	SHS 1780
Keyed-hash message authentication			
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (digest sizes and block sizes of 160, 256, 384 and 512 bits)	FCS_COP.1/KeyedHash	FIPS Pub 198-1 FIPS Pub 180-4 ISO/IEC 9797-2:2011	HMAC 1233

Nitrox III crypto engine

Functions	Requirement	Standard	Certificate #
Encryption/Decryption			
AES CBC (128, 192 and 256 bits)	FCS_COP.1/DataEncryption	FIPS Pub 197 ISO 10116 NIST SP 800-38A ISO 19772	C 1026
AES GCM (128, 192 and 256 bits)	FCS_COP.1/DataEncryption	ISO 19772 FIPS Pub 197 NIST SP 800-38A	C 1026
Cryptographic hashing			
SHA-1, SHA-256, SHA-384, SHA-512	FCS_COP.1/Hash	FIPS Pub 180-4 ISO/IEC 10118-3:2004	C 1026
Keyed-hash message authentication			
HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (digest sizes and block sizes of 160, 256, 384 and 512 bits)	FCS_COP.1/KeyedHash	FIPS Pub 198-1 FIPS Pub 180-4 ISO/IEC 9797-2:2011	C 1026

Nitrox V crypto engine

All algorithms were tested on their tested and claimed operational environments.

1.2 References

The following evidence was used to complete the Assurance Activities:

- Cisco FTD 7.4 on Firepower 4100 and 9300 Series with FMC/FMCv, Version 1.0, February 25, 2025 **(ST)**
- Cisco FTD v7.4 on Firepower 4100 and 9300 Series with FMC/FMCv Common Criteria Supplemental User Guide, Version 0.1, February 5, 2025 **(FTD Admin Guide)**
- Cisco FXOS 2.14 on Firepower 4100/9300 for FTD Preparative Procedures & Operational User Guide for the Common Criteria Certified Configuration, Version 1.0, February 25, 2025 **(FXOS Admin Guide)**
- Cisco FTD v7.4 with FMC/FMCv Common Criteria User Guide Supplement IPS & VPN Functionality, Version 0.4, November 22, 2024 **(Supplement)**

2. Protection Profile SFR Assurance Activities

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

2.1 Security audit (FAU)

2.1.1 Audit Data Generation (NDcPP22e:FAU_GEN.1)

2.1.1.1 NDcPP22e:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.1.2 NDcPP22e: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 should 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 6.1 (FAU_GEN.1) in the ST states that for audit messages related to management of cryptographic keys, the audit message details include the name of the certificate associated with the key.

Section 9 in the ST provides a mapping of the distributed TOE components to the SFRs and the required audit events. The evaluator confirmed that all components generating audit information for a particular SFR also contribute to that SFR and that the table accounts for all TOE auditable events.

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 “Auditable Events” in the FTD Admin Guide states that each appliance generates an audit event for each user interaction with the web interface and CLI command executed. Each event includes at least a timestamp, the username of the user whose action generated the event, a source IP, and text describing the event. This section provides a table identifying all of the required audit events consistent with the ST along with a sample of each record for FTD and FMC for each SFR, where applicable.

Section “Auditable Events” in the FXOS Admin Guide states that the appliances that are part of the Cisco FP 4100 and 9300 System generate an audit record for each user interaction with the web interface, and also record system status messages in the system log. For the CLI, the appliance also generates an audit record for every action executed. Each appliance generates an audit event for each user interaction with the web interface and CLI command executed. Each event includes at least a timestamp, the user name of the user whose action generated the event, a source IP, and text describing the event. This section provides a table identifying all of the required audit events consistent with the ST along with a sample of each record for FXOS for each SFR, where applicable.

From a review of the ST, the Guidance, and from 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 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 event when running the other security functional tests described by the protection profiles. For example, the required event for FPT_STM.1 is Changes to Time. The evaluator collected these audit records when modifying the clock using administrative commands. The evaluator then recorded these 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.

As the TOE is distributed, audit collection was performed on each distinct component (FXOS, FTD, and FMC devices). Each has its own logging mechanism. The evaluator verified that auditable events including multiple components such as FCO_CPC_EXT.1 (ITT communications) were audited on both TOE components. These audits were collected and recorded in the DTR, where both the FTD and FMC have a collected audit for initiation and termination of the secure ITT channel, for example. (This example is not relevant to FXOS which is not directly involved in the ITT process between FTD and FMC).

2.1.2 Security Audit Data Generation (STFFW14e:FAU_GEN.1)

2.1.2.1 STFFW14e:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: No additional Evaluation Activities are specified.

No additional Evaluation activities are specified.

Component Guidance Assurance Activities: In addition to the Evaluation Activities specified in the Supporting Document for the Base-PP, the evaluator shall check the guidance documentation to ensure that it describes the audit records specified in Table 2 of the PP-Module in addition to those required by the Base-PP. If the optional SFR FFW_RUL_EXT.2 is claimed by the TOE, the evaluator shall also check the guidance documentation to ensure that it describes the relevant audit record specified in Table 3 of the PP-Module.

Section "Auditable Events" of the FTD Admin Guide states that the appliances that are part of the Cisco FTD System generate an audit record for each user interaction with the web interface, and also record system status messages in the system log. For the CLI, the appliance also generates an audit record for every command executed.

Each event includes at least a timestamp, the username of the user whose action generated the event, a source IP, and text describing the event. This section provides a table identifying all of the required audit events consistent with the ST along with a sample of each record for each TOE component and SFR where applicable. The evaluator verified that all audit records specified in Table 2 of the PP-Module and those required by the Base-PP are described.

Component Testing Assurance Activities: In addition to the Evaluation Activities specified in the Supporting Document for the Base-PP, the evaluator shall perform tests to demonstrate that audit records are generated for the auditable events as specified in Table 2 of the PP-Module and, if the optional SFR FFW_RUL_EXT.2 is claimed by the TOE, Table 3.

The evaluator created a list of the required audit events. The evaluator then collected the audit event when running the other security functional tests described by the protection profiles. For example, the required events for FFW_RUL_EXT.1 are indications that an access-list rule is processed and the traffic is handled accordingly. The evaluator sends traffic matching access-list rules and ensures that the TOE logs the events. The evaluator then recorded these 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.

2.1.3 Audit Data Generation (IPS) - per TD0595 (IPS10:FAU_GEN.1/IPS)

2.1.3.1 IPS10:FAU_GEN.1.1/IPS

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.2 IPS10:FAU_GEN.1.2/IPS

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 describes how the TOE can be configured to log IPS data associated with applicable policies.

The evaluator shall verify that the TSS describes what (similar) IPS event types the TOE will combine into a single audit record along with the conditions (e.g., thresholds and time periods) for so doing. The TSS shall also describe to what extent (if any) that may be configurable.

For IPS_SBD_EXT.1, for each field, the evaluator shall verify that the TSS describes how the field is inspected and if logging is not applicable, any other mechanism such as counting that is deployed.

Section 6.1 (IPS_SBD_EXT.1) of the ST states that the administrator can configure the Sensor in either a passive or inline deployment. The administrator can configure one or more physical ports on a managed Sensor as passive interfaces and deploy the intrusion policy to that interface via security zone (i.e., the interface is added to the zone). In an inline IPS deployment, the administrator configures the Sensor transparently on a network segment by binding two ports together. The administrator can configure one or more physical ports on a managed Sensor as inline interfaces then assign a pair of inline interfaces to an inline set. The intrusion policy is then deployed to that inline set via security zone. The management interface (typically eth0) is separate from the other data monitoring interfaces (used as passive or inline) on the Sensor. It is used to set up and register the Sensor to the FMC.

When the system identifies a possible intrusion, it generates an intrusion or preprocessor event (sometimes collectively called intrusion events). Each intrusion event in the database includes an event header and contains information about the event name and classification; the source and destination IP addresses; ports; the process that generated the event; and the date and time of the event, as well as contextual information about the source of the attack and its target. For packet-based events, the TOE also logs a copy of the decoded packet header and payload for the packet or packets that triggered the event. The packet decoder, the preprocessors, and the intrusion rules engine can all cause the TOE to generate an event.

Within the intrusion rules engine, most intrusion rules are written so that they generate intrusion events when triggered by packets. Until the administrator deploys new policies to the network interface, rules in the currently deployed intrusion policies behave as follows:

- Disabled rules remain disabled.
- Rules set to **Alert** continue to generate events when triggered.
- Rules set to **Block** continue to generate events and drop offending packets when triggered.

The administrator can set thresholds for individual rules, per intrusion policy, to limit the number of times the system logs and displays an intrusion event based on how many times the event is generated within a specified time period. This can prevent the TOE from being overwhelmed with a large number of identical events.

Section 6.1 (FAU_GEN.1/IPS) in the ST states that for each possible intrusion identified by the system, the TOE will generate an event log for each intrusion event that occurs and event types are not combined. Managed Sensors will transmit their events to the FMC where the administrators can view the aggregated data and gain a greater understanding of the attacks against the entire network. This section also includes a list of the intrusion event information that can be viewed, searched, filtered and sorted by the system including: Protocol, Application Protocol, Destination IP, Source IP, Destination Port / ICMP Code, Source Port / ICMP Type, Access Control Policy, Access Control Rule, Intrusion Policy and Count. The definition of Count indicates the number of events that match the information that appears in each row. The Count field appears only after a constraint is applied that creates two or more identical rows. Basic contents such as date, time, and type can also be used to filter and sort. Note only Administrators and Intrusion Admins have access to the intrusion events.

The TOE can be configured to generate intrusion events. There are certain header fields that should not be used to trigger intrusion events (in Inline mode or Passive mode). Logging events related to these fields would generate a deluge of intrusion audit records that would prevent IPS analysts from figuring out what security incidents occur in their monitored network. In addition, logging these fields will provide no benefits. Per mod_ips_v1.0, the following fields can be inspected and if in inline mode, dropped or modified (i.e., normalized):

- All checksum fields

- TCP Reserved field

- TCP Urgent Pointer field

In inline mode, the TOE can count invalid checksum packets that are dropped. The TOE can also count the packets that get normalized or dropped because of failed normalization.

The Threshold feature allows administrators to control the number of events that are generated per rule over time. They can limit notification to the specified number of event instances per time period or provide notification once per time period after a specified number of event instances. The administrator must specify if the event instances will be tracked by source or destination IP address, the count or the number of event instances, and the number of seconds for the time period for which event instances are tracked. (The ST notes that the TOE's definition of the term "threshold" matches the definition of the term "frequency" in the IPS PP module, thus 'frequency' rather than 'threshold' is selected in the IPS_ABD_EXT.1.1 requirement).

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance describes how to configure the TOE to result in applicable IPS data logging.

The evaluator shall verify that the operational guidance provides instructions for any configuration that may be done in regard to logging similar events (e.g., setting thresholds, defining time windows, etc.).

Section "Managing Intrusion Policies" in the Supplement describes the creation of IPS events and what options are available to a system administrator to monitor or react to rules being triggered. Subsection "Intrusion Rule Actions" describes how to enable or disable a rule and specifies that when an event is enabled (either to Block or Alert), a log will be generated each time the policy is triggered. If an event is disabled, no log will be generated. Subsection "Adding and Modifying Intrusion Event Thresholds" references an external link to the Snort configuration guide which describes how to modify intrusion event thresholds including a count of events and timeframe of events.

Component Testing Assurance Activities: The evaluator shall test that the interfaces used to configure the IPS polices yield expected IPS data in association with the IPS policies. A number of IPS policy combination and ordering scenarios need to be configured and tested by attempting to pass both allowed and anomalous network traffic matching configured IPS policies in order to trigger all required IPS events. Note the following:

- This activity should have been addressed with a combination of the Test EAs for the other IPS requirements.

- As part of testing this activity, the evaluator shall also ensure that the audit data generated to address this SFR can be handled in the manner that FAU_STG_EXT.1 requires for all audit data.

This requirement has been satisfied by the collecting of audits during IPS testing in IPS_SBD_EXT.1, IPS_ABD_EXT.1, and IPS_IPB_EXT.1 which were all captured on a remote syslog server, matching the behavior identified in FAU_STG_EXT.1.

2.1.4 Audit Data Generation (VPN Gateway) (VPNGW13:FAU_GEN.1/VPN)

2.1.4.1 VPNGW13:FAU_GEN.1.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.4.2 VPNGW13:FAU_GEN.1.2/VPN

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 verify that it describes the audit mechanisms that the TOE uses to generate audit records for VPN gateway behavior. If any audit mechanisms the TSF uses for this are not used to generate audit records for events defined by FAU_GEN.1 in the Base-PP, the evaluator shall ensure that any VPN gateway-specific audit mechanisms also meet the relevant functional claims from the Base-PP.

For example, FAU_STG_EXT.1 requires all audit records to be transmitted to the OE over a trusted channel.

This includes the audit records that are required by FAU_GEN.1/VPN. Therefore, if the TOE has an audit mechanism that is only used for VPN gateway functionality, the evaluator shall ensure that the VPN gateway related audit records meet this requirement, even if the mechanism used to generate these audit records does not apply to any of the auditable events defined in the Base-PP.

The ST does not identify and VPN-Gateway specific audit mechanisms. Once generated, audits are stored in different internal databases depending on the type of event. However, the TSF uses the same audit mechanism to generate all FAU_GEN.1 audits, including VPNGW audits.

Section 6.1, FAU_GEN.1 of [ST] states that Audit events recording FMC administrator actions are stored in the Audit Event Database, network traffic events transmitted from FTD to FMC are stored in separate databases on FMC: firewall events (triggered by Access Control Policy rules) are stored in Connection Database; VPN events are stored in the VPN Troubleshooting Database; and the IPS events are stored in Intrusion Event Database.

Messages generated by FTD, including FTD system messages, firewall events, and VPN events are stored locally on FTD and are immediately transmitted from FTD to an external syslog server. As mentioned in the preceding paragraph, the firewall, VPN and IPS events are directly sent to FMC for retention in the FMC databases via secure TLS channel (Note: The IPS events are not stored locally on FTD but are transmitted to an external syslog server via the FMC. IPS events generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC). If the connection between FTD and FMC is interrupted, the IPS messages are transmitted once connectivity is restored. As the system, firewall event and VPN event messages are generated by FTD, they are immediately transmitted from FTD to a remote syslog server and stored in a local buffer (buffer size configurable from 4096-52428800 bytes) which overwrites old messages with new ones when storage limits are reached. The local logs are viewable from the FTD CLI shell by using “show logging”.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to verify that it identifies all security-relevant auditable events claimed in the ST and includes sample records of each event type. If the TOE uses multiple audit mechanisms to generate different sets of records, the evaluator shall verify that the operational guidance identifies the audit records that are associated with each of the mechanisms such that the source of each audit record type is clear.

Section “Auditable Events” of the FTD Admin Guide states that the appliances that are part of the Cisco FTD System generate an audit record for each user interaction with the web interface, and also record system status messages in the system log. For the CLI, the appliance also generates an audit record for every command executed.

Each event includes at least a timestamp, the username of the user whose action generated the event, a source IP, and text describing the event. This section provides a table identifying all of the required audit events consistent with the ST along with a sample of each record for each TOE component and SFR where applicable.

Component Testing Assurance Activities: The evaluator shall test the audit functionality by performing actions that trigger each of the claimed audit events and verifying that the audit records are accurate and that their format is consistent with what is specified in the operational guidance. The evaluator may generate these audit events as a consequence of performing other tests that would cause these events to be generated.

The evaluator created a list of the required audit events. The evaluator then collected the audit event when running the other security functional tests described by VPNGW13. For example, one of the required events for FTP_ITC.1/VPN is "Initiation of the trusted channel". The evaluator collected these audit records when testing VPN connections. The evaluator then recorded these audit events in the proprietary Detailed Test Report (DTR). This process was performed for all required audits applicable to VPNGW13.

The evaluator found the contents of these audits and their format to be consistent with the AGD.

2.1.5 User identity association (NDcPP22e:FAU_GEN.2)

2.1.5.1 NDcPP22e: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 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 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 NDcPP2e: FAU_GEN.1.1 where this test was performed in conjunction.

For the distributed TOEs case, each TOE component has a distinct logging mechanism and an associated secure channel. The evaluator observed that when the secure ITT channel was built, audit events were generated on the FTD and FMC that included the identity of the components involved with the connection. These are presented in the propriety Detailed Test Report (DTR).

2.1.6 Security Audit Generation (NDcPP22e:FAU_GEN_EXT.1)

2.1.6.1 NDcPP22e:FAU_GEN_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

See FAU_GEN.1

Component Guidance Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

See FAU_GEN.1

Component Testing Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

See NDcPP2e: FAU_GEN.1.1

2.1.7 Audit Review (IPS10:FAU_SAR.1)

2.1.7.1 IPS10:FAU_SAR.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.7.2 IPS10:FAU_SAR.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 verify that it describes the ability of administrators to view IPS data from the IPS events, the format in which this IPS data is displayed, and how an administrator is authorized to view this data.

Section 6.1 (FAU_SAR.1 [IPS]) in the ST states that the TOE will generate an event log for each intrusion event that occurs. Each event log will include a record of the date, time, type of exploit, and contextual information about the source of the attack and its target. For packet-based events, a copy of the packet or packets that triggered the event is also recorded. Managed Sensors will transmit their events to the FMC where the administrators can view the aggregated data and gain a greater understanding of the attacks against the entire network. Only Administrators and Intrusion Admins have access to the intrusion events. The administrators can also deploy the managed Sensors inline allowing them to configure the Sensors to drop or modify packets that are harmful. The web-based UI is the only way to view the intrusion events (Analysis > Intrusions > Events).

This section further provides a list and description of the intrusion event information that can be viewed, searched, filtered, and sorted by the system. In addition, basic contents such as date, time, and type can also be used to filter and sort.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to verify that it provides instructions on how to access and interpret IPS events using the TOE's management interface.

Section “Management of Intrusion Events” in the FTD Admin Guide states that when the system identifies a possible intrusion, it generates an intrusion event, which is a record of the date, time, the type of exploit, and contextual information about the source of the attack and its target. For packet-based events, a copy of the packet or packets that triggered the event is also recorded. Managed Devices transmit their events to the Firepower Management Center where the aggregated data can be viewed in order to gain a greater understanding of the attacks against network assets. A managed device can also be deployed as an inline, switched, or routed intrusion system, which allows the device to be configured to drop or replace packets that are known to be harmful.

Sections “Viewing Intrusion Events”, “Searching Intrusion Events” and “Sorting and filtering Intrusion Events” in the FTD Admin Guide provide instructions for viewing, searching and filtering intrusion event information. This section also provides a sample view of some intrusion events and a description of each field.

Section “Managing Intrusion Policies” in the Supplement contains many subsections, each outlining instructions on performing or understanding the different tasks related to administering intrusion policies. These subsections include instructions for how to access and interpret IPS events and include the following topics: Create Intrusion Policy, Intrusion Rule Actions, Adding and Modifying Intrusion Event Thresholds, Intrusion Rules Editor, Intrusion Rules Import, Stateful Session Behaviors, Portscan Detection, Rate-Based Attack Prevention, Specific Attacks, Checksum Verification and Portscan Event Packet View.

Component Testing Assurance Activities: The evaluator shall devise tests that demonstrate that IPS data (generated as defined in FAU_GEN.1/IPS) can be interpreted by authorized administrators from the TOE's management interface.

Throughout the course of all IPS testing the events were logged and stored on the FMC devices. All IPS event data is viewed from an FMC device and is stored in the same manner regardless of FTD devices. The evaluator logged onto the FMC device as an authorized administrator and was able to view details of the IPS events. This testing was performed through the FMC device's trusted path on the management interface.

2.1.8 Restricted Audit Review (IPS10:FAU_SAR.2)

2.1.8.1 IPS10:FAU_SAR.2.1

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 identifies what group of users is considered to be 'administrators' for the purpose of being granted access to view IPS data.

Section 6.1 (FMT_SMF.1/IPS) in the ST states that the Administrators can deploy intrusion policy with intrusion rules to any interface. An interface, however, can only have one policy applied to that interface. The Administrators can also import vendor-defined signatures from Cisco, create their own intrusion rules, create rules to define which traffic is inspected and analyzed, enable anomaly rules/detections, modify thresholds and threshold duration, and configure good-list/bad-list. The IPS Analysts (Intrusion Admins) can create, modify, or delete intrusion policies but only the IPS Administrators can deploy the policies. Here are the security roles in addition to the all-powerful "Administrator" role.

• "IPS Administrator" (or Administrator): Have all privileges and access

• "IPS Analyst" (or Intrusion Admin): Have all access to intrusion policies, IPS policies and network analysis privileges but cannot deploy policies

• Access Admin: Have all access to Access Control policies but cannot deploy policies

• Discovery Admin: Have all access to network discovery, application detection, and correlation features but cannot deploy policies

• Security Analyst: Have all access to security event analysis feature

The above defined user account types provide the user with a subset of permissions regarding IPS rules, with only the IPS administrator having access to everything.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to determine what actions are needed to grant or revoke a user's ability to view IPS data.

Section "User and Role Management" in the FTD Admin Guide states that the Administrator Role can use the web interface to view and manage user accounts on an FMC or a managed Device, including adding, modifying, and deleting accounts. User accounts without Administrator Role are restricted from accessing user management functions.

Section "Adding New User Accounts" in the FTD Admin Guide provides instructions for configuring user accounts and identifies the access roles that can be granted to the user including the "IPS Administrator" (or Administrator) role which has all privileges and access. The roles that can be selected are consistent with those identified in the ST: IPS Administrator/Administrator, IPS Analyst, Access Admin, Discovery Admin and Security Analyst.

Component Testing Assurance Activities: The evaluator shall log on to the TOE as various users to confirm that only those users that are intended to be granted read access to IPS data are able to view it.

This test was performed as part of IPS10:FAU_STG.1 where the evaluator logged on as an administrator who had permission to view the IPS data, and a discovery admin who did not have permissions to view the IPS data. The evaluator confirmed that accounts without permissions to view the IPS data could not view it.

2.1.9 Selectable Audit Review (IPS10:FAU_SAR.3)

2.1.9.1 IPS10:FAU_SAR.3.1

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 includes a description of how the TOE has the ability to apply filtering and sorting of IPS data using the parameters listed in the requirement.

Section 6.1 (FAU_SAR.3 [IPS]) in the ST states that the TOE will generate an event log for each intrusion event that occurs. Each event log will include a record of the date, time, type of exploit, and contextual information about the source of the attack and its target. For packet-based events, a copy of the packet or packets that triggered the event is also recorded. Managed Sensors will transmit their events to the FMC where the administrators can view the aggregated data and gain a greater understanding of the attacks against the entire network. Only Administrators and Intrusion Admins have access to the intrusion events. Administrators view the intrusion events via the web-based UI (Analysis > Intrusions > Events). This section further provides a list and description of the intrusion event information that can be viewed, searched, filtered, and sorted by the system. This includes the parameters listed in the requirement such as Application Risk, Destination IP, Source IP, Inline Result (Actions) and Signature ID. In addition, basic contents such as date, time, and event type can also be used to filter and sort.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance to ensure that the guidance itemizes all event types, as well as describes all attributes that are to be selectable in accordance with the requirement, to include those attributes listed in the assignment. The administrative guidance shall also contain instructions on how to set the pre- selection, as well as explain the syntax (if present) for multi-value pre-selection. The administrative guidance shall also identify those audit records that are always recorded, regardless of the selection criteria currently being enforced.

Section “Logs of Intrusion and Firewall Events” in the FTD Admin Guide indicates that the connection and intrusion events are generated by the “log” operation in a rule and provides a table identifying the information that is associated with each event. It further provides examples of events for access control rules, connection events and intrusion events.

Section “Management of Intrusion Events” in the FTD Admin Guide states that when the system identifies a possible intrusion, it generates an intrusion event, which is a record of the date, time, the type of exploit, and contextual information about the source of the attack and its target. For packet-based events, a copy of the packet or packets that triggered the event is also recorded. Managed Devices transmit their events to the Firepower Management Center (FMC) where the aggregated data can be viewed in order to gain a greater understanding of the attacks against network assets. A managed device can also be deployed as an inline, switched, or routed intrusion system, which allows the device to be configured to drop or replace packets that are known to be harmful.

Sections “Viewing Intrusion Events”, “Searching Intrusion Events” and “Sorting and filtering Intrusion Events” in the FTD Admin Guide provide instructions for viewing, searching and filtering intrusion event information. This section also provides a sample view of some intrusion events and a description of each field. These fields include all of the attributes listed in the requirement and the TSS.

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

Test 1: For each attribute listed in the requirement, the evaluator shall devise a test to show that selecting the attribute causes only audit events with that attribute (or those that are always recorded, as identified in the administrative guidance) to be recorded.

Test 2 [conditional]: If the TSF supports specification of more complex audit pre-selection criteria (e.g., multiple attributes, logical expressions using attributes) then the evaluator shall devise tests showing that this capability is correctly implemented. The evaluator shall also, in the test plan, provide a short narrative justifying the set of tests as representative and sufficient to exercise the capability.

Test 1: The evaluator utilized the TOE’s IPS event filtering abilities and was able to successfully filter and sort through logged IPS events.

Test 2: Not applicable. The TOE does not support preselection criteria.

2.1.10 Protected Audit Trail Storage (IPS Data) (IPS10:FAU_STG.1/IPS)

2.1.10.1 IPS10:FAU_STG.1.1/IPS

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.10.2 IPS10:FAU_STG.1.2/IPS

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 how IPS data is protected from unauthorized modification and deletion.

Section 6.1 (FAU_STG.1 [IPS]) in the ST states that only Administrators and Intrusion Admins have access to intrusion events. The intrusion events cannot be modified but they can be deleted by the Administrators or Intrusion Admins who have restricted access. When the intrusion events storage is full, the newest data will overwrite the oldest data.

Component Guidance Assurance Activities: The evaluator shall confirm the guidance documentation describes how to protect IPS data from unauthorized modification and deletion.

Section “Manage the FMC Audit Log and Syslog” in the FTD Admin Guide states that FMCs and managed Devices log read-only auditing information for user activity. Audit logs are presented in a standard event view that allows administrator to view, sort, and filter audit log messages based on any item in the audit view. Administrators can delete and report on audit information and can view detailed reports of the changes that users make.

Section "Viewing Intrusion Events" in the FTD Admin Guide states that only Administrators and Intrusion Admins have access to the intrusion events. The TOE protects IPS data based upon a user's role. Section "Adding New User Accounts" in the FTD Admin Guide identifies the available roles, and permissions for each role.

• "IPS Administrator" (or Administrator): Have all privileges and access.

• "IPS Analyst" (or Intrusion Admin): Have all access to intrusion policies, and network analysis privileges but cannot deploy policies

• Access Admin: Have all access to Access Control policies but cannot deploy policies

• Discovery Admin: Have all access to network discovery, application detection, and correlation features but cannot deploy policies

• Security Analyst: Have all access to security event analysis feature

Component Testing Assurance Activities: The evaluator shall devise tests that demonstrate that IPS data can be protected from unauthorized modification and deletion.

The evaluator logged into the TOE devices using different permission-based accounts and tested the ability to delete IPS data. Only administrators specified to have deletion privileges were able to delete IPS data.

2.1.11 Protected Audit Event Storage (NDcPP22e:FAU_STG_EXT.1)

2.1.11.1 NDcPP22e:FAU_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.11.2 NDcPP22e:FAU_STG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.11.3 NDcPP22e:FAU_STG_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 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 the amount of audit data that are stored locally; what happens when the local audit data store is full; and how these records are protected against unauthorized access.

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 the behavior 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.

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit information to an external IT entity can be done in real-time or periodically. In case the TOE does not perform transmission in real-time the evaluator needs to verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible acceptable frequency for the transfer of audit data.

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).

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 6.1 (FAU_STG_EXT.1) in the ST states that the TOE (FMC) includes an internal log database implementation that can be used to store and review audit records locally. However, the internal log only stores a default of 100,000 entries in the local database. When the audit log is full, the oldest audit records are overwritten by the newest audit records. In addition, the TOE (FMC) also includes a local syslog storage in /var/log/messages and these logs are viewable through the FMC Web UI. The contents are stored in flat files which are rotated automatically. Similar to the audit log, when the syslog is full, the oldest messages are overwritten by the newest one.

For audit log, the events are stored in partitioned event tables. The TOE will prune (i.e., delete) the oldest partition whenever the oldest partition can be pruned without dropping the number of events count below the configured event limit. Note this limit defaults to 10,000 if you set it any lower. For example, if you set the limit to 10,000 events, the events count may need to exceed 15,000 events before the oldest partition can be deleted. For the local syslog storage, the logs are stored in /var/log/messages and are rotated daily or when the log file size exceeds 25 MB. After the maximum number of backlog files is reached, the oldest is deleted and the numbers on the other backlogs file are incremented.

Modifications are not allowed by the interfaces and only authorized administrators can delete the audit logs.

To prevent the losing of critical audit records, the administrators can configure the system to transmit all the audit events (i.e., audit log and syslog) in real-time over a secure TLS connection or an IPsec connection (FXOS-only) to an external audit server in the operational environment. When an audit event is generated, it is sent to the local storage and external audit server simultaneously. This ensures that current audit events can be viewed locally while all events, new or old, are stored off-line as required by the NDcPP.

Messages generated by FTD, including FTD system messages, firewall events, and VPN events are stored locally on FTD and are immediately transmitted from FTD to an external syslog server. As mentioned in the preceding paragraph, the firewall, VPN and IPS events are directly sent to FMC for retention in the FMC databases via secure TLS channel (Note: The IPS events are not stored locally on FTD but are transmitted to an external syslog server via the FMC. IPS events generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC). If the connection between FTD and FMC is interrupted, the IPS messages are transmitted once connectivity is restored. As the system, firewall event and VPN event messages are generated by FTD, they are immediately transmitted from FTD to a remote syslog server and stored in a local buffer (buffer size configurable from 4096-

52428800 bytes) which overwrites old messages with new ones when storage limits are reached. The local logs are viewable from the FTD CLI by using “show logging”.

The local storage of audit events in FXOS (e.g., admin authentication, TLS/HTTPS session state, etc.) is viewable from the “fxos” shell (after using “connect fxos”) by using “show logging”. The storage limit of the local buffer is configurable via FXOS CLI with configurable size limit of 4096-4194304 bytes. This is a circular log (oldest records will be overwritten by new ones when the size limit is reached).

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 determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server. For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and 'cleared' periodically by sending the data to the audit server.

The evaluator shall also ensure that the guidance documentation describes all possible configuration options for FAU_STG_EXT.1.3 and the resulting behaviour of the TOE for each possible configuration. The description of possible configuration options and resulting behaviour shall correspond to those described in the TSS.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instruction for configuring the FMC and FTD to transmit audit records securely to an external audit server. This includes installing syslog-ng, version 3.7 or later. Rsyslog with stunnel can also be used. To securely transmit log messages to an audit server, Transport Layer Security (TLS) is used between the FMC and the syslog-ng server and between the FTD and the syslog-ng server. To securely send the logs to a trusted audit server, a signed audit client certificate must be imported and the communication channel must be configured with the audit server to use TLS.

Section “Configure Logging” in the FTD Admin Guide describes the two categories of audit messages on the FMC: the “System Log” stores syslog messages (for system level events, including CLI login/logout events); and the “Audit Log” stores messages as database records (for configuration changes via WebUI or CLI. The local storage of system (syslog) messages is not configurable.

Section “Manage the FMC Audit Log and Syslog” in the FTD Admin Guide indicates that audit messages can be configured to be transmitted directly to a remote syslog server, in which case each message will be simultaneously transmitted to the remote logging server as the message is written locally. This section also provides instructions for setting the default retention level for the Audit Event Database which is 100,000 and is configurable from 1-100,000. This is performed via System > Configuration > Database > Audit Event Database > Maximum Audit Events in the Web UI.

Section “Configure Logging” in the FTD Admin Guide indicates that for FTD once use of a remote audit server has been configured, messages will be simultaneously written locally and transmitted to the remote server. This section also provides instructions for configuring the FTD via the CLI or via the FMC Web UI to transmit audit messages to a remote server. Local storage on the FTD can be configured via the Logging Setup tab in the FMC Web UI. The size of the internal buffer can be specified between 4096 and 52428800 bytes. When the buffer fills up, it is overwritten.

Section “Manage the FMC Audit Log and Syslog” in the FTD Admin Guide states that messages generated by FTD, including FTD system messages, firewall events, and VPN events are stored locally on FTD and are immediately transmitted from FTD to an external syslog server. The firewall, VPN and IPS events are directly sent to FMC for retention in the FMC databases via secure TLS channel (Note: The IPS events are not stored locally on FTD but are transmitted to an external syslog server via the FMC. IPS events generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC). If the connection between FTD and FMC is interrupted, the IPS messages are transmitted once connectivity is restored. As the system, firewall event and VPN event messages are generated by FTD, they are immediately transmitted from FTD to a remote syslog server and stored in a local buffer (buffer size configurable from 4096-52428800 bytes) which overwrites old messages with new ones when storage limits are reached. The local logs are viewable from the FTD CLI shell by using “show logging”.

Section "Configure Secure Connection with Audit Server" of the FXOS Admin Guide states that System logging is a method of collecting messages from devices to a server running a syslog daemon. Logging to a central syslog server helps in aggregation of logs and alerts. By default, a syslog service accepts messages and stores them in the local files, or prints them according to a simple configuration file. This form of logging provides protected long-term storage for logs. Logs are useful both in routine troubleshooting and in incident handling. The syslog events are sent to the local store and syslog server simultaneously, if an external syslog server is configured. In the evaluation configuration, syslog traffic must be sent to the syslog server over IPsec.

To view the local syslog messages,

```
Firepower-chassis# connect fxos
```

```
Firepower-chassis(fxos)# show logging logfile
```

Section "Configure IPsec Secure Channel" of the FXOS Admin Guide provides instructions for configuring IPsec on FXOS to ensure that syslog traffic is secured in IPsec.

Section "Configure Secure Connection with Audit Server" of the FXOS Admin Guide states that syslog events are sent to the local store and syslog server simultaneously, if an external syslog server is configured.

According to step 9 in section "Configure Syslog via CLI" of FXOS Admin Guide, the administrator can specify the maximum file size, in bytes, before the system begins to write over the oldest messages with the newest ones (in the local buffer). This value can be specified up to 4MB.

Component Testing Assurance Activities: Testing of the trusted channel mechanism for audit will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall perform the following additional 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: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall perform operations that generate audit data until the local storage space is exceeded and verifies that the TOE complies with the behaviour defined in FAU_STG_EXT.1.3. Depending on the configuration this means that the evaluator has to check the content of the audit data when the audit data is just filled to the maximum and then verifies that

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

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

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

c) Test 3: If the TOE complies with FAU_STG_EXT.2/LocSpace the evaluator shall verify that the numbers provided by the TOE according to the selection for FAU_STG_EXT.2/LocSpace are correct when performing the tests for FAU_STG_EXT.1.3.

d) Test 4: For distributed TOEs, Test 1 defined above should be applicable to all TOE components that forward audit data to an external audit server. For the local storage according to FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3 the Test 2 specified above shall be applied to all TOE components that store audit data locally. For all TOE components that store audit data locally and comply with FAU_STG_EXT.2/LocSpace Test 3 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

Test 1: The successful establishment of the TLS syslog connection for all TOE devices was demonstrated in FTP_ITC.1. In each case the TOE initiated the connection without administrator intervention. The use of TLS 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 for FTD and FMC are stored locally in the filesystem. The evaluator copied random data to the log file until it exceeded limit specified in the administrative guidance. The evaluator observed that the audit data rolled over into a maximum of 10 messages.gz files. Once another log file was created past the limit of 10, the oldest one was overwritten. Additionally, the FMC includes an internal log database implementation that can be used to store audit records locally up to a configured limit before purging the oldest logs periodically. For testing purposes, the evaluator configured a limit of 5 records for local audit data and performed repeated administrative actions to generate audit data. The evaluator monitored the total number of audit records stored by the TOE before and after this limit was configured. After the new audit storage limit was configured and about 10 minutes had passed, the evaluator confirmed that the number of audits was purged down.

A similar process was used for FXOS which also stores logs locally in the file system. The evaluator issues a command on the CLI which shows the entire contents of the local log file. The evaluator waited for some time then issues the command again. When comparing the output of the commands, the evaluator found that the oldest entries in the log file had been overwritten by the newest entries.

Test 3: Not applicable. The TOE does not claim support for FAU_STG_EXT.2/LocSpace.

Test 4: Tests 1 and 2 cover all auditing scenarios, which includes components that store audit data locally and components that send audit data to an external audit server.

2.1.12 Protected Local Audit Event Storage for Distributed TOEs (NDcPP22e:FAU_STG_EXT.4)

2.1.12.1 NDcPP22e:FAU_STG_EXT.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator examines the TSS to confirm that it describes which TOE components store their security audit events locally and which send their security audit events to other TOE components for local storage. For the latter, the target TOE component(s) which store security audit events for other TOE components shall be identified. For every sending TOE component the corresponding receiving TOE component(s) need to be identified. For every transfer of audit information between TOE components it shall be described how the data is secured during transfer according to FTP_ITC.1 or FPT_ITT.1.

For each TOE component which does not store audit events locally by itself, the evaluator confirms that the TSS describes how the audit information is buffered before sending to another TOE component for local storage.

Section 6.1 (FAU_STG_EXT.4) in the ST indicates that all TOE components store audit events locally and can be configured to export syslog records to an administrator-specified, external syslog server. FMC transmits syslog over TLS and FTD transmits syslog over TLS and IPsec. FXOS transmits syslog over IPsec.

All audit records are stored locally and when the local storage is full, the newest data will overwrite the oldest data. On FMC, log messages (those generated locally and those forwarded from FTD) are stored locally on FMC in a database. Messages generated by FTD, including FTD system messages, firewall events, and VPN events are stored locally on FTD and are immediately transmitted from FTD to an external syslog server. The firewall, VPN and IPS events are directly sent to FMC for retention in the FMC databases via secure TLS channel (Note: The IPS events are not stored locally on FTD but are transmitted to an external syslog server via the FMC. IPS events generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC). If the connection between FTD and FMC is interrupted, the IPS messages are transmitted once connectivity is restored. As the system, firewall event and VPN event messages are generated by FTD, they are immediately transmitted from FTD to a remote syslog server and stored in a local buffer (buffer size configurable from 4096-52428800 bytes) which overwrites old messages with new ones when storage limits are reached. The local logs are viewable from the FTD CLI shell by using “show logging”.

The local storage of audit events in FXOS (e.g., admin authentication, TLS/HTTPS session state, etc.) is viewable from the “fxos” shell (after using “connect fxos”) by using “show logging”. The storage limit of the local buffer is configurable via FXOS CLI with configurable size limit of 4096-4194304 bytes. This is a circular log (oldest records will be overwritten by new ones when the size limit is reached).

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to ensure that it describes how the link between different TOE components is established if audit data is exchanged between TOE components for local storage. The guidance documentation shall describe all possible configuration options for local storage of audit data and provide all instructions how to perform the related configuration of the TOE components.

The evaluator shall also ensure that the guidance documentation describes for every TOE component which does not store audit information locally how audit information is buffered before transmission to other TOE components.

See the FCO_CPC_EXT.1 guidance assurance activities for a description of how the link between the FMC and FTD is established.

Section "Configure Logging" in the FTD Admin Guide states that Firewall (Access Control Policy) messages can be viewed in the local logging buffer of FTD using the command "show logging". These messages are sent from the FTD TLS client to an external syslog server. In addition, these messages can optionally be configured to also be sent over TLS from FTD to FMC where they would be viewable in FMC as they are stored in the connection database. Instructions for enabling logging to FMC are provided in a sub-section which describes the "Enable Logging to FMC" checkbox in the Logging Setup tab.

Section "Configure Logging" in the FTD Admin Guide states that VPN messages are sent from the FTD TLS client to an external syslog server. In addition, these messages can optionally be configured to be sent over TLS from FTD to FMC where they would be viewable in FMC via System > Monitoring > Syslog. Instructions for enabling logging to FMC are provided in a sub-section which describes the "Enable Logging to FMC" checkbox in the Logging Setup tab.

Section "Configure Logging" in the FTD Admin Guide states that IPS messages are automatically transmitted over TLS by FTD to FMC for storage, and are viewable via the "Audit Log" within FMC. IPS messages generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC, so if the connection from FTD to FMC is interrupted the IPS messages will be transmitted once connectivity is restored.

FXOS is logically distinct from the FTD and FMC ITT distributed-TOE communications. In other words, it does not transmit audit data to any other TOE component, only to an external syslog server. However, section "Configure Secure Connection with Audit Server" in the FXOS Admin Guide contains instructions for configuring the storage of local audits.

Component Testing Assurance Activities: For at least one of each type of distributed TOE components (sensors, central nodes, etc.), the following tests shall be performed using distributed TOEs.

Test 1: For each type of TOE component, the evaluator shall perform a representative subset of auditable actions and ensure that these actions cause the generation of appropriately formed audit records. Generation of such records can be observed directly on the distributed TOE component (if there is appropriate interface), or indirectly after transmission to a central location.

Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

While performing these tests, the evaluator shall verify that the TOE behaviour observed during testing is consistent with the descriptions provided in the TSS and the Guidance Documentation. Depending on the TOE configuration, there might be a large number of different possible configurations. In such cases, it is acceptable to perform subset testing, accompanied by an equivalency argument describing the evaluator's sampling methodology.

Test 1: Each TOE component generates its own set of audit events. Throughout testing, a subset of audits was captured for each device and a table was created to show that all expected audits were captured. Refer to NDcPP22e:FAU_GEN.1 for more audit event details. The audits were collected for each component from the remote syslog server running rsyslog version 8.16.0. The TLS channel transporting those audits was demonstrated in NDcPP22e:FTP_ITC.1-t4.

Test 2: The successful establishing of the TLS syslog connection for all devices, including both FTD and FMC devices, was demonstrated in FTP_ITC.1. In each case the TOE initiated the connection without administrator intervention. The use of TLS ensured 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 3: The FTD devices send the firewall, VPN and IPS events to the FMC for retention in the FMC databases via secure TLS channel. In NDcPP22e:FPT_ITT.1_t3, packet captures confirmed that the TLS handshake in the ITT channel completed successfully, and application data is seen between the FMC1600/FMCv and the managed FTD. The TLS channel ensures no data is viewed in cleartext. IPS10:FAU_SAR.1-t1 demonstrates an audit that was successfully transmitted from the FTD to be viewed on the FMC.

The evaluator found the TOE behavior was consistent with the descriptions provided in the TSS and AGD.

2.1.13 Protected Remote Audit Event Storage for Distributed TOEs (NDcPP22e:FAU_STG_EXT.5)

2.1.13.1 NDcPP22e:FAU_STG_EXT.5.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator examines the TSS to confirm that it describes which TOE components store their security audit events locally and which send their security audit events to other TOE components for local storage. For the latter, the target TOE component(s) which store security audit events for other TOE components shall be identified. For every sending TOE component the corresponding receiving TOE component(s) need to be identified. For every transfer of audit information between TOE components it shall be described how the data is secured during transfer according to FTP_ITC.1 or FPT_ITT.1. For each TOE component which does not store audit events locally by itself, the evaluator confirms that the TSS describes how the audit information is buffered before sending to another TOE component for local storage.

See FAU_STG_EXT.4 where this activity has already been performed.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to ensure that it describes how the link between different TOE components is established if audit data is exchanged between TOE components for local storage. The guidance documentation shall describe all possible configuration options for local storage of audit data and provide all instructions how to perform the related configuration of the TOE components. The evaluator shall also ensure that the guidance documentation describes for every TOE component which does not store audit information locally how audit information is buffered before transmission to other TOE components.

See the FCO_CPC_EXT.1 guidance assurance activities for a description of how the link between the FMC and FTD is established.

See the FAU_STG_EXT.1 guidance assurance activities for a description of how the local audit storage is configured for FTD and FMC.

Section “Configure Logging” in the FTD Admin Guide states that Firewall (Access Control Policy) messages can be viewed in the local logging buffer of FTD using the command “show logging”. These messages are sent from the FTD TLS client to an external syslog server. In addition, these messages can optionally be configured to also be sent over TLS from FTD to FMC where they would be viewable in FMC as they are stored in the connection database. Instructions for enabling logging to FMC are provided in a sub-section which describes the “Enable Logging to FMC” checkbox in the Logging Setup tab.

Section “Configure Logging” in the FTD Admin Guide states that VPN messages are sent from the FTD TLS client to an external syslog server. In addition, these messages can optionally be configured to be sent over TLS from FTD to FMC where they would be viewable in FMC via System > Monitoring > Syslog. Instructions for enabling logging to FMC are provided in a sub-section which describes the “Enable Logging to FMC” checkbox in the Logging Setup tab.

Section “Configure Logging” in the FTD Admin Guide states that IPS messages are automatically transmitted over TLS by FTD to FMC for storage, and are viewable via the “Audit Log” within FMC. IPS messages generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC, so if the connection from FTD to FMC is interrupted the IPS messages will be transmitted once connectivity is restored. Section “Manage the FMC Audit Log and Syslog” in the FTD Admin Guide states that the audit records are stored locally and when the local storage is full, the newest data will overwrite the oldest data. On FMC, log messages (those generated locally and those forwarded from FTD) are stored locally on FMC in a database. Messages generated by FTD, including FTD system messages, firewall events, and VPN events are stored locally on FTD and are immediately transmitted from FTD to an external syslog server. The firewall, VPN and IPS events are directly sent to FMC for retention in the FMC databases via secure TLS channel (Note: The IPS events are not stored locally on FTD but are transmitted to an external syslog server via the FMC. IPS events generated on FTD are temporarily stored locally on FTD in a database prior to transmission to FMC). If the connection between FTD and FMC is interrupted, the IPS messages are transmitted once connectivity is restored. As the system, firewall event and VPN event messages are generated by FTD, they are immediately transmitted from FTD to a remote syslog server and stored in a local buffer (buffer size configurable from 4096-52428800 bytes) which overwrites old messages with new ones when storage limits are reached. The local logs are viewable from the FTD CLI shell by using "show logging".

FXOS is logically distinct from the FTD and FMC ITT distributed-TOE communications. In other words, it does not transmit audit data to any other TOE component, only to an external syslog server. However, section "Configure Secure Connection with Audit Server" in the FXOS Admin Guide contains instructions for configuring the storage of local audits.

Component Testing Assurance Activities: For at least one of each type of distributed TOE components (sensors, central nodes, etc.), the following tests shall be performed using distributed TOEs.

Test 1: For each type of TOE component, the evaluator shall perform a representative subset of auditable actions and ensure that these actions cause the generation of appropriately formed audit records. Generation of such records can be observed directly on the distributed TOE component (if there is appropriate interface), or indirectly after transmission to a central location.

Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

While performing these tests, the evaluator shall verify that the TOE behaviour observed during testing is consistent with the descriptions provided in the TSS and the Guidance Documentation. Depending on the TOE configuration, there might be a large number of different possible configurations. In such cases, it is acceptable to perform subset testing, accompanied by an equivalency argument describing the evaluator's sampling methodology.

Test 1: The results for NDcPP22e:FAU_GEN.1, demonstrate that appropriate audit records are recorded for each TOE component.

Test 2: The results for NDcPP22e:FTP_ITC.1 test 1, demonstrate that each TOE component established a successful connection to the external audit log server in order to pass audit events in a secure manner. Through packet captures, the evaluator verified that no audit data was transmitted in cleartext.

Test 3: The packet captures in NDcPP22e:FPT_ITT.1_t3, demonstrate that the TLS handshake in the ITT channel completed successfully, and application data is seen between the FMC1600/FMCv and the managed device.

The evaluator found the TOE behavior was consistent with the descriptions provided in the TSS and AGD.

2.2 Communication (FCO)

2.2.1 Component Registration Channel Definition (NDcPP22e:FCO_CPC_EXT.1)

2.2.1.1 NDcPP22e:FCO_CPC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.2 NDcPP22e:FCO_CPC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.3 NDcPP22e:FCO_CPC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below. In carrying out these activities the evaluator shall determine answers to the following questions based on a combination of documentation analysis and testing (possibly also using input from carrying out the Evaluation Activities for the relevant registration channel, such as FTP_TRP.1(2)/Join), and shall report the answers.

Questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities:

a) What stops [The intent of the phrasing 'what stops...' as opposed to 'what secures...' is for the evaluator to pursue the answer to its lowest level of dependency, i.e. a level at which the security can clearly be seen to depend on things that are under appropriate control. For example, a channel may be protected by a public key that is provided to the relying party in a self-signed certificate. This enables cryptographic mechanisms to be applied to provide authentication (and therefore invites an answer that 'the check on the public key certificate secures...'), but does not ultimately stop an attacker from apparently authenticating because the attacker can produce their own self-signed certificate. The question 'what stops an unauthorised component from successfully communicating...' focuses attention on what an attacker needs to do, and therefore pushes the answer down to the level of whether a self-signed certificate could be produced by an attacker. Similarly a well-known key, or a key that is common to a type of device rather than an individual device, may be used in a confidentiality mechanism but does not provide confidentiality because an attacker can find the well-known key or obtain his own instance of a device containing the non-unique key.] a component from successfully communicating with TOE components (in a way that enables it to participate as part of the TOE) before it has properly authenticated and joined the TOE?

b) What is the enablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

1) What stops anybody other than a Security Administrator from carrying out this step?

2) How does the Security Administrator know that they are enabling the intended component to join? (Identification of the joiner might be part of the enablement action itself or might be part of secure channel establishment, but it must prevent unintended joining of components)

c) What stops a component successfully joining if the Security Administrator has not carried out the enablement step; or, equivalently, how does the TOE ensure that an action by an authentic Security Administrator is required before a component can successfully join?

d) What stops a component from carrying out the registration process over a different, insecure channel?

e) If the FTP_TRP.1(2)/Join channel type is selected in FCO_CPC_EXT.1.2 then how do the registration process and its secure channel ensure that the data is protected from disclosure and provides detection of modification?

f) Where the registration channel does not rely on protection of the registration environment, does the registration channel provide a sufficient level of protection (especially with regard to confidentiality) for the data that passes over it?

g) Where the registration channel is subsequently used for normal internal communication between TOE components (i.e. after the joiner has completed registration), do any of the authentication or encryption features of the registration channel result in use of a channel that has weaker protection than the normal FPT_ITT.1 requirements for such a channel?

h) What is the disablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

i) What stops a component successfully communicating with other TOE components if the Security Administrator has carried out the disablement step?

The evaluator shall examine the TSS to confirm that it:

a) Describes the method by which a Security Administrator enables and disables communications between pairs of TOE components.

b) Describes the relevant details according to the type of channel in the main selection made in FCO_CPC_EXT.1.2:

- First type: the TSS identifies the relevant SFR iteration that specifies the channel used

- Second type: the TSS (with support from the operational guidance if selected in FTP_TRP.1.3/Join) describes details of the channel and the mechanisms that it uses (and describes how the process ensures that the key is unique to the pair of components) - see also the Evaluation Activities for FTP_TRP.1(2)/Join.

The evaluator shall confirm that if any aspects of the registration channel are identified as not meeting FTP_ITC.1 or FPT_ITT.1, then the ST has also selected the FTP_TRP.1(2)/Join option in the main selection in FCO_CPC_EXT.1.2.

Section 6.1 (FCO_CPC_EXT.1) in the ST states that in order for TOE components to communicate as part of a distributed TOE System, they must successfully complete a registration process. Each TOE component comes with a manufacture's TLS certificate. To start the registration process, the administrator must enable or register the TOE components. On the FMC, the administrator must go to Device Management UI and click on "Add Device". At the same time, the administrator must go to the FTD CLI or GUI, and click or enter "Configure Manager Add". The administrator must specify the peer hostname or IP address and the registration key used for the initial authentication. During the registration process, the manufacture's TLS certificates are used to setup the initial TLS channel on the internal trusted management network. If the authentication succeeded, the resident CA on the FMC will sign and issue a TLS certificate along with the private key to the FTD which will be used for subsequent TLS channel. To disable or de-register the FTD, the administrator must initiate a "Delete Device" on the FMC Device Management UI and then perform a "Configure Manager Delete" action on the CLI of the FTD. This will destroy (i.e., zeroize) the TLS certificate and private key. Once this has occurred, no farther communication can happen without another registration process.

Component Guidance Assurance Activities: (Note: paragraph 274 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

The evaluator shall examine the guidance documentation to confirm that it contains instructions for enabling and disabling communications with any individual component of a distributed TOE. The evaluator shall confirm that the method of disabling is such that all other components can be prevented from communicating with the component that is being removed from the TOE (preventing the remaining components from either attempting to initiate communications to the disabled component, or from responding to communications from the disabled component).

The evaluator shall examine the guidance documentation to confirm that it includes recovery instructions should a connection be unintentionally broken during the registration process.

If the TOE uses a registration channel for registering components to the TOE (i.e. where the ST author uses the FTP_ITC.1/FPT_ITT.1 or FTP_TRP.1(2)/Join channel types in the main selection for FCO_CPC_EXT.1.2) then the evaluator shall examine the Preparative Procedures to confirm that they:

a) describe the security characteristics of the registration channel (e.g. the protocol, keys and authentication data on which it is based) and shall highlight any aspects which do not meet the requirements for a steady-state inter-component channel (as in FTP_ITC.1 or FPT_ITT.1)

b) identify any dependencies between the configuration of the registration channel and the security of the subsequent inter-component communications (e.g. where AES-256 inter-component communications depend on transmitting 256 bit keys between components and therefore rely on the registration channel being configured to use an equivalent key length)

c) identify any aspects of the channel can be modified by the operational environment in order to improve the channel security, and shall describe how this modification can be achieved (e.g. generating a new key pair, or replacing a default public key certificate).

As background for the examination of the registration channel description, it is noted that the requirements above are intended to ensure that administrators can make an accurate judgement of any risks that arise from the default registration process. Examples would be the use of self-signed certificates (i.e. certificates that are not chained to an external or local Certification Authority), manufacturer-issued certificates (where control over aspects such as revocation, or which devices are issued with recognised certificates, is outside the control of the operational environment), use of generic/non-unique keys (e.g. where the same key is present on more than one instance of a device), or well-known keys (i.e. where the confidentiality of the keys is not intended to be strongly protected - note that this need not mean there is a positive action or intention to publicise the keys).

In the case of a distributed TOE for which the ST author uses the FTP_TRP.1(2)/Join channel type in the main selection for FCO_CPC_EXT.1.2 and the TOE relies on the operational environment to provide security for some aspects of the registration channel security then there are additional requirements on the Preparative Procedures as described in section 3.5.1.2.

Section "Device Registration" of the FTD Admin Guide provides instructions on how to register an FTD device with the FMC, as specified in the ST. Network settings must be correctly configured on the FTD device and the management network should be an internal, trusted network separated physically or logically from the monitored network.

In order for the FMC and FTD to communicate, they must successfully complete a registration process, which requires administrative actions on the FMC and corresponding administrative actions on the FTD. The administrative actions on FMC and FTD require the administrator to input a "registration key" that the two devices will use to authenticate their initial TLS communications. During the registration process, the FMC and FTD confirm they have a matching registration key, and use their initial self-signed TLS certificates to uniquely identify themselves to each other (each device certificate signed by FMC, including its own, contains a unique identifier stored as an 'id-at-title' attribute, which FMC and FTD each as the unique reference identifier for each other). If the authentication succeeds, the local CA within the FMC will sign and issue a new TLS certificate for the FTD and send (over the existing TLS session) the FTD's new identity certificate and associated keys, and the FMC's root CA cert, and the FMC's root CA certificate and the device certificates which it signed will be used to authenticate all subsequent TLS sessions between the two devices.

If device registration fails due to mismatched registration keys, or incorrect IP address or hostname, the information on the FMC and/or FTD should be corrected and registration should be reinitiated from the FMC. If the connection between FMC and FTD is broken during device registration, the FMC and FTD will continue to attempt to reconnect and retry registration for up to two minutes. If the registration has not completed within two minutes, connectivity should be restored between the FMC and FTD and the registration should be reinitiated from the FMC.

The communication between the FMC and FTD is protected by TLSv1.2. TLS provides authentication, key exchange, encryption and integrity protection of all data transmitted between the TOE components. TLS session resumption is not supported in case the TLS connection between the TOE components is unintentionally broken. If connectivity is lost between FMC and FTD after device registration each endpoint will automatically attempt to re-initiate connection to the other until connectivity is restored, no administrative action is required other than resolving any connectivity issues in the networks between the FMC and FTD. The current status of each device can be viewed on the Device Management page (Devices > Device Management) where an icon indicates the current status (error, critical, warning, normal/recovered, or disabled). Detailed health conditions can be viewed on the Health Monitor page (System > Health > Monitor). The date and time each FTD was last seen by FMC can be viewed for each device individually by checking the Last Contacted timestamp under the status icon (Devices > Device Management > edit any device > view the Device tab > view the Management section). The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST. Sub-section "Device Registration on FTD" in the FTD Admin Guide provides the specific instructions for device registration on the FTD including the "configure manager delete" command which should be used to de-register a manager after the device is first deleted from FMC. Sub-section "Device Registration on FMC" in the FTD Admin Guide provides the specific instructions for device registration on the FMC including a note indicating that clicking on the trash can icon next to the Device will deregister and remove the device.

Component Testing Assurance Activities: (Note: paragraph 274 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

The evaluator shall carry out the following tests:

a) Test 1.1: the evaluator shall confirm that an IT entity that is not currently a member of the distributed TOE cannot communicate with any component of the TOE until the non-member entity is enabled by a Security Administrator for each of the non-equivalent TOE components [An 'equivalent TOE component' is a type of distributed TOE component that exhibits the same security characteristics, behaviour and role in the TSF as some other TOE component. In principle a distributed TOE could operate with only one instance of each equivalent TOE component, although the minimum configuration of the distributed TOE may include more than one instance (see discussion of the minimum configuration of a distributed TOE, in section B.4). In practice a deployment of the TOE may include more than one instance of some equivalent TOE components for practical reasons, such as performance or the need to have separate instances for separate subnets or VLANs.] that it is required to communicate with (non-equivalent TOE components are as defined in the minimum configuration for the distributed TOE)

b) Test 1.2: the evaluator shall confirm that after enablement, an IT entity can communicate only with the components that it has been enabled for. This includes testing that the enabled communication is successful for the enabled component pair, and that communication remains unsuccessful with any other component for which communication has not been explicitly enabled.

Some TOEs may set up the registration channel before the enablement step is carried out, but in such a case the channel must not allow communications until after the enablement step has been completed.

The evaluator shall repeat Tests 1.1 and 1.2 for each different type of enablement process that can be used in the TOE.

c) Test 2: The evaluator shall separately disable each TOE component in turn and ensure that the other TOE components cannot then communicate with the disabled component, whether by attempting to initiate communications with the disabled component or by responding to communication attempts from the disabled component.

d) Test 3: The evaluator shall carry out the following tests according to those that apply to the values of the main (outer) selection made in the ST for FCO_CPC_EXT.1.2.

1) If the ST uses the first type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_ITC.1 or FPT_ITT.1 according to the second selection - the evaluator shall ensure that the test coverage for these SFRs includes their use in the registration process.

2) If the ST uses the second type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_TRP.1(2)/Join.

3) If the ST uses the 'no channel' selection then no test is required.

e) Test 4: The evaluator shall perform one of the following tests, according to the TOE characteristics identified in its TSS and operational guidance:

1) If the registration channel is not subsequently used for inter-component communication, and in all cases where the second selection in FCO_CPC_EXT.1.2 is made (i.e. using FTP_TRP.1(2)/Join) then the evaluator shall confirm that the registration channel can no longer be used after the registration process has completed, by attempting to use the channel to communicate with each of the endpoints after registration has completed.

2) If the registration channel is subsequently used for intercomponent communication then the evaluator shall confirm that any aspects identified in the operational guidance as necessary to meet the requirements for a steady-state intercomponent channel (as in FTP_ITC.1 or FPT_ITT.1) can indeed be carried out (e.g. there might be a requirement to replace the default key pair and/or public key certificate).

f) Test 5: For each aspect of the security of the registration channel that operational guidance states can be modified by the operational environment in order to improve the channel security (cf. AGD_PRE.1 refinement item 2 in (cf. the requirements on Preparative Procedures in 3.5.1.2), the evaluator shall confirm, by following the procedure described in the operational guidance, that this modification can be successfully carried out.

Test 1: (1.1 and 1.2) The evaluator registered the FTD with the FMC for TOE distributed communication and verified that the communication was successful. This test was repeated with the FTD and the FMCv. For the next test (1.2), the evaluator registered the FTD device with the FMC. After the registration, the evaluator noted that only communications were allowed to flow between the two registered devices. The FMCv, which is the other FMC that the FTD device can pair with, was not able to communicate with or push policies to the FTD. The TOE includes only one enablement process which was tested.

Test 2: The evaluator verified that the FTD TOE was registered and that distributed communication was enabled with the FMC, the evaluator then unregistered the FTD TOE and verified that the device was no longer registered with the FTD device. The evaluator then verified that the TOE components could not communicate with the device after it was no longer registered.

Test 3 - This test is met by NDcPP22e:FPT_ITT.1 where the ITT channel was tested and shown to encrypt all ITT communication. The channel used in FPT_ITT.1 is the same channel that device enablement (registration) occurs on.

Test 4 - There are no necessary actions to meet the requirements for a steady-state inter-component channel.

Test 5 - No actions exist to improve channel security.

2.3 Cryptographic support (FCS)

2.3.1 Cryptographic Key Generation (NDcPP22e:FCS_CKM.1)

2.3.1.1 NDcPP22e: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) in the ST provides a table which identifies the usage for each scheme. The TOE uses a cryptographic module (i.e., Cisco FIPS Object Module) to provide supporting cryptographic functions. The TOE supports RSA, FFC, and ECDSA in the evaluated configuration. RSA and ECDSA digital signature are used in TLS connections and SSH connections (RSA only) and the TOE can be configured to use RSA and ECDSA to authenticate IPsec connections (RSA only for IPsec on FXOS).

Key generation for asymmetric keys on all models of the TOE implements ECDSA with NIST curve sizes P-256, P-384, and P-521 according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4, RSA with key sizes 2048 and 3072 bits according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 and FFC schemes using 'safe-prime' groups that meet RFC 3526 and RFC 7919. Asymmetric cryptographic keys used for IKE peer authentication are generated according to FIPS PUB 186-5, Appendix B.3 for RSA schemes and Appendix B.4 for ECDSA schemes.

Key establishment for asymmetric keys on the TOE implements RSA-based (RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447), ECDSA-based and DH-based key establishment schemes as specified in NIST SP 800-56A "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography". In addition, the TOE also supports DH group 14 key establishment scheme that meets standard RFC 3526, section 3 for interoperability. The TOE's software implementation uses the prime number and generator value specified in RFC 3526 Section 3 when generating parameters for the DH Group 14 key exchange.

The supported key sizes are specified in Section 7.4, Table 27 and Table 28 in the ST which provide the CAVP mapping.

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 "CC Mode and FIPS Mode" in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. There are additional features such as enabling the power-up integrity HMAC-SHA-512 self-test, enabling FIPS mode, and other TLS required checks such as the ones specified in section 6 of RFC 6125. CC Mode must be enabled in the evaluated configuration.

Section "Common Criteria (CC) Mode" in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS, SSH and IPsec algorithms to the ones specified in the ST.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

Section “Custom Web Server Certificate” in the FTD Admin Guide provides instructions for generating an HTTPS Server Certificate signing request and importing an HTTPS certificate on the FMC. Section “Configure SSH Public-Key authentication” in the FTD Admin Guide provides the steps for generating an SSH keypair to be used for SSH public key authentication for both the FMC and the FTD device. Section “Configure SSH Rekey Configuration” in the FTD Admin Guide provides instructions for configuring the SSH rekey parameters.

Section “PKI Infrastructure” in the Supplement describes the PKI framework for FTD IPsec communication and indicates that an RSA or ECDSA key pair can be generated and used for both signing and encryption, or separate keys can be generated for each purpose. TLS uses a key for encryption but not signing, however, IKE uses a key for signing but not encryption.

Section “Adding Certificate Enrollment Objects” in the Supplement provides steps for enrolling a certificate and specifying the certificate contents including the Key Type: RSA or ECDSA for FTD certificate-based authentication.

Section “FTD Remote Access VPN” and section “FTD site-to-site VPN” in the Supplement provide instructions for configuring IPsec and IKEv2 policies on FTD.

The section entitled “*Generate the SSH Host Key”* in the FXOS Admin Guide describes the generation of keys for SSH on FXOS interfaces. The section entitled “*Configure PKI”* and its subsections in FXOS Admin Guide contains subsections that explain how to generate keys for certificates used for FXOS interfaces using TLS and IPsec protocols.

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 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.

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

a) Random Primes:

- Provable primes

- Probable primes

b) Primes with Conditions:

- Primes p1, p2, q1, q2, p and q shall all be provable primes

- Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes

- Primes p1, p2, q1, q2, p and q shall all be probable primes

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall require the implementation under test (IUT) to generate 10 private/public key pairs. The private key shall be generated using an approved random bit generator (RBG). To determine correctness, the evaluator shall submit the generated key pairs to the public key verification (PKV) function of a known good implementation.

FIPS 186-4 Public Key Verification (PKV) Test

For each supported NIST curve, i.e., P-256, P-384 and P-521, the evaluator shall generate 10 private/public key pairs using the key generation function of a known good implementation and modify five of the public key values so that they are incorrect, leaving five values unchanged (i.e., correct). The evaluator shall obtain in response a set of 10 PASS/FAIL values.

Key Generation for Finite-Field Cryptography (FFC)

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.

(TD0580 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.2 Cryptographic Key Generation (for IKE Peer Authentication) (VPNGW13:FCS_CKM.1/IKE)

2.3.2.1 VPNGW13:FCS_CKM.1.1/IKE

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

- For each applicable section of Appendix B, any omission of functionality related to 'shall' or 'should' statements shall be described;

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

Section 6.1 (FCS_CKM.1/IKE) in the ST states that the TOE utilizes a cryptographic module (i.e., Cisco FIPS Object Module) to provide supporting cryptographic functions. The TOE supports RSA, FFC, and ECDSA in the evaluated configuration. The TOE can be configured to use RSA and ECDSA (FTD only) to authenticate IPsec connections. Key generation for asymmetric keys on all models of the TOE implements ECDSA with NIST curve sizes P-256, P-384, and P-521 according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 and RSA with key sizes 2048 and 3072 according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3. Asymmetric cryptographic keys used for IKE peer authentication are generated according to FIPS PUB 186-5, Appendix B.3 for RSA schemes and Appendix B.4 for ECDSA schemes. The TOE does not implement any functionality defined as 'shall not/should not' and does not omit any functionality related to 'shall/should' statements in Appendix B.3 and B.4. The ST does not define any TOE-specific extensions, processing or alternative implementations that are not included in the Appendices.

Key establishment for asymmetric keys on the TOE implements RSA-based (RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447), ECDSA-based and DH-based key establishment schemes as specified in NIST SP 800-56A "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography". In addition, the TOE also supports DH group 14 key establishment scheme that meets standard RFC 3526, section 3 for interoperability. The TOE's software implementation uses the prime number and generator value specified in RFC 3526 Section 3 when generating parameters for the DH Group 14 key exchange.

The supported key sizes are specified in Section 7.4, Table 27 and Table 28 in the ST which provide the CAVP mapping.

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

Section "Common Criteria (CC) Mode" in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS, SSH and IPsec algorithms to the ones specified in the ST.

Section "PKI Infrastructure" in the Supplement indicates that an RSA or ECDSA key pair can be generated and used for both signing and encryption, or separate keys can be generated for each purpose. TLS uses a key for encryption but not signing, however, IKE uses a key for signing but not encryption.

Section “Adding Certificate Enrollment Objects” in the Supplement provides steps for enrolling a certificate and specifying the certificate contents including the Key Type: RSA or ECDSA and the Key Size. This key generation process on FMC results in private keys stored in binary format on FMC and also on the FTD that will use the certificate associated with the generated keys. The private keys are stored in FMC and FTD in locations that are not accessible via any administrative interface, and only the unique identification of each key’s associated PKI certificate is visible (on FMC via Devices > Certificates; and on FTD by viewing output of “show crypto ca certificates”).

FXOS does not perform key generation for the purpose of supporting a VPN.

Component Testing Assurance Activities: For FFC Schemes using 'safe-prime' groups:

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

For all other selections:

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.3 Cryptographic Key Establishment (NDcPP22e:FCS_CKM.2)

2.3.3.1 NDcPP22e: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:

Scheme | SFR | Service

RSA | FCS_TLSS_EXT.1 | Administration

ECDH | FCS_SSHC_EXT.1 | Audit Server

ECDH | FCS_IPSEC_EXT.1 | Authentication Server

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.

(TD0580 applied)

Section 6.1 (FCS_CKM.2) in the ST provides a table which identifies the usage for each scheme and includes the scheme, SFR and service. Each FMC and each FTD appliance utilize a cryptographic module (i.e., Cisco FIPS Object Module) to provide supporting cryptographic functions. The TOE supports RSA, FFC, and ECDSA in the evaluated configuration. RSA and ECDSA digital signature are used in TLS connections and SSH connections (RSA only). The TOE can be configured to use RSA and ECDSA to authenticate IPsec connections.

For FXOS, this same section says that the FXOS TOE supports RSA, FFC, and ECDSA in the evaluated configuration. RSA and ECDSA digital signature are used in TLS connections and SSH connections (RSA only). The TOE can be configured to use RSA to authenticate IPsec connections. Key establishment for asymmetric keys on the TOE implements RSA-based (RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447), ECDSA-based and DH-based key establishment schemes as specified in NIST SP 800-56A "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography". In addition, the TOE also supports DH group 14 key establishment scheme that meets standard RFC 3526, section 3 for interoperability. The TOE's software implementation uses the prime number and generator value specified in RFC 3526 Section 3 when generating parameters for the DH Group 14 key exchange.

Key generation for asymmetric keys on all models of the TOE implements ECDSA with NIST curve sizes P-256, P-384, and P-521 according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 and RSA with key sizes 2048 and 3072 bits according to FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3. Asymmetric cryptographic keys used for IKE peer authentication are generated according to FIPS PUB 186-5, Appendix B.3 for RSA schemes and Appendix B.4 for ECDSA schemes.

AKey establishment for asymmetric keys on the TOE implements RSA-based (RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447), ECDSA-based and DH-based key establishment schemes as specified in NIST SP 800-56A "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography". In addition, the TOE also supports DH group 14 key establishment scheme that meets standard RFC 3526, section 3 for interoperability. The TOE's software implementation uses the prime number and generator value specified in RFC 3526 Section 3 when generating parameters for the DH Group 14 key exchange.

The supported key sizes are specified in Section 7.4, Table 31 and Table 32 in the ST which provide the CAVP mapping.

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

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. 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 will also individually inject errors in both parties' static public keys, both parties' ephemeral public keys and the TOE's static private key to assure the TOE detects errors in the public key validation function and/or the partial key validation function (in ECC only). At least two of the test vectors shall remain unmodified and therefore should result in valid key agreement results (they should pass).

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.

(TD0580 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

FFC safe primes and RSA were tested using a known good implementation.

For FFC safe primes, the test cases that demonstrate this are FCS_SSHS_EXT.1 for SSH, FCS_IPSEC_EXT.1 for IPSec, FCS_TLSS_EXT.1 and FCS_TLSC_EXT.1 for TLS.

For RSA key establishment, the test cases that demonstrate this are FCS_TLSS_EXT.1 and FCS_TLSC_EXT.1.

2.3.4 Cryptographic Key Destruction (NDcPP22e:FCS_CKM.4)

2.3.4.1 NDcPP22e: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 examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted for2). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

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 examines the TSS to ensure that it describes how that pattern is obtained and used, and that this justifies the claim that the pattern does not contain any CSPs.

Section 6.1 (FCS_CKM.4) in the ST indicates that the TOE is designed to zeroize secret and private keys when they are no longer required by the TOE. The secret keys used for symmetric encryption, private keys, and CSPs used to generate keys, are zeroized immediately after use (for IPsec VPN functions, within FTD only), or on system shutdown (for all other functions). For plaintext keys unrelated to IPsec VPN: the TOE destroys the reference to the keys stored in volatile memory directly followed by a request for garbage collection; the TOE destroys the abstraction that represents the key for keys stored in non-volatile storage the TSF.

For plaintext keys in FXOS the TOE destroys the reference to the keys stored in volatile memory directly followed by a request for garbage collection, and the TOE destroys the abstraction that represents the key for keys stored in non-volatile storage the TSF.

Section 7.3 in the ST provides a table which identifies the applicable secret and private keys, their storage location and how and when they are zeroized.

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 "VPN Basics" in the Supplement states that the secret keys used for symmetric encryption, private keys, and CSPs used to generate keys, are zeroized immediately after use (for IPsec VPN functions, within FTD only), or on system shutdown (for all other functions). For plaintext keys unrelated to IPsec VPN like the TLS and SSH related keys: the TOE destroys the reference to the keys stored in volatile memory directly followed by a request for garbage collection; the TOE destroys the abstraction that represents the key for keys stored in non-volatile storage the TSF.

Section "Generate the SSH Host Key" in the FXOS Admin Guide states that for plaintext keys in FXOS the TOE destroys the reference to the keys stored in volatile memory directly followed by a request for garbage collection, and the TOE destroys the abstraction that represents the key for keys stored in non-volatile storage the TSF. Using the "delete ssh-server host-key" command will zeroize (overwrite) the existing key. The "show ssh-server host-key" command can be used to show whether the key has been zeroized (overwritten), which occurs after using the "commit-buffer" command. No further steps are necessary to ensure they keys are destroyed in accordance with CC requirements. The secret keys used for symmetric encryption, private keys, and CSPs used to generate keys, are zeroized immediately after use (for IPsec VPN functions, within FTD only), or on system shutdown (for all other functions). For plaintext keys unrelated to IPsec VPN: the TOE destroys the reference to the keys stored in volatile memory directly followed by a request for garbage collection; the TOE destroys the abstraction that represents the key for keys stored in non-volatile storage the TSF. When an administrator using the FMC WebUI initiates deletion of keys (e.g. certificates and their associated keys), the actual key destruction is delayed at the physical layer until those instructions are pushed from FMC to FTD and implemented on FTD.

Both Guidance Documents do 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

2.3.5 Cryptographic Operation (AES Data Encryption/Decryption) (NDcPP22e:FCS_COP.1/DataEncryption)

2.3.5.1 NDcPP22e: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 6.1, FCS_COP.1/DataEncryption of the ST states that the FTD/FMC/FMCv TOE supports data encryption and decryption with AES using CBC and GCM modes using key sizes of 128 and 256. The same section also states that the FXOS TOE supports data encryption and decryption with AES using CBC and GCM modes with key sizes of 128, 192 (FXOS only) and 256 (CBC mode only).

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 “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. The SSH AES algorithms are specified consistent with the SSH requirements in the ST.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST. The SSH algorithms and TLS ciphersuites specified in this section are consistent with those claimed in the ST.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

Section “Deciding which Encryption Algorithm to Use” in the Supplement identifies the encryption algorithms that are available for IKEv2 and IPsec proposals where the algorithm is used by ESP. In the evaluated configuration only AES and AES-GCM are allowed.

The section entitled "*Configure SSH Access*" in the FXOS Admin Guide includes commands to allow an administrator to choose the encryption algorithms used by the FXOS SSH interface. The section entitled "*Configuring HTTPS*" in the FXOS Admin Guide includes a command that will limit the set of ciphersuites offered by the FXOS TLS Server interface, which defines the encryption algorithms supported for use with TLS. The section entitled "*Configure IPsec secure channel*" in the FXOS Admin Guide includes commands to limit the algorithms that the FXOS IPsec interface can negotiate to only those specified by the Security Target.

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 through FCS_SSH*_EXT.1.4. If CBC and/or GCM are selected in FCS_COP.1/DataEncryption, the test activities for those modes sufficiently demonstrate the correctness of the AES algorithm. If CTR is the only selection in FCS_COP.1/DataEncryption, the AES-CBC Known Answer Test, AES-GCM Known Answer Test, or the following test shall be performed (all of these tests demonstrate the correctness of the AES algorithm):

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.

2.3.6 Cryptographic Operation (Hash Algorithm) (NDcPP22e:FCS_COP.1/Hash)

2.3.6.1 NDcPP22e: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 6.1 (FCS_TLS*_EXT.*) in the ST indicates that the hash function is used in conjunction with TLSv1.2.

Section 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that SHA-based HMAC algorithms and hash functions are used in conjunction with IPSec using the ESP and IKE protocols.

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST indicates that SHA-based HMAC algorithms are used for integrity and authenticity in support of SSHv2.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST indicates that SHA-based HMAC algorithms are used for integrity of SSH sessions to FXOS.

Section 6.1 (FPT_APW_EXT.1) in the ST indicate that passwords are stored in hashed form using SHA-512.

Section 6.1 (FPT_TST_EXT.1) in the ST indicates that firmware integrity tests verify the digital signature of the code image using RSA-2048 with SHA-512.

Section 6.1 (FCS_COP.1/Hash) in the ST states that the algorithms supported for cryptographic hashing services are SHA-1, SHA-256, SHA-384 and SHA-512 with message digest sizes of 160, 256, 384 and 512 bits respectively.

The claims in FCS_COP.1/Hash are consistent with claims regarding hashing functions in other requirements.

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 “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. The SSH hash algorithms are specified consistent with the SSH requirements in the ST.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST. The SSH HMAC algorithms specified in this section are consistent with those claimed in the ST.

Section “Configure Authentication” in the FTD Admin Guide indicates that FMC passwords are stored hashed using Approved SHA-512 with a 32-bit salt value.

The section entitled "*Configure SSH via CLI*" in the FXOS Admin Guide describes commands to set the approved SSH cipher suites and hashing algorithms for the CC configuration.

Section "*Configuring HTTPS*" in [AGD-FXOS] the FXOS Admin Guide contends that once the device is in CC mode, no additional configuration is necessary to set the approved TLS ciphersuites and hashing algorithms.

Section "*Configure IPsec secure channel*" in the FXOS Admin Guide describes that once CC mode is enabled, FXOS will only support the listed hashing algorithms.

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 evaluators devise an input set consisting of m+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m bits. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Short Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode

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

Selected Long Messages Test - Byte-oriented Mode

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.7 Cryptographic Operation (Keyed Hash Algorithm) (NDcPP22e:FCS_COP.1/KeyedHash)

2.3.7.1 NDcPP22e: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 6.1 (FCS_COP.1/KeyedHash) in the ST indicates that the algorithms supported for keyed-hash message authentication by the TOE are HMAC-SHA-1 (block size - 512 bits), HMAC-SHA-256 (block size - 512 bits), HMAC-SHA-384 (block size - 1024 bits) and HMAC-SHA-512 (block size - 1024 bits) with key sizes 160, 256, 384 and 512 bits and message digest sizes of 160, 256, 384 and 512 bits respectively.

Section 7.4, Table 32 in the ST specifies that the TOE supports HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512.

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 “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. The SSH HMAC algorithms are specified consistent with the SSH requirements in the ST.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST. The SSH HMAC algorithms specified in this section are consistent with those claimed in the ST.

Section “Deciding Which Hash Algorithm to Use” of the Supplement specifies the hash algorithms available in the evaluated configuration which are consistent with those identified in the ST for IPSec.

Section “Configure IKEv2 Policy Object” in the Supplement provides instructions for configuring the integrity algorithm used in the IKE policy. Section “Configure IKEv2 IPsec Proposal Object” in Supplement provides instructions for configuring the ESP hash or integrity algorithm to use in the Proposal for authentication.

The section entitled "*Configure SSH via CLI*" in the FXOS Admin Guide describes commands to set the approved SSH cipher suites and hashing algorithms for the CC configuration.

Section "*Configuring HTTPS*" in [AGD-FXOS] the FXOS Admin Guide contends that once the device is in CC mode, no additional configuration is necessary to set the approved TLS ciphersuites and hashing algorithms.

Section "*Configure IPsec secure channel*" in the FXOS Admin Guide describes that once CC mode is enabled, FXOS will only support the listed hashing algorithms.

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.

2.3.8 Cryptographic Operation (Signature Generation and Verification) (NDcPP22e:FCS_COP.1/SigGen)

2.3.8.1 NDcPP22e: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 7.4, Table 32 in the ST indicates that for signature services the TOE supports RSA 2048 and 3072 and ECDSA curves P-256, P-384 and P-521 with key sizes 256, 384 and 521 bits.

For FTD and FMC, Section 6.1, FCS_COP.1/SigGen in the ST states that the TOE supports RSA, FFC, and ECDSA in the evaluated configuration. RSA and ECDSA digital signature are used in TLS connections and SSH connections (RSA only). The TOE can be configured to use RSA and ECDSA to authenticate IPsec connections.

For FXOS, the same section states that the TOE supports RSA, FFC, and ECDSA in the evaluated configuration. RSA and ECDSA digital signature are used in TLS connections and SSH connections (RSA only). The TOE can be configured to use RSA to authenticate IPsec connections.

Key generation for asymmetric keys on all models of the TOE implements ECDSA with NIST curve sizes P-256, P-384, and P-521 according to FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4 and RSA with key sizes 2048 and 3072 bits according to FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3. Asymmetric cryptographic keys used for IKE peer authentication are generated according to FIPS PUB 186-5, Appendix B.3 for RSA schemes and Appendix B.4 for ECDSA schemes. The TOE does not implement any functionality defined as ‘shall not/should not’ and does not omit any functionality related to ‘shall/should’ statements in Appendix B.3 and B.4. There are no TOE-specific extensions, processing or alternative implementations that are not included in the Appendices.

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 “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. There are additional features such as enabling the power-up integrity HMAC-SHA-512 self-test, enabling FIPS mode, and other TLS required checks such as the ones specified in section 6 of RFC 6125. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

Section “Custom Web Server Certificate” in the FTD Admin Guide provides instructions for generating an HTTPS Server Certificate signing request and importing an HTTPS certificate on the FMC.

Section “Configure SSH Public-Key authentication” in the FTD Admin Guide provides the steps for generating an SSH keypair to be used for SSH public key authentication for both the FMC and the FTD device.

Section “Configure SSH Rekey Configuration (optional)” in the FTD Admin Guide provides instructions for configuring the SSH rekey parameters.

Section “Deciding which Authentication Method to use” in the Supplement states that digital certificates use RSA or ECDSA key pairs to sign and encrypt IKE key management messages.

Section “PKI Infrastructure” in the Supplement indicates that an RSA or ECDSA key pair can be generated and used for both signing and encryption, or separate keys can be generated for each purpose. TLS uses a key for encryption but not signing, however, IKE uses a key for signing but not encryption.

Section “Adding Certificate Enrollment Objects” in the Supplement provides steps for enrolling a certificate and specifying the certificate contents including the Key Type: RSA or ECDSA.

Section “*Enable FIPS and CC Mode*” in the FXOS Admin Guide states that the TOE is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC and FIPS mode are enabled.

The section entitled "*Enable FIPS and CC Mode*" in the FXOS Admin Guide provides instructions to place the FXOS into the proper configuration to force the use of FIPS supported and PP allowed cryptography.

Component Testing Assurance Activities: ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate 10 1024-bit long messages and obtain for each message a public key and the resulting signature values R and S. To determine correctness, the evaluator shall use the signature verification function of a known good implementation.

ECDSA FIPS 186-4 Signature Verification Test

For each supported NIST curve (i.e., P-256, P-384 and P-521) and SHA function pair, the evaluator shall generate a set of 10 1024-bit message, public key and signature tuples and modify one of the values (message, public key or signature) in five of the 10 tuples. The evaluator shall obtain in response a set of 10 PASS/FAIL values.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator generates or obtains 10 messages for each modulus size/SHA combination supported by the TOE. The TOE generates and returns the corresponding signatures.

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

For each modulus size/hash algorithm selected, the evaluator generates a modulus and three associated key pairs, (d, e). Each private key d is used to sign six pseudorandom messages each of 1024 bits using a trusted reference implementation of the signature generation algorithm. Some of the public keys, e, messages, or signatures are altered so that signature verification should fail. For both the set of original messages and the set of altered messages: the modulus, hash algorithm, public key e values, messages, and signatures are forwarded to the TOE, which then attempts to verify the signatures and returns the verification results.

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.9 HTTPS Protocol (NDcPP22e:FCS_HTTPS_EXT.1)

2.3.9.1 NDcPP22e:FCS_HTTPS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.9.2 NDcPP22e:FCS_HTTPS_EXT.1.2

TSS Assurance Activities: None Defined_

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.9.3 NDcPP22e:FCS_HTTPS_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 examine the TSS and determine that enough detail is provided to explain how the implementation complies with RFC 2818.

Section 6.1 (FCS_HTTPS_EXT.1) in the ST indicates that the TOE implements HTTP over TLS (or HTTPS) to support remote administration on FMC and FXOS, TLS clients to support secure syslog connections, and TLS server and clients to support FPT_ITT.1. A remote administrator can connect over HTTPS to the TOE with their web browser. FTD supports two different TLS clients that send syslog messages to the external syslog server- FTD TLS client and FTD OS TLS Client. When CC mode is enabled, the TOE is restricted to only support TLSv1.2 for HTTPS sessions, client/server communications between TOE components and for syslog communications with AES 128- or 256-bit symmetric ciphers in CBC and GCM modes, in conjunction with SHA, RSA, and ECDSA. The TOE implements HTTPS according to RFC 2818 by using a TLSv1.2 session to secure the HTTP connection.

When CC mode is enabled, the TOE is restricted to only support TLSv1.2 for HTTPS sessions and client/server communications between TOE components. The TOE implements HTTPS according to RFC 2818 by using a TLSv1.2 session to secure the HTTP connection. A remote administrator can connect over HTTPS to the TOE with their web browser. If the TLS client does not support TLSv1.2, the TLS connection will fail and the administrators will not establish a HTTPS web-based session with the TOE.

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

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. There are additional features such as enabling the power-up integrity HMAC-SHA-512 self-test, enabling FIPS mode, and other TLS required checks such as the ones specified in section 6 of RFC 6125. CC Mode must be enabled in the evaluated configuration.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled. By default, port 443 (HTTPS), which is used to access the web interface, is enabled for any IP address. The access list is part of the system policy that the administrator configures. Instructions are provided for the administrator to create the access list and specify a list of IP addresses and then enable HTTPS or SSH for these IP addresses.

Section “Login Remotely to GUI Web Interface” in the FTD Admin Guide indicates that the FMC has a web interface and that users can access the web interface by logging into the appliance using a web browser. This section further provides instructions for the user to direct their web browser to https://hostname/ to connect to the hostname that corresponds with the name of the appliance or use the IP address of the appliance.

Sections “Generating an HTTPS Server Certificate Signing Request” and “Importing HTTPS Server Certificate” in the FTD Admin Guide provide instructions for generating a CSR for an HTTPS certificate and then importing it.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide describes how to configure the TOE for secure communication with a syslog server over TLS including generating a certificate request and importing a client certificate. This section indicates that if the system is configured to use CRLs, it will use the same CRL to validate both audit client certificates and HTTPS certificates to secure the HTTPS connection between the system and a web browser.

Configuration of FXOS HTTPS server is described in the section entitled "Configuring HTTPS" in the FXOS Admin Guide.

Component Testing Assurance Activities: This test is now performed as part of FIA_X509_EXT.1/Rev testing.

Tests are performed in conjunction with the TLS evaluation activities.

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

The TOE’s HTTPS channel does not support mutual authentication and therefore FIA_X509_EXT.1 is not applicable to HTTPS in this evaluation. The testing of the HTTPS channel was demonstrated in FCS_TLSS_EXT.1 and FTP_TRP.1.

2.3.10 IPsec Protocol - Per TD0800 (NDcPP22e:FCS_IPSEC_EXT.1)

2.3.10.1 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that a crypto map (the Security Policy Definition) set can contain multiple entries, each with a different access list. The crypto map entries are searched in a top-down sequence - the TOE attempts to match the packet to the crypto access control list (ACL) specified in that entry. The crypto ACL can specify a single address or a range of addresses and the crypto map can be applied to an inbound interface or an outbound interface. When a packet matches a permit entry in a particular access list, the method of security in the corresponding crypto map of that interface is applied. If the crypto map entry is tagged as “ipsecisakmp”, IPsec is triggered. The traffic matching the permit crypto ACLs would then flow through the IPsec tunnel and be classified as PROTECTED. Traffic that does not match a permit crypto ACL or match a deny crypto ACL in the crypto map, but is permitted by other ACLs on the interface is allowed to BYPASS the tunnel. Traffic that does not match a permit crypto ACL or match a deny crypto ACL in the crypto map, and is also blocked by other non-crypto ACLs on the interface would be DISCARDED.

In FXOS, the SPDs are pretty simple because FXOS is not operating as a VPN gateway, and the SPDs are just based on IP addresses, so the type of traffic being tunneled (e.g. syslog) is irrelevant to the tunneling decisions.

- The local-addr is the local management IP.
- The remote-addr is the IP of the IPsec peer (in tunnel mode or transport mode).
- A remote-subnet is applicable only in tunnel mode, and defines the subnet that would be reachable beyond the remote-addr.
- Outbound traffic will be **encrypted** when the source address is local-addr, ***and***:
    - the destination address is the remote-addr (in tunnel or transport mode); ***or***
    - the destination address is on the remote-subnet (in tunnel mode).
- Outbound traffic will **bypass** the tunnel if:

  - the destination address is *__not__* the remote-addr; *__and__*
  - the destination address is *__not__* on the remote-subnet.
- Inbound traffic will be **dropped** if:
  - the source address (prior to decryption) is on the remote-subnet (in tunnel mode); *__or__*
  - the source address is the remote-address, *__and__* the packets are *__not__* IKE or ESP.

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 “Configure Access Control Policies” in the FTD Admin Guide explains and provides instructions for how an administrator can construct entries for the SPD. It explains that an access control policy determines how the system handles traffic on the monitored network. Administrators can configure one or more access control policies, which they can then apply to one or more managed Devices. Each Device can have only one applied policy. Access control rules can be added to a policy to provide granular control of how traffic is handled and logged. An access control policy and all rules under the policy are associated to an interface. For each rule, the administrator can specify a rule action, that is, whether to trust, block, or inspect matching traffic with an intrusion policy. Each rule contains a set of conditions that identify the specific traffic you want to control. Rules can be simple or complex, matching traffic by any combination of security zone, IP address, application, protocols, ports, etc. The system matches traffic to access control rules in order; the first matched rule handles the traffic. The default access control policy blocks all traffic from the network.

Section “IPsec” in the Supplement states that an IPsec Proposal policy defines the settings required for IPsec tunnels. An IPsec proposal is a collection of one or more crypto-maps that are applied to the VPN interfaces on the devices. A crypto-map combines all the components required to setup IPsec security associations, including IPsec rules, proposals, remote peers, and other parameters that are necessary to define an IPsec SA. When two peers try to establish an SA, they must have at least one compatible crypto-map entry. Section “Editing FTD Remote Access VPN Policy” in the Supplement provides instructions for configuring the crypto map.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide details FXOS IPSec configuration consistent with the TSS. This includes SPDs, ciphers, dh groups, hash configuration, key size strength enforcement, certificate trustpoint configuration, revocation, and session lifetime.

Testing Assurance Activities: The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

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

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

Test 1: The evaluator created rules for each type of SPD policy. The evaluator then verified the rules by establishing an IPsec tunnel and successfully establishing an administrator connection. It was observed that the Bypass rule for administrator traffic, the Permit rule for IPsec traffic and the Discard rule were all successfully enforced against the traffic going through the IPsec tunnel. The evaluator also confirmed that packets not matching a Permit, Bypass or Discard rule were ignored/dropped.

Test 2: The FTD uses access-lists in order to control the flow of traffic for SPDs. For conflicting entries and overlapping ranges, the results that demonstrate the FTD's SPDs are demonstrated in its firewall testing (FFW_RUL_EXT.1.8 and FFW_RUL_EXT.1.9). Inbound and outbound packets, as well as packets that either establish SAs or packets that are part of established SAs are tested in Test 1 as part of the Protect test case. The evaluator found this behavior to be consistent with the TSS and AGD. For FXOS, the SPD is hard-coded and its behavior is demonstrated in Test 1 above.

2.3.10.2 NDcPP22e: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 uses the guidance documentation to configure the TOE to carry out the following tests:

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

This test was performed as part of NDcPP22e: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.

2.3.10.3 NDcPP22e:FCS_IPSEC_EXT.1.3

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

Section 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that the TOE (FTD & FXOS) supports both transport and tunnel modes.

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

Section “Configuring FTD Site-to-site VPN” of the Supplement provides instructions for configuring a VPN connection including 3 different options for configuration which allows the VPN connection to be set to Tunnel Mode, Transport Preferred, or Transport Required.

Configuring IPsec Secure Channel in the FXOS Admin Guide details FXOS IPSec tunnel and transport configuration.

· **IPsec Mode**: tunnel, transport

o set mode {tunnel |transport}

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

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

Test 2: If transport mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in transport mode and also configures a VPN peer to operate in transport mode. The evaluator configures the TOE and the VPN peer to use any of the allowed cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

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

2.3.10.4 NDcPP22e: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 ensures 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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that the TOE (FTD) implements IPsec using the ESP protocol as defined by RFC 4303, using the cryptographic algorithms AES-CBC-128, AES-CBC-256, AES-GCM-128 and AES-GCM-256 (both specified by RFCs 3602 and 4106) along with SHA-based HMAC algorithms. The TOE allows the administrator to define the IPsec proposal for any IPsec connection to use specific encryption methods and authentication methods including ESP Hash methods: sha1, sha256, sha384 or sha512. These algorithms are consistent with the algorithms specified in FCS_IPSEC_EXT.1.4 and FCS_COP.1/KeyedHash.

It goes on to state that FXOS implements IPsec using the ESP protocol as defined by RFC 4303, using the cryptographic algorithms AES-CBC-128, AES-CBC-256, and AES-GCM-128 (both specified by RFCs 3602 and 4106) along with SHA-based HMAC algorithms, and using IKEv2. The list of these HMAC algorithms identified further in the TSS is SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512. This is consistent with the algorithms specified in FCS_IPSEC_EXT.1.4 and FCS_COP.1/KeyedHash.

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

Section “FTD Site-to-site VPN” of the Supplement provides instructions for creating an IKEv2 IPsec Proposal in the “Configure IKEv2 IPsec Proposal Object” sub-section. This includes choosing the ESP Hash method, the hash or integrity algorithm to use in the Proposal for authentication, and the ESP Encryption method, the Encapsulating Security Protocol (ESP) encryption algorithm for this Proposal.

Section “Editing FTD Remote Access VPN Policy” of the Supplement provides instructions for adding a new IKEv2 policy which includes choosing the integrity algorithm (hash) used in the IKEv2 policy and the encryption algorithm used to establish the Phase 1 SA for protecting the Phase 2 negotiations.

Section “Configure IPsec Secure Channel” in the FXOS Admin Guide details how to configure the ciphers in FXOS to use the algorithms specified by the requirement.

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.

For this test, the evaluator alternately configured a test peer to accept each of the algorithms claimed and supported by the TOE. The evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was successful with each of the supported algorithms.

2.3.10.5 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that in the evaluated configuration, only IKEv2 is supported and the IKE SA exchanges use only main mode. The IKEv2 protocols implement Peer Authentication using RSA and ECDSA (FTD only) algorithms with X.509v3 certificates. IKEv2 separates negotiation into two phases: SA and Child SA. IKE SA creates the first tunnel, which protects later IKE negotiation messages. The key negotiated in IKE SA enables IKE peers to communicate securely in IKE Child SA. During Child SA IKE establishes the IPsec SA. IKE maintains a trusted channel, referred to as a Security Association (SA), between IPsec peers that is also used to manage IPsec connections, including:

• The negotiation of mutually acceptable IPsec options between peers (including peer authentication parameters),

• The establishment of additional Security Associations to protect packet flows using Encapsulating Security Payload (ESP), and

• The agreement of secure bulk data encryption AES keys for use with ESP. After the two peers agree upon a policy, the security parameters of the policy are identified by an SA established at each peer, and these IKE SAs apply to all subsequent IKE traffic during the negotiation

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 “Configuring FTD Site-to-site VPN” of the Supplement provides instructions for configuring an IPsec VPN connection including choosing the IKE versions to use during IKE negotiations. The default is IKEv2 and only IKEv2 is allowed in the evaluated configuration. This section further describes how to configure the TOE to perform NAT traversal by enabling NAT settings -> Keepalive Messages Traversal under the Advanced tab.

Section “Editing FTD Remote Access VPN Policy” in the Supplement provides instructions for configuring a remote access VPN which includes enabling IKEv2 settings. This section further describes how to configure the TOE to perform NAT traversal by enabling NAT settings -> Keepalive Messages Traversal.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide states that when CC mode is enabled FXOS supports IKEv2 in main mode. NAT traversal for ESP packets to pass through one or more NAT devices, is enabled by default.

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 should 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 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

Test 1: Not applicable. The TOE does not support IKEv1.

Test 2: The evaluator configured the TOE such that a VPN session from a test server traversed a NAT device. The evaluator initiated an IPsec connection and observed that the TOE correctly negotiated the NAT connection to establish a protected IPsec connection.

2.3.10.6 NDcPP22e: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.

For FTD, Section 6.1 (FCS_IPSEC_EXT.1(2)) in the ST indicates that the following encryption algorithms are supported for the IKEv2 payload: AES-CBC-128, AES-CBC-256, AES-GCM-128 and AES-GCM-256 (FTD only). This is consistent with the algorithms selected in FCS_IPSEC_EXT.1.6.

For FXOS, Section 6.1 (FCS_IPSEC_EXT.1(1)) in the ST indicates that FXOS implements IPsec using the ESP protocol as defined by RFC 4303, using the cryptographic algorithms AES-CBC-128, AES-CBC-192, AES-CBC-256, and AES-GCM-128 (both specified by RFCs 3602 and 4106).

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

Section “FTD Site-to-site VPN” of the Supplement provides instructions for configuring the IKEv2 policies in the “Configure IKEv2 IPsec Proposal Object” sub-section. This includes choosing the Integrity Algorithms portion of the Hash algorithm used in the IKE policy and choosing the Encryption algorithm used to establish the Phase 1 SA for protecting Phase 2 negotiations.

Section “Editing FTD Remote Access VPN Policy” of the Supplement provides instructions for adding a new IKEv2 policy which includes choosing the integrity algorithm (hash) used in the IKEv2 policy and the encryption algorithm used to establish the Phase 1 SA for protecting the Phase 2 negotiations.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide details FXOS cipher support.

· **IKEv2 Ciphers***:

o **Encryption algorithms**: AES-CBC-128, AES-CBC-192, AES-CBC-256, AES-GCM-128

o **Integrity algorithms**: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512. Additionally, FXOS supports HMAC-SHA1-96, a truncated version of HMAC-SHA-1.

o **DH Groups**: 14, 15, 16, 19, 20, and 21

· **ESP Ciphers***:

o **Encryption algorithms:** AES-CBC-128, AES-CBC-192, AES-CBC-256, AES-GCM-128

o **Integrity algorithms:** HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512. Additionally, FXOS supports HMAC-SHA1-96, a truncated version of HMAC-SHA-1.

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 will confirm the algorithm was that used in the negotiation.

The evaluator configured IKEv2 profiles for the supported algorithms on each TOE (FTD & FXOS). The evaluator then configured a VPN peer with the corresponding algorithms. The evaluator confirmed that the TOE could establish a session with each algorithm and that the tunnel successfully established with the selected algorithm.

2.3.10.7 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that in the evaluated configuration, only IKEv2 is supported. The IKE SA exchanges use only main mode and the IKE SA lifetimes are able to be limited to 24 hours for Phase 1 (SAs) and 8 hours for Phase 2 (Child SAs). The IKEv2 SA lifetime can be configured in seconds. A value from 120 to 2,147,483,647 seconds can be configured with the default being 86,400 seconds. For FXOS, this value is also able to be limited to 24 hours for Phase 1 and 8 hours for phase 2. The IKEV2 SA lifetime can be configured in seconds in a range from 30 to 1440 minutes. This is consistent with the requirements which indicate support for IKEv2.

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. (TD0800 applied)

Section “FTD Site-to-site VPN” of the Supplement provides further detail for configuring the lifetime of the SA in seconds in the “Configure IKEv2 Policy Object” sub-section.

Section “Editing FTD Remote Access VPN Policy” in the Supplement provides instructions for configuring a remote access VPN connection including configuration of the IKEv2 SA lifetime in seconds.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide details the FXOS IPSec lifetime configuration.

· **IKE SA Life Time:** Configurable up to 24 hours. Only time is supported.

o set ike-rekey-time *minutes*

· **IKE Child SA Life Time:** Configurable up to 8 hours. Only time is supported.

o set esp-rekey-time *minutes*

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 Phase 1 SA lifetime that exceeds the lifetime Phase 1 SA 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. (TD0800 applied)

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

Test 2: The evaluator configured the TOE to have a 24-hour IKE limit and configured the test peer with a 25-hour limit. The evaluator then connected the IPsec VPN between the test peer and the TOE. The evaluator observed through logs and packet captures that the connection was successful and that the TOE rekeyed well before the configured time limit was reached ensuring that the keys were renegotiated successfully and there was no data loss during the rekey.

2.3.10.8 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that in the evaluated configuration, only IKEv2 is supported. The IKE SA exchanges use only main mode and the IKE SA lifetimes are able to be limited to 24 hours for Phase 1 (SAs) and 8 hours for Phase 2 (Child SAs). The IKEv2 SA lifetime can be configured in seconds. A value from 120 to 2,147,483,647 seconds can be configured with the default being 86,400 seconds. For FXOS, this value is also able to be limited to 24 hours for Phase 1 and 8 hours for phase 2. The IKEV2 SA lifetime can be configured in seconds in a range from 30 to 1440 minutes. This is consistent with the requirements which indicate support for IKEv2.

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. (TD0800 applied)

Section “Configuring FTD Site-to-site VPN” in the Supplement provides instructions for configuring a VPN connection including configuration of the lifetime in seconds and in kilobytes. Subsection “Configure IKEv2 Policy Object” provides further detail for configuring the lifetime of the SA in seconds.

Section “Editing FTD Remote Access VPN Policy” in the Supplement provides instructions for configuring a remote access VPN connection including configuration of the IKEv2 Child SA lifetime in seconds and kilobytes and the IKEv2 SA lifetime in seconds.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide includes commands to specify the IKE-SA lifetime and to set the Chile SA lifetime values.

· **IKE SA Life Time:** Configurable up to 24 hours. Only time is supported.

o set ike-rekey-time *minutes*

· **IKE Child SA Life Time:** Configurable up to 8 hours. Only time is supported.

- set esp-rekey-time *minutes*

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:

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.

Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 8 hours for the Phase 2 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 2 SA lifetime that exceeds the Phase 2 SA lifetime on the TOE. 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. (TD0800 applied)

Test 1: The evaluator followed the AGD to configure a maximum lifetime value by data volume, and established a connection with a VPN peer. The IPsec SA timed out after the configured data size was exceeded and the connection reset. A new Phase 2 SA negotiation was required. This test is applicable to FTD only which supports rekey by data volume. FXOS does not support this feature.

Test 2: The evaluator configured a max lifetime of 8 hours on the TOE and a max lifetime of 9 hours on the test peer. The evaluator then connected the IPsec VPN between the test peer and the TOE. The evaluator observed through logs and packet captures that the connection was successful and that the IPsec SA timed out before the configured time limit was reached and a new SA was negotiated. A new Phase 2 SA negotiation was required.

In both tests, the (non-TOE) IPsec test peer was configured with higher lifetime values than the TOE. The TOE was the initiator of the rekey in both cases.

2.3.10.9 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that the IKEv2 protocols supported by the FTD TOE implement the following DH groups: 14 (2048-bit MODP), 19 (256-bit Random ECP) and 20 (384-bit Random EC). The section for FXOS identifies support for the following DH Groups: 14, 15, 16, 19, 20 and 21.

For both FTD and FXOS, the secret ‘x’ generated is 64 bytes long (or 512 bits), and is generated with the DRBG specified in FCS_RBG_EXT.1. This is almost double the size of the highest comparable strength value which is 384 bits.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.10.10 NDcPP22e: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(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that the IKEv2 protocols supported by the FTD TOE implement the following DH groups: 14 (2048-bit MODP), 19 (256-bit Random ECP) and 20 (384-bit Random EC). The section for FXOS identifies support for the following DH Groups: 14, 15, 16, 19, 20 and 21.

For both FTD and FXOS the secret ‘x’ generated is 64 bytes long (or 512 bits), and is generated with the DRBG specified in FCS_RBG_EXT.1. This is almost double the size of the highest comparable strength value which is 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.

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 where these activities have been addressed.

2.3.10.11 NDcPP22e: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 checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Section 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST indicates that the IKEv2 protocols supported by the FTD TOE implement the following DH groups: 14 (2048-bit MODP), 19 (256-bit Random ECP) and 20 (384-bit Random EC). The section for FXOS identifies support for the following DH Groups: 14, 15, 16, 19, 20 and 21. In this section, the TSS describes how to specify which DH groups the TOE will accept.

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

Section “Deciding Which Diffie-Hellman Modulus Group to Use” in the Supplement outlines the supported DH groups and specifies that only DH groups 14, 19 and 20 are supported when in CC mode.

Section “Configuring FTD Site-to-Site VPN” in the Supplement provides instructions for configuring an IPsec VPN connection including configuring the modulus group. Subsection “Configure IKEv2 Policy Object” provides more detailed instructions for adding a DH group to the IKEv2 policy.

Section “Editing FTD Remote Access VPN Policy” in the Supplement provides instructions for configuring remote access IPsec VPN connection including configuration of a DH group in the IKEv2 policy.

Section *Configuring IPsec Secure Channel* in the FXOS Admin Guide details FXOS IPSec configuration for dh groups.

· **IKEv2 Ciphers***:

o **Encryption algorithms**: AES-CBC-128, AES-CBC-192, AES-CBC-256, AES-GCM-128

o **Integrity algorithms**: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512. Additionally, FXOS supports HMAC-SHA1-96, a truncated version of HMAC-SHA-1.

o **DH Groups**: 14, 15, 16, 19, 20, and 21

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 configured the TOE to use the claimed DH groups, and made a successful IPsec connection to an IPsec peer using each of the claimed DH groups. The evaluator was able to capture each DH group using a packet capture to ensure the correct DH group was used.

The TOE supports only IKEv2.

2.3.10.12 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that the FTD TOE has a configuration option to deny tunnel if the phase 2 SA is weaker than the phase 1. The crypto strength check is enabled via the

Enable Security Association (SA) Strength Enforcement checkbox. The FXOS TOE has a similar configuration option identified by the TSS that performs the same configuration step.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator simply follows the guidance to configure the TOE to perform the following tests.

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

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 - The evaluator established the IKE SA with AES-CBC-128 and attempted to establish the ESP SA with AESGCM-256. The evaluator observed that the connection failed due to the ESP key size being larger than IKE.

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 in conjunction with Test 3 above.

2.3.10.13 NDcPP22e:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1(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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that the IKEv2 protocols implement Peer Authentication using the RSA (FTD and FXOS), ECDSA (FTD Only) algorithm with X.509v3 certificates. This is consistent with the algorithms specified in FCS_COP.1/SigGen.

The TOE can be configured to authenticate IPsec connections using RSA (FTD, FXOS) and ECDSA (FTD Only) signatures. When using RSA and ECDSA signatures for authentication, the TOE and its peer must be configured to obtain certificates from the same certification authority (CA).

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

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 will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA and ensure a valid certificate for that CA is loaded into the TOE and marked 'trusted'.

Section “Deciding Which Authentication Method to Use” in the **Supplement** states that only digital certificates are the methods of authentication available for VPNs. Digital certificates use RSA or ECDSA key pairs to sign and encrypt IKE key management messages.

Section “Digital Certificates” in the **Supplement** specifies the ability to use a certificate authority within the TOE, and the “Certificate Authority Certificates” section outlines what a CA is and what the requirements are for a certificate to be a CA.

Section “Configuring FTD Site-to-site VPN” in the **Supplement** provides instructions for configuring the IKEv2 policy which includes selecting the Authentication type to ‘Certificate’. In the ‘Certificate’ field, a selected PKI enrollment object is used to generate a trustpoint with the same name on the managed device. The trustpoint is created when the PKI enrollment object is associated with that device.

Section "FTD Certificate Based Authentication" in **the Supplement** provides instructions for installing a certificate manually or importing a PKCS12 file. The FMC installs a CA certificate (provided in the enrollment object) on the managed device, authenticates the CA server and creates a trustpoint on the managed device.

Section "Editing FTD Remote Access VPN Policy" in the **Supplement** provides instructions for creating a connection profile and configuring a certificate by selecting Interface Identity Certificate from a drop-down list. It also describes configuring the Certificate Maps which are certificate to connection profile maps used for certificate authentication.

Pre-shared keys not used in the evaluated configuration, so the generation and establishment of them is not applicable.

Sections *Configuring IPsec Secure Channel*, *Creating A Trust Point*, and *Importing a Certificate into A Key Ring* in the FXOS Admin Guide details FXOS IPSec configuration for certificates.

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.

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.

2.3.10.14 NDcPP22e: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 6.1 (FCS_IPSEC_EXT.1(1)/FCS_IPSEC_EXT.1(2)[VPN]) in the ST states that the TOE's IKEv2 protocols implement Peer Authentication using the RSA (FTD and FXOS) and ECDSA (FTD only) algorithms with X.509v3 certificates. The TOE can be configured to authenticate IPsec connections using RSA (FTD and FXOS) and ECDSA (FTD only) signatures. When using RSA and ECDSA signatures for authentication, the TOE and its peer must be configured to obtain certificates from the same certification authority (CA). For FTD, The administrator defines rules for matching the DN or FQDN of the IPsec peer certificate by creating a certificate map and adding a rule to the certificate map to match the "Alternative Subject" field of the certificate to a value (FQDN/DN). FXOS supports DN, and the TSS indicates that rules need to be defined for matching the one presented in the certificate.

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 "FTD Certificate Based Authentication" in the Supplement describes configuring the FTD Certificate Map Object which is used to provide an association between a received certificate and a Remote Access VPN connection profile. This includes setting the field for the matching rule to the Subject Alternative Name (SAN) and entering the FQDN of the VPN peer.

Sections *Configuring IPsec Secure Channel, Creating A Trust Point,* and *Importing a Certificate into a Key Ring* in the FXOS Admin Guide details FXOS IPSec configuration for certificates, reference identifiers and trustpoints.

The second part of the Assurance Activity is not applicable.

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:

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.

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.

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

a) 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.

b) 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.

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

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

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

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.

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

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

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

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

Test 2: These results were iterated for IPsec RSA and IPsec ECDSA. For part 1 of this test, the evaluator alternately configured a test peer to use an authentication certificate with the correct SAN: IP address, DNS address (FQDN), and user FQDN (depending on fields the TOE supports). The evaluator then attempted to connect the IPsec VPN between the test peer and the TOE 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. This test is not applicable to FXOS which does not claim SAN, only DN.

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

Test 4: These results are iterated for IPsec RSA and IPsec ECDSA. (Part 1) For this test, the evaluator alternately configured the TOE to look for each of the supported SAN reference identifiers. 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 then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was rejected by the TOE. This test is not applicable to FXOS which does not claim SAN, only DN.

Test 5: These results are iterated for IPsec RSA (FTD and FXOS) and IPsec ECDSA (FTD Only). The evaluator configured a test peer to send an authentication certificate with an authorized DN and confirmed that the IPsec connection succeeded.

Test 6: These results are iterated for IPsec RSA (FTD and FXOS) and IPsec ECDSA (FTD Only). (Part A) For this test, the evaluator configured a test peer to first send an authentication certificate with an authorized DN, and then a nearly identical certificate but with a DN containing a duplicate CN. In each case, the evaluator attempted to establish an IPsec connection, and confirmed that the TOE rejected the certificate containing a duplicate CN. (Part B) For this test, the evaluator configured a test peer to first send an authentication certificate with an authorized DN, and then a nearly identical certificate in which the Organization field of the DN has a trailing null character (84) appended. In each case, the evaluator attempted to establish an IPsec connection, and confirmed that the TOE rejected the certificate containing a DN Organization containing an appended null character.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.11 IPsec Protocol - per TD0824 (VPNGW13:FCS_IPSEC_EXT.1)

2.3.11.1 VPNGW13:FCS_IPSEC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.2 VPNGW13:FCS_IPSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.3 VPNGW13:FCS_IPSEC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.4 VPNGW13:FCS_IPSEC_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.5 VPNGW13:FCS_IPSEC_EXT.1.5

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.6 VPNGW13:FCS_IPSEC_EXT.1.6

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.7 VPNGW13:FCS_IPSEC_EXT.1.7

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.8 VPNGW13:FCS_IPSEC_EXT.1.8

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.9 VPNGW13:FCS_IPSEC_EXT.1.9

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.10 VPNGW13:FCS_IPSEC_EXT.1.10

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.11 VPNGW13:FCS_IPSEC_EXT.1.11

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.12 VPNGW13:FCS_IPSEC_EXT.1.12

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.13 VPNGW13:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.14 VPNGW13:FCS_IPSEC_EXT.1.14

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: All existing activities regarding 'Pre-shared keys' apply to all selections including pre-shared keys. If any selection with 'Pre-shared keys' is included, the evaluator shall check to ensure that the TSS describes how the selection works in conjunction with the authentication of IPsec connections.

Not applicable. No selection with 'Pre-shared keys' is included.

Component Guidance Assurance Activities: If any selection with 'Pre-shared Keys' is selected, the evaluator shall check that the operational guidance describes any configuration necessary to enable any selected authentication mechanisms.

'Pre-shared Keys' is not selected in the ST for IPsec.

Component Testing Assurance Activities: None Defined

2.3.12 NTP Protocol (NDcPP22e:FCS_NTP_EXT.1)

2.3.12.1 NDcPP22e:FCS_NTP_EXT.1.1

TSS Assurance Activities: The evaluator shall examine the TSS to ensure it identifies the version of NTP supported, how it is implemented and what approach the TOE uses to ensure the timestamp it receives from an NTP timeserver (or NTP peer) is from an authenticated source and the integrity of the time has been maintained. The TOE must support at least one of the methods or may use multiple methods, as specified in the SFR element 1.2. The evaluator shall ensure that each method selected in the ST is described in the TSS, including the version of NTP supported in element 1.1, the message digest algorithms used to verify the authenticity of the timestamp and/or the protocols used to ensure integrity of the timestamp.

Section 6.1, (FCS_NTP_EXT.1(1)/FCS_NTP_EXT.1(2)) states that Administrators can update the TOE's clock manually via FXOS or FMC and can also configure the TOE (FXOS and FMC) to use NTP to synchronize the TOE's clock with an external time source. The FTD automatically synchronizes its clock with the FXOS clock. NTPv4 is supported by FMC.

NTPv3 is supported by the FXOS and the NTP timestamp is not updated from broadcast or multicast addresses. IPsec is used to secure the connection between the FXOS and the NTP time source.

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

Section "Configure NTP (FMC/FMCv)" states that NTPv4 only is supported. The section further contains instructions on how to configure an NTP server to use NTP authentication with SHA-1 which is consistent with the ST. The section contains instructions for configuring multiple NTP servers.

The Section *Setting the Date and Time Using NTP* in the FXOS Admin Guide describes the commands, windows and actions necessary to configure FXOS to use NTP to set its date and time. These steps allow the administrator to specify the NTP server that will provide updates and indicate that the administrator may specify up to four (4) NTP servers.

The sections *Set the Date and Time Manually via CLI* and *Set the Date and Time Manually via GUI* in the FXOS Admin Guide describes the commands to manually specify the date and time on the FXOS.

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

The TOE claims support for NTPv4. The evaluator demonstrated a successful connection and time synchronization to an external NTP server using NTPv4 in FCS_NTP_EXT.1.2.

FXOS claims support for NTPv3. The evaluator demonstrated a successful connection and time synchronization to an external NTP server using NTPv3 in FCS_NTP_EXT.1.4.

2.3.12.2 NDcPP22e:FCS_NTP_EXT.1.2

TSS Assurance Activities: None Defined

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

Assurance Activity Note:

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

See FCS_NTP_EXT.1.1 where this activity is addressed.

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

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

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

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

The TOE claims to support SHA-1 in element 1.2 for its NTP message digest algorithm, which is selected in FCS_COP.1 SFRs. The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

The evaluator first configured the TOE to synch its time with an external NTP server. The evaluator also configured the NTP Server for NTP authentication using a SHA-1 key, then followed the AGD to configure the corresponding value on the TOE, ensuring it was correct. The evaluator then set the time of the NTP server to be ahead of the TOE. The evaluator observed that the TOE successfully synchronized its time with the NTP server when configured to use the SHA-1 NTP key. The evaluator used a packet sniffer to capture the traffic between the TOE and the NTP server during this connection and observed that NTPv4 was used by both peers, as well as the SHA-1 key. The evaluator also confirmed that the time was synchronized by confirming that the time reported by the CLI of the TOE matched the time reported by the NTP server.

The evaluator repeated the same test, however this time maintaining the configuration of the TOE, but changing the NTP digest algorithm of the NTP server such that it no longer matched with the TOE. The key remained unchanged, so only the digest algorithm was changed. The evaluator found that despite receiving the NTP traffic and waiting over 10 minutes, that the TOE would no longer synchronize its time with the NTP server as a result of the mismatched configuration.

FXOS does not claim support for NTP message digest for authentication but instead uses IPsec to protect NTP communications. IPsec was tested throughout FCS_IPSEC_EXT.1 SFRs.

2.3.12.3 NDcPP22e:FCS_NTP_EXT.1.3

TSS Assurance Activities: None Defined

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

Section "Configure NTP (FMC/FMCv)" in the FTD Admin Guide states that no configuration of the TOE is necessary to meet this requirement. Rather, it is the default behavior of the TSF.

Section “Setting the Date and Time Using NTP” in the FXOS Admin Guide states that by default FXOS will not accept NTP broadcast or multicast packets, so no additional configuration is necessary.

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

No configuration of the TOE is necessary to prevent NTP updates from broadcast and/or multicast NTP packets.

The evaluator configured an NTP server to send broadcast and multicast NTP traffic that reached the TOE. The evaluator found that despite using the correct NTPv4 (for FMC/FMCv) or NTPv3 (for FXOS), and waiting over 10 minutes, the TOE did not synchronize to the NTP broadcast and/or multicast traffic as expected.

2.3.12.4 NDcPP22e:FCS_NTP_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

(TD0528 applied)

The evaluator found that the TOE was able to be configured with a minimum of 3 NTP time sources. Throughout NTP testing, the evaluator found the behavior to be consistent with RFC5905.

Test 1: The evaluator configured three different, valid NTP servers on the TOE. The evaluator found that after a few minutes, the TOE had chosen one of the three valid NTP servers and successfully synchronized its time with it using the correct NTP version.

Test 2: The evaluator utilized a second NTP server to send valid (but unsolicited) NTP updates to the TOE with a timestamp 20 minutes ahead of the TOE. This second NTP server was not configured on the TOE. After waiting more than 10 minutes, the evaluator found that the TOE did not synchronize to this NTP server that was not configured.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.13 Random Bit Generation (NDcPP22e:FCS_RBG_EXT.1)

2.3.13.1 NDcPP22e:FCS_RBG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.13.2 NDcPP22e: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 6.1 (FCS_RBG_EXT.1) in the ST states that the TOE uses a platform-based random bit generator that complies with ISO/IEC 18031:2011 using HMAC_DRBG w/SHA-256 Deterministic Random Bit Generation (DRBG) operating in FIPS mode. In addition, the DRBG is seeded by an entropy source that is at least 256-bit value derived from various highly sensitive and proprietary noise sources described in the proprietary Entropy Design document. Section 7.4, Table 32 in the ST identifies the DRBG type as HMAC_DRBG which is consistent with the requirement.

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) Cisco proprietary document that has been delivered to NIAP for approval. Note that the entropy analysis has been accepted by NIAP/NSA.

Sections “CC Mode and FIPS Mode” and “Common Criteria (CC) Mode” in the FTD Admin Guide outline what must be done in order to enable CC mode which includes enabling FIPS mode which configures the switch to use the proper DRBG methods.

Section *Enable FIPS and CC Mode* in the FXOS Admin Guide provide instructions for enabling FIPS and CC mode for FXOS, and details that RNG functionality is configured automatically when these modes are enabled.

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 verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. 'generate one block of random bits' means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

If the RNG does not have prediction resistance, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) reseed, (4) generate a second block of random bits (5) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The fifth value is additional input to the first call to generate. The sixth and seventh are additional input and entropy input to the call to reseed. The final value is additional input to the second generate call.

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 CAVP certificates identified in Section 1.1.2.

2.3.14 SSH Server Protocol - per TD0631 (NDcPP22e:FCS_SSHS_EXT.1)

2.3.14.1 NDcPP22e:FCS_SSHS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.14.2 NDcPP22e:FCS_SSHS_EXT.1.2

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

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

If password-based authentication method has been selected in the FCS_SSHS_EXT.1.2, then the evaluator shall confirm its role in the authentication process is described in the TSS. (TD0631 applied)

FMC and FTD

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST indicates that the TOE supports SSH host key authentication using rsa-sha2-256, rsa-sha2-512, and ecdsa-sha2-nistp384 algorithms and SSH user public key-based authentication using ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and ecdsa-sha2-nistp521.

FXOS

Section 6.1 (FCS_SSHS_EXT.1(1)) states that the TOE supports Public key algorithms rsa-sha2-256, rsa-sha2-512 and ecdsa-sha2-nistp384 for signing and verification as part of the SSH authentication (i.e., for Host Key authentication). It further states that the TOE supports SSH user public key-based authentication using ssh-rsa, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384 and ecdsa-sha2-nistp521.

In both cases, the TOE ensures and verifies that the SSH client's presented public key matches one that is stored within the TOE's SSH server's authorized keys file. The TOE also supports password-based user authentication for SSH. Additionally, in both cases, the algorithms available for host key authentication and public key authentication are consistent with the selections made in FCS_COP.1.1/SigGen.

Guidance Assurance Activities: None Defined

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

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

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

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

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

(TD0631 applied)

Test 1: The evaluator generated a public/private keypair corresponding to each SSH public key algorithm supported by the TOE. These are outlined in the TSS. The evaluator followed the AGD to configure this keypair on the TOE, then was able to perform a successful login using each public key authentication algorithm supported.

Test 2: The evaluator attempted an SSH connection, with the SSH client configured to present a public key for authentication that is not configured on the TOE SSH server. The TOE rejected this connection attempt as expected.

Test 3: The evaluator attempted to connect to the TOE using a SSH client alternately using the correct and incorrect password. The evaluator found that only the correct password would yield a successful SSH session.

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

2.3.14.3 NDcPP22e:FCS_SSHS_EXT.1.3

TSS Assurance Activities: The evaluator shall check that the TSS describes how 'large packets' in terms of RFC 4253 are detected and handled.

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST states that for FTD, SSH connections will be dropped if the TOE receives a packet larger than 262126 bytes.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST states that for FXOS, SSH connections will be dropped if the TOE receives a packet larger than 262,126 bytes.

Guidance Assurance Activities: None Defined

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

The evaluator created and sent a packet from an SSH client to the TOE SSH server that was larger than the maximum packet size. After this packet was received by the TOE, the TOE rejected the packet and the connection was closed.

2.3.14.4 NDcPP22e:FCS_SSHS_EXT.1.4

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

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST states that the FTD/FMC/FMCv TOE supports SSHv2 with the following encryption algorithms: aes128-cbc, aes256-cbc, AEAD_AES_128_GCM and AEAD_AES_256_GCM. These encryption algorithms are consistent with those specified in the requirement.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST states that the FXOS TOE supports Encryption algorithms, AES-CBC-128, AES-CBC-256 and AEAD-AES-256-GCM to ensure confidentiality of the session. These encryption algorithms are consistent with those specified in the requirement.

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

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST. Section "Configure SSH Host Key Algorithms" also defines configuration for ensuring the correct SSH host key algorithms are being used.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section *Configure SSH Access* in the FXOS Admin Guide describes the configuration and implementation of SSH, including cipher selection, host key generation and deletion, and public key configuration for FXOS.

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

The evaluator attempted to connect to the TOE using a SSH client alternately using each of the ciphers that can be claimed to verify they are supported with successful connections. In each case the evaluator viewed the Server: Key Exchange Init packet and saw that no additional ciphers beyond those that were claimed were seen as supported.

2.3.14.5 NDcPP22e:FCS_SSHS_EXT.1.5

TSS Assurance Activities: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the SSH server's host public key algorithms supported are specified and that they are identical to those listed for this component. (TD0631 applied)

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST indicates that the FTD/FMC/FMCv TOE supports SSH host key authentication using rsa-sha2-256, rsa-sha2-512, and ecdsa-sha2-nistp384 algorithms.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST indicates that the FXOS TOE supports Public key algorithms rsa-sha2-256, rsa-sha2-512 and ecdsa-sha2-nistp384 for signing and verification as part of the SSH authentication (i.e., Host Key authentication).

These algorithms listed in the TSS are consistent with the ones specified in the requirement. This is consistent with the algorithms selected in FCS_COP.1/SigGen. The TOE ensures and verifies that the SSH client’s presented public key matches one that is stored within the TOE’s SSH server’s authorized keys file. The TOE also supports password-based user authentication for SSH.

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

Section "Configure SSH HostKey Algorithms" defines the necessary configuration to ensure that only the selected SSH Host Key algorithms are being used on the FMC and FTD.

Section *Configure SSH Access* in the FXOS Admin Guide describes the configuration and implementation of SSH, including cipher selection, host key generation and deletion, and public key configuration for FXOS.

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

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

Has effectively been moved to FCS_SSHS_EXT.1.2.

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

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

(TD0631 applied)

Test 1: The evaluator established an SSH connection with the TOE using each claimed host key algorithm. These connections were successful.

Test 2: The evaluator attempted to connect to the TOE using a host public key algorithm that is not included in the ST selection and observed that the connection failed.

2.3.14.6 NDcPP22e:FCS_SSHS_EXT.1.6

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

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST states that the FTD/FMC/FMCv TOE supports SSHv2 with HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-512, AEAD_AES_128_GCM and AEAD_AES_256_GCM for integrity and authenticity.

Section 6.1 (FCS_SSHS_EXT.1(1) in the ST states that the FXOS TOE supports Hashing algorithm hmac-sha1 to ensure the integrity of the session for FXOS. FXOS additionally supports hmac-sha2-256, hmac-sha2-512 and AEAD-AES-256-GCM.

These integrity algorithms are consistent with those specified in the requirement.

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

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section *Configure SSH Access* in the FXOS Admin Guide describes the configuration of SSH including how to specify only the approved algorithms.

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

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

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

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

Test 1: The evaluator established an SSH connection with the TOE using each of the claimed integrity algorithms. The evaluator observed a successful connection using each claimed integrity algorithm.

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

These connections were performed using an algorithm other than one of the aes*-gcm@openssh.com encryption algorithms as required.

2.3.14.7 NDcPP22e:FCS_SSHS_EXT.1.7

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

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST states that the FTD/FMC/FMCv TOE supports SSHv2 with diffie-hellman-group14-sha1 (supported only by FTD), ecdh-sha2-nistp256 (supported only by FTD), ecdh-sha2-nistp384, and ecdh-sha2-nistp521 (supported only by FTD) for the key exchange methods. In other words, this means that the FMC only supports ecdh-sha2-nistp384. The FTD supports all of the listed key exchange algorithms.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST states that the FXOS TOE supports SSHv2 requiring use of DH group 14 and ecdsa-sha2-nistp384 as key exchange methods.

These are consistent with the lists in the components.

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

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section *Configure SSH Access* in the FXOS Admin Guide describes the configuration of SSH including how to specify only the approved dh group: Diffie-hellman-group14-sha1.

Firepower-chassis /system/services # **set ssh-server kex-algorithm diffie-hellman-group14-sha1**

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

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

Test 1: The evaluator attempted to establish an SSH connection with the TOE using diffiehellman-group1-sha1 key exchange. The connection attempt failed.

Test 2: The evaluator attempted to establish an SSH connections with the TOE using an SSH client configured for only one of each allowed key exchange method at a time, iterating through the TOE's claimed key exchange algorithms. These connections succeeded.

2.3.14.8 NDcPP22e:FCS_SSHS_EXT.1.8

TSS Assurance Activities: The evaluator shall check that the TSS specifies the following:

a) Both thresholds are checked by the TOE.

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

Section 6.1 (FCS_SSHS_EXT.1(2)) in the ST indicates that the FTD/FMC/FMCv TOE manages a tracking mechanism for each SSH session so that it can initiate a new key exchange when either approximately 1 hour of time or 1GB of data is reached.

Section 6.1 (FCS_SSHS_EXT.1(1)) in the ST indicates that for the FXOS TOE, an SSH connection will be rekeyed after a configurable time or data limit, whichever threshold is met first.

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

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section “Configure SSH ReKey Configuration (Optional)” in the FTD Admin Guide states that when CC mode is enabled, the SSH rekeying will occur approximately at 1 hour of time or after 1 GB of data has been transmitted, whichever occurs first. This section further provides instructions for lowering these thresholds if desired.

Section *Configure SSH Access* in the FXOS Admin Guide describes the configuration of SSH including specification of the rekey limits based on volume and time.

Firepower /system/services # **set ssh-server rekey-limit volume [KB] time [Minutes]**

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

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

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

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

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

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

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

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

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

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

The SSH Data and time rekey thresholds are not configurable so the default of 1GB and 1 hour was tested for all devices.

The evaluator attempted to connect to the TOE using a SSH client generating 1GB of data and verified that a rekey happened when the threshold was reached.

The evaluator attempted to connect to the TOE using a SSH client waiting an hour and verified that a rekey happened right before the 1-hour threshold.

Furthermore, the evaluator observed that the TOE was the initiator of the rekey in both cases, as the test SSH client was observed receiving the rekey initiation request from the TOE. The evaluator also verified the amount of traffic that was sent in the traffic-based threshold (data) case by checking the total amount of data sent over the SSH session before the rekey, and observing that it was less than 1GB.

There were no hardware limitations of the TOE that would have prevented the threshold from being met.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.15 TLS Client Protocol Without Mutual Authentication - per TD0670 & TD0790 (NDcPP22e:FCS_TLSC_EXT.1)

2.3.15.1 NDcPP22e:FCS_TLSC_EXT.1.1

TSS Assurance Activities: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that the ciphersuites supported are specified. The evaluator shall check the TSS to ensure that the ciphersuites specified include those listed for this component.

Section 6.1 (FCS_TLSC_EXT.1) in the ST states that when CC mode is enabled, the TOE is restricted to only support TLSv1.2 for HTTPS sessions and client/server communications between TOE components and for syslog communications with AES 128- or 256-bit symmetric ciphers in CBC and GCM modes, in conjunction with SHA, RSA, and ECDSA. The TOE implements TLS clients to support secure syslog connections, and TLS server and clients to support FPT_ITT.1. This section specifies the ciphersuites for each TOE component and TLS communication channel by reference to the requirements.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Section "CC Mode and FIPS Mode" in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. There are additional features such as enabling the power-up integrity HMAC-SHA-512 self-test, enabling FIPS mode, and other TLS required checks such as the ones specified in section 6 of RFC 6125. CC Mode must be enabled in the evaluated configuration.

Section "Common Criteria (CC) Mode" in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section "Restrict Access and Enable CC Mode" in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section "Configure Syslog over TLS for FMC and FTD" in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server. This includes generating a certificate request and importing a client certificate. This section indicates that if the system is configured to use CRLs, it will use the same CRL to validate both audit client certificates and HTTPS certificates to secure the HTTPS connection between the system and a web browser.

Section "Device Registration" in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

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

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

Test 3: The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send an ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite). The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.

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

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

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

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

Test 5: The evaluator shall perform the following modifications to the traffic:

a) Change the TLS version selected by the server in the Server Hello to a non-supported TLS version and verify that the client rejects the connection.

b) [conditional]: If using DHE or ECDH, modify the signature block in the Server's Key Exchange handshake message, and verify that the handshake does not finished successfully, and no application data flows. This test does not apply to cipher suites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

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

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

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

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

These tests, and the following TLSC_EXT.1.* tests were iterated on all TOE components that act as a TLS Client. These include

1. Syslog over TLS from FMC/FMCv

2. Syslog over TLS from FTD OS TLS Client & 3. Syslog over TLS from FTD TLS Client

4. The FPT_ITT.1 channel (for the FTD only, as it acts as the TLS Client on this channel)

Test 1: The evaluator established a TLS session from the TOE to a test server with the test server configured to accept connections with only one of the claimed cipher suites. The evaluator used a network sniffer to capture the TLS session negotiation. The evaluator examined each traffic capture and observed that the expected TLS cipher was negotiated. The evaluator repeated this test for each claimed ciphersuite. The evaluator observed that the claimed ciphersuites were able to yield a successful connection.

Test 2: As part of FIA_X509_EXT.1, test 1, the evaluator configured the test server to send a certificate with the Server Authentication purpose in the extendedKeyUsage field and observed that the TLS session was accepted by the TOE. Next the TLS session was attempted again with the test server using a cert that is missing the Server Authentication purpose in the extendedKeyUsage field. The evaluator captured the TLS session negotiation and observed that the TLS session was rejected by the TOE.

Test 3: The evaluator established a TLS session from the TOE. A modified test server negotiates a ciphersuite, but returns a certificate using a different algorithm. Using a network sniffer to capture the TLS session negotiation, the evaluator observed that the TLS session was rejected.

Test 4: The evaluator configured a test server to accept only the TLS_NULL_WITH_NUL_NULL ciphersuite. The evaluator then attempted to establish a TLS session from the TOE to that test server. Using a network sniffer, the evaluator captured the TLS session negotiation and observed that the TLS session is rejected by the TOE. The evaluator then configured the test server to send a client hello with a ciphersuite that is not supported by the TOE. The TLS session is rejected by the TOE. As the TOE supports ciphersuites with ECDH key exchange, the evaluator set up the test server to attempt a TLS session negotiation using an unsupported curve size (P-192). The TOE successfully rejected the connection attempt.

Test 5: The evaluator configured a test server to send an invalid TLS version (1.4). The evaluator then attempted to establish a TLS session from the TOE to that test server. The packet capture indicated that the TOE rejected the invalid TLS version and disconnected the session. As the TOE supports DHE ciphers, the evaluator configured a test server to modify the signature block in the Server's Key Exchange handshake message. The evaluator then attempted to establish a TLS session from the TOE to that test server. The packet capture indicated that the TOE rejected the invalid handshake and disconnected the session.

Test 6: The evaluator connected the TOE to a TLS server which modified traffic according to the scenarios identified above for this test case (a, b, c). In each case, the TOE successfully detected the modifications and rejected the connection to the TOE.

2.3.15.2 NDcPP22e:FCS_TLSC_EXT.1.2

TSS Assurance Activities: The evaluator shall ensure that the TSS describes the client's method of establishing all reference identifiers from the administrator/application-configured reference identifier, including which types of reference identifiers are supported (e.g. application-specific Subject Alternative Names) and whether IP addresses and wildcards are supported.

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

If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC5952 for IPv6, RFC 3986 for IPv4) is enforced.

Section 6.1 (FCS_TLSC_EXT.1) in the ST states that when in CC mode and the TOE acts as a TLS client (e.g., connection to the syslog server), the TOE will verify the server Subject Alternative Name (SAN) against the reference identity (wildcard is supported as required in section 6 of RFC 6125 and per RFC 5280 Appendix A). RFC 5280 is supported for the TLS connection between the distributed TOE components (FMC and FTD) and the attribute type “id-at-title” is used by the TOE client to match the presented identifier with the configured identifier. If verification fails, the TLS connection will not be established. Mutual authentication must be configured with the client-side X.509v3 certificate with RSA 2048-bits (or higher) and SHA-256 (or higher). The key agreement parameters of the server key exchange message are specified in the RFC 5246 (section 7.4.3) for TLSv1.2 and the TOE conforms to this RFC.

IP addresses are not supported in the CN as reference identifiers.

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 includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

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

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server. Subsection “Specify the external audit server” provides instructions for specifying the destination host for the audit information by using the FQDN of the syslog server in the “Host” field. This will be used as its reference identifier. If the certificate Subject Alternative Name (SAN) does not match the expected hostname (i.e., reference identifier), the audit log connection will fail.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. During the registration process, the FMC and FTD confirm they have a matching registration key, and use their initial self-signed TLS certificates to uniquely identify themselves to each other (each device certificate signed by FMC, including its own, contains a unique identifier stored as an ‘id-at-title’ attribute, which FMC and FTD each use as the unique reference identifier for each other).

Testing Assurance Activities: Note that the following tests are marked conditional and are applicable under the following conditions:

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

or

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

or

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

Note that for some tests additional conditions apply.

IP addresses are binary values that must be converted to a textual representation when presented in the CN of a certificate. When testing IP addresses in the CN, the evaluator shall follow the following formatting rules:

- IPv4: The CN contains a single address that is represented a 32-bit numeric address (IPv4) is written in decimal as four numbers that range from 0-255 separated by periods as specified in RFC 3986.

- IPv6: The CN contains a single IPv6 address that is represented as eight colon separated groups of four lowercase hexadecimal digits, each group representing 16 bits as specified in RFC 4291. Note: Shortened addresses, suppressed zeros, and embedded IPv4 addresses are not tested.

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

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

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

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

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

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

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

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

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

f) Objective: The objective of this test is to ensure the TOE is able to differentiate between IP address identifiers that are not allowed to contain wildcards and other types of identifiers that may contain wildcards.

Test 6: [conditional]If IP address identifiers are supported in the SAN or CN, the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) (e.g. CN=*.168.0.1 when connecting to 192.168.1.20, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

This negative test corresponds to the following section of the Application Note 64/105: 'The exception being, the use of wildcards is not supported when using IP address as the reference identifier.'

Remark: Some systems might require the presence of the SAN extension. In this case the connection would still fail but for the reason of the missing SAN extension instead of the mismatch of CN and reference identifier. Both reasons are acceptable to pass Test 6.

(TD0790 applied, supersedes TD0670)

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

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

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

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

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

For all TOE TLSC components, the evaluator performed the following tests (as applicable):

Test 1: The evaluator attempted a TLS session from the TOE targeting a server using a valid certificate with a CN matching the domain name used by the client and observed that the connection was successful. The evaluator then attempted a TLS session from the TOE targeting a server using a server certificate that contains an identifier that does not match the Common Name (CN) and omits the Subject Alternative Name (SAN) and observed that connection was rejected.

Test 2: The evaluator attempted a TLS session from the TOE targeting a server using a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The evaluator observed that the connection was rejected.

Test 3: The evaluator attempted a TLS session from the TOE targeting a server using a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The evaluator observed that the connection was successful.

Test 4: The evaluator attempted a TLS session from the TOE targeting a server using a server certificate that contains a CN that does not match the reference identifier but does contain an identifier in the SAN that matches. The evaluator observed that the connection was successful.

Test 5: The evaluator tested hostname wildcards by configuring an expected DNS on the TOE with the test server's certificates configured with wildcard DNS names. Each of the scenarios outlined by the Assurance Activity were tested. The following table shows the observed results:

Certificate Contents | Host ID | Expected Result

Part 1: Control Test

Part 2: CN=bar..example.com | bar.foo.example.com | No Connection*

Part 3: SAN=bar..example.com | bar.foo.example.com | No Connection*

Part 4: CN=.example.com | foo.example.com | Successful Connection*

Part 5: SAN=.example.com | foo.example.com | Successful Connection*

Part 6: CN=.com | example.com | No Connection*

Part 7: SAN=.com | example.com | No Connection*

Part 8: CN=.example.com | bar.foo.example.com | No Connection*

Part 9: SAN=.example.com | bar.foo.example.com | No Connection*

The connection was only successful where the leftmost label in the server certificate was a wildcard, and the TOE was configured with a corresponding reference identifier containing a single leftmost label. In all other scenarios, the connection was rejected by the TOE as expected.

Test 6: The TOE does not support IP addresses as the reference identifier.

Test 7: For ITT (Distributed TOE) communications, the ST claims to support matching the id-at-title according to RFC 5280 Appendix A to the presented reference identifier. As such, the evaluator configured the TOE to connect with the distributed TOE peer using TLS with the peer alternately configured with a certificate identifier as listed below. The evaluator verified that the TOE only connected when the identifier fulfilled the required rules:

- Incorrect UUID: the TOE rejected the certificate and the connection failed as expected.

- Correct UUID, wrong attribute: the TOE rejected the certificate and the connection failed as expected.

- Correct UUID, no SAN: the TOE accepts the certificate and proceeds with successful session establishment.

- Wildcard in left side of UUID: the TOE detects a mismatch and rejects the connection.

- Wildcard in right side of UUID: the TOE detects a mismatch and rejects the connection.

2.3.15.3 NDcPP22e:FCS_TLSC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

Test 1: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds and a trusted channel can be established.

Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 3 : The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g.

inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

Test 1: This test was performed as part of NDcPP22e:FCS_TLSC_EXT.1.1-t1 where the various cipher suites were tested. Each successful TLS connection was using a valid cert chain.

Test 2: This test has been performed in several other test activities. Specifically, this test repeats the assurance activities as described here.

match the reference identifier -- Corresponds to FCS_TLSC_EXT.1.2 Tests 1 through 7.

validate certificate path -- Corresponds to FIA_X509_EXT.1/Rev.1 Test 1

validate expiration date -- Corresponds to FIA_X509_EXT.1/Rev.1 Test 2

determine the revocation status -- Corresponds to FIA_X509_EXT.2 Test 1.

Test 3: Not applicable. The TOE does not support any override mechanisms.

2.3.15.4 NDcPP22e:FCS_TLSC_EXT.1.4

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

Section 6.1 (FCS_TLSC_EXT.1) in the ST states that the following NIST curves are presented with the Client Hello by default - secp256r1, secp384r1, and secp521r1.

Guidance Assurance Activities: If the TSS indicates that the Supported Elliptic Curves/Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Elliptic Curves/Supported Groups Extension.

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

The FTD component’s Syslog TLS Client must be configured to meet the requirement. Section “Configure Logging” contains instructions to be performed along with initial setup of logging that restrict the Supported Groups to meet the requirement.

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

The evaluator configured the test server to support only one ECDHE group at a time. The evaluator then attempted a connection from the TOE to the test server. The TOE successfully allowed the connection attempts for all of the supported curve sizes.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.16 TLS Client Support for Mutual Authentication - per TD0670 (NDcPP22e:FCS_TLSC_EXT.2)

2.3.16.1 NDcPP22e:FCS_TLSC_EXT.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 TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Section 6.1 (FCS_TLSC_EXT.2) in the ST states that mutual authentication must be configured with the client-side X.509v3 certificate with RSA 2048-bits (or higher) and SHA-256 (or higher). Mutual authentication is supported by the FMC/FMCv TLS client for syslog over TLS, as well as the FTD TLS client for the ITT channel.

Component Guidance Assurance Activities: If the TSS indicates that mutual authentication using X.509v3 certificates is used, the evaluator shall verify that the AGD guidance includes instructions for configuring the client-side certificates for TLS mutual authentication.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide indicates that for a successful secure TLS connection between the FMC and an external syslog server, a signed audit client certificate must be imported. The administrator can generate a certificate request, send it to a CA, and then import the signed certificate received from the CA. This section also provides the instructions for generating a CSR and importing the resulting certificate for both the FMC or the FTD. Sub-Section “Enable Syslog over TLS and Mutual Authentication” in the FTD Admin Guide provides instructions for enabling mutual authentication when configuring syslog over TLS for the FMC. This section provides a warning that if mutual authentication is enabled without importing a valid audit client certificate, the connection will fail.

Component Testing Assurance Activities: For all tests in this chapter the TLS server used for testing of the TOE shall be configured to require mutual authentication.

Test 1: The evaluator shall establish a connection to a peer server that is configured for mutual authentication (i.e. sends a server Certificate Request (type 13) message). The evaluator observes that the TOE TLS client sends both client Certificate (type 11) and client Certificate Verify (type 15) messages during its negotiation of a TLS channel and that Application Data is sent.

In addition, all other testing in FCS_TLSC_EXT.1 and FIA_X509_EXT.* must be performed as per the requirements.

Testing was executed using mutual authentication for the FMC/FMCv component that requires mutual authentication, as well as the FTD ITT channel.

For the FMC/FMCv syslog TLSC component, the evaluator configured a test server to support TLS with mutual authentication. The evaluator caused the TOE to attempt a connection to the test server. The evaluator observed that the TOE sent both a certificate and certificate verify message after receiving the certificate request. The TOE and test peer finished the handshake, allowing application data to flow.

For the FTD ITT component, the evaluator used a packet sniffer to capture a standard TLS handshake between the FMC and FTD device on the ITT channel. The evaluator found that the FMC sent a certificate request, and that the

FTD correctly responded with the client certificate and certificate verify messages, and continued to finish the handshake, allowing application data to flow.

2.3.17 TLS Server Protocol Without Mutual Authentication - per TD0635 (NDcPP22e:FCS_TLSS_EXT.1)

2.3.17.1 NDcPP22e:FCS_TLSS_EXT.1.1

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

Section 6.1 (FCS_TLSS_EXT.1) in the ST states that the TOE implements HTTP over TLS (or HTTPS) to support remote administration on FMC and FXOS, TLS clients to support secure syslog connections, and TLS server and clients to support FPT_ITT.1. A remote administrator can connect over HTTPS to the TOE with their web browser. This section further identifies the ciphersuites that are supported by the TOE TLSS which are identical to the ones selected in the requirement.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE may have to be restricted to meet the requirements).

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

The section "*Configuring HTTPS*" in the FXOS Admin Guide describes that when CC mode is enabled, FXOS will restrict the TLS version to 1.2 and ciphersuites to only the set of algorithms selected in the Security Target. The evaluator further verified that this list matched the ones selected in the requirement.

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

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

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

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

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

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

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

These results, as well as the other FCS_TLSS_EXT.1.* tests were iterated for 2 variations: HTTPS and ITT (Distributed TOE communications). HTTPS tests were performed on the FMC and FXOS components. ITT tests were similarly performed on the FMC components, as the FMC/FMCv acts as the TLS Server in the ITT channel (FXOS is excluded from ITT testing).

Test 1: The evaluator attempted to connect to the TOE using each of the claimed ciphersuites. A packet capture was obtained for each connection attempt. The evaluator confirmed that for each of the claimed ciphersuites, the connection was successful.

Test 2: The evaluator attempted to connect to the TOE using the TLS_NULL_WITH_NULL_NULL cipher suite and observed that the connection was rejected. The evaluator then attempted to connect to the TOE using all ciphers except those defined in the PP and observed that the TOE rejected the connection attempt.

Test 3: (a,b): The evaluator made connection attempts from a client to the TOE. The client implementation of the TLS protocol was modified as stated in parts a and b of the assurance activity. In Scenario 3 b), the evaluator observed the packet capture and ensured that the first byte of the encrypted Finished message does not equal 0x14.

2.3.17.2 NDcPP22e:FCS_TLSS_EXT.1.2

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

Section 6.1 (FCS_TLSS_EXT.1) in the ST states that when CC mode is enabled, the TOE is restricted to only support TLSv1.2 for HTTPS sessions and client/server communications between TOE components. If the TLS client does not support TLSv1.2, the TLS connection will fail.

Guidance Assurance Activities: The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

Section *Configuring HTTPS* in the FXOS Admin Guide describes configuration of TLS for FXOS interfaces.

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

The evaluator attempted to establish a TLS session on the TOE with each of the unsupported versions of SSL and TLS. The evaluator used a network sniffer to capture the session negotiation and observed that the expected protocol and version were offered and rejected during negotiation.

2.3.17.3 NDcPP22e:FCS_TLSS_EXT.1.3

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

Section 6.1 (FCS_TLSS_EXT.1) in the ST states that the key agreement parameters for each of the TLS connections in the TOE are as follows:

1. FMC/FMCv (HTTPS/TLS)- 2048-bit RSA, ECDHE secp256r1, secp384r1 and secp521r1
2. FMC/FMCv and FTD (ITT) – 2048-bit RSA, ECDHE secp256r1, secp384r1 and secp521r1
3. FXOS (HTTPS/TLS) - 2048-bit RSA, ECDHE secp256r1, secp384r1 and secp521r1

Guidance Assurance Activities: The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

Section “CC Mode and FIPS Mode” in the FTD Admin Guide provides instructions for enabling CC mode on the FMC which should be done prior to enabling CC mode on a managed device (i.e. FTD). This section states that enabling CC mode will restrict the SSH algorithms, SSH rekey, TLS versions and TLS cipher suites (including elliptical curves) to the approved ones claimed in the Security Target. CC Mode must be enabled in the evaluated configuration.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC mode and FIPS mode on the FTD. Enabling CC mode will limit the TLS and SSH algorithms to the ones specified in the ST.

Section “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server.

Section *Configuring HTTPS* in the FXOS Admin Guide describes how to configure the FXOS TLS Server to support evaluated ciphersuites. This determines the supported key establishment methods allowed by FXOS interfaces. FXOS performs key establishment using ECDSA with secp256r1, secp384rl or secp521r1 NIST curves. There is no further configuration needed.

Testing Assurance Activities: Test 1: [conditional] If ECDHE ciphersuites are supported:

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

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

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

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

Test 1: The evaluator attempted to establish a TLS session with the TOE and configure the remote peer (i.e., the client) to offer each supported ECDH key exchange. Using a network sniffer to capture the TLS session negotiation and observed that the TLS session is accepted by the TOE when using supported ECDH key exchange groups [Part a]. The evaluator then configured the remote peer (i.e., the client) to offer up an unsupported ECDHE curve size (P-192) and verified that the TOE rejected the connection [Part b].

Test 2: Not applicable. The TOE does not support TLSS DHE ciphersuites.

Test 3: The evaluator attempted to establish a TLS session with the TOE and configure the remote peer (i.e., the client) to offer each RSA key size supported by the TOE. Using a network sniffer to capture the TLS session negotiation and observed that the TLS session is accepted by the TOE when using supported RSA key sizes.

2.3.17.4 NDcPP22e:FCS_TLSS_EXT.1.4

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

If session tickets are supported, the evaluator shall verify that the TSS describes that the session tickets are encrypted using symmetric algorithms consistent with FCS_COP.1/DataEncryption. The evaluator shall verify that the TSS identifies the key lengths and algorithms used to protect session tickets.

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

Section 6.1, FCS_TLSS_EXT.1 of the TSS states that TLS session resumption is supported for the following TLS connections of the TOE – the WebUI of the FMC/FMCv and the WebUI of FXOS. The session tickets used for TLS session resumption are encrypted using symmetric algorithms consistent with FCS_COP.1/DataEncryption claims in this ST – AES used in CBC and GCM modes and key sizes of 128 and 256 bits. The session tickets adhere to the structural format provided in section 4 of RFC 5077.

Guidance Assurance Activities: None Defined

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

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

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps: Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

d) The client completes the TLS handshake and captures the SessionID from the ServerHello.

e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).

f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

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

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).

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

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

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with a ServerHello with an empty SessionTicket extension, NewSessionTicket, ChangeCipherSpec and Finished messages (as seen in figure 2 of RFC 5077).

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

Test 1: Not applicable. The TOE does support session resumption using session tickets.

Test 2: Not applicable. The TOE does not support session resumption based on session ID.

Test 3: The evaluator first attempted to resume a session using a valid session ticket. The TOE correctly reuses the client's proposed session ticket and the session is successfully resumed. In this case, the sessionID in the second ServerHello packet matches the client hello's proposed sessionID. The evaluator then attempted a second connection in which the session ticket is modified to be different from the server provided session ticket. The TOE correctly rejects the proposed session ticket as an invalid ticket and sends a new NewSessionTicket packet and performs a full handshake.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.18 TLS Server Support for Mutual Authentication (NDcPP22e:FCS_TLSS_EXT.2)

2.3.18.1 NDcPP22e:FCS_TLSS_EXT.2.1

TSS Assurance Activities: The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

The evaluator shall verify the TSS describes how the TSF uses certificates to authenticate the TLS client. The evaluator shall verify the TSS describes if the TSF supports any fallback authentication functions (e.g. username/password, challenge response) the TSF uses to authenticate TLS clients that do not present a certificate. If fallback authentication functions are supported, the evaluator shall verify the TSS describes whether the fallback authentication functions can be disabled.

Section 6.1 of the ST, FCS_TLSS_EXT.2 states that Mutual authentication must be configured with the client-side X.509v3 certificate with RSA 2048-bits (or higher) and SHA-256 (or higher). The FMC and FTD must successfully complete a registration process to communicate, which requires administrative actions on the FMC and corresponding administrative actions on the FTD. The administrative actions on FMC and FTD require the administrator to input a “registration key” that the two devices will use to authenticate their initial TLS communications. During the registration process, the FMC and FTD confirm they have a matching registration key and use their initial self-signed TLS certificates to uniquely identify themselves to each other (each device certificate signed by FMC, including its own, contains a unique identifier stored as an ‘id-at-title’ attribute, which FMC and FTD each as the unique reference identifier for each other). If the authentication succeeds, the local CA within the FMC will sign and issue a new TLS certificate for the FTD and send (over the existing TLS session) the FTD’s new identity certificate and associated keys, and the FMC’s root CA cert, and the FMC’s root CA certificate and the device certificates which it signed will be used to authenticate all subsequent TLS sessions between the two devices. If device registration fails due to mismatched registration keys, or incorrect IP address or hostname, the information on the FMC and/or FTD needs to be corrected and the registration from FMC reinitiated. In this ITT exchange, the FMC will act as the TLS Server, and the FTD as the TLS Client.

It further states that for mutual authentication, no fallback authentication for certificates is supported. The TOE will reject the connection when the certificate is deemed invalid.

Guidance Assurance Activities: If the TSS indicates that mutual authentication using X.509v3 certificates is used, the evaluator shall verify that the AGD guidance includes instructions for configuring the client-side certificates for TLS mutual authentication.

The evaluator shall verify the guidance describes how to configure the TLS client certificate authentication function. If the TSF supports fallback authentication functions, the evaluator shall verify the guidance provides instructions for configuring the fallback authentication functions. If fallback authentication functions can be disabled, the evaluator shall verify the guidance provides instructions for disabling the fallback authentication functions.

Section "Additional Configuration for FMC/FTD Communication (ITT)" states that on the ITT communications channel between the FMC and FTD, the FMC will act as the TLS Server and the FTD will act as the TLS Client. Both channels support mutual authentication, however, no configuration is necessary for these channels to utilize mutual authentication. Similarly, no manual configuration of the certificates is required on this channel, other than registering the FTD to the FMC as specified in section “Device Registration on FMC”. The TOE does not support fallback authentication methods, and no corresponding configuration is necessary.

Testing Assurance Activities: Test 1a [conditional]: If the TOE requires or can be configured to require a client certificate, the evaluator shall configure the TOE to require a client certificate and send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 1b [conditional]: If the TOE supports fallback authentication functions and these functions cannot be disabled. The evaluator shall configure the fallback authentication functions on the TOE and configure the TOE to send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify the TOE authenticates the connection using the fallback authentication functions as described in the TSS.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

Test 2 [conditional]: If TLS 1.2 is claimed for the TOE, the evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the connection is denied.

Test 3: The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.

Test 4: The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.

Test 5: The evaluator shall perform the following modifications to the traffic:

a) Configure the server to require mutual authentication and then connect to the server with a client configured to send a client certificate that is signed by a Certificate Authority trusted by the TOE. The evaluator shall verify that the server accepts the connection.

b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message (see RFC5246 Sec 7.4.8). The evaluator shall verify that the server rejects the connection.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

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

Test 6: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds, and a trusted channel can be established.

Test 7: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 8 [conditional]: The purpose of this test is to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

Test 1: The evaluator configured a test TLS client to connect to the TOE TLS server.

[Part a] The evaluator first attempted a control test to verify correct connectivity to the TLS server in a typical case. The evaluator established a second connection, but this time the TLS client did not provide a certificate as requested by the TOE. The client was unable to connect when the server required a certificate and none was provided.

[Part b] This conditional test is not applicable, as the TOE does not support fallback authentication.

Test 2: The TOE does support TLS 1.2. The evaluator configured a TLS client to attempt to connect to the TOE TLS server where the client was modified to send a certificate request using an unsupported signature algorithm. The evaluator found that the client was unable to connect due to the TOE rejecting the connection because of the mismatched signature algorithm.

Test 3: Test 1: The evaluator configured a test TLS client to attempt to connect with a certificate containing an issuer field that identifies a CA recognized by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not in fact correspond to the CA certificate trusted by the TOE (effectively an imposter CA). The TOE TLS server rejected this certificate from the test TLS client and a connection was not established because of the invalid certificate.

Test 4: The evaluator configured a test TLS client to connect to the TOE TLS server for two connections. The first connection attempt was made with a valid certificate including the client authentication purpose in the extendedKeyUsage field. During the second connection attempt the test TLS client provides a certificate without client authentication purpose. The evaluator found that the first connection was successful, and that the second attempt was rejected by the TOE TLS server as a result of the missing client authentication extendedKeyUsage value.

Test 5: The evaluator configured a test TLS client to connect to the TOE TLS server for two connections.

[Part a] The first connection used a client configured to send a certificate that chains to a CA trusted by the TOE. This connection attempt was successful.

[Part b] No configuration was necessary to enable mutual authentication. The evaluator then modified a bye in the TLS client's certificate verify message during a second connection attempt. The evaluator observed that the TOE TLS server rejected the connection attempt.

Test 6: The evaluator attempted to connect to the TOE TLS Server using a valid client certificate that chains to a CA trusted by the TOE. The connection attempt to the TOE TLS server was successful in this case.

Test 7: As the FMC/FMCv ITT channel supports mutual authentication, it was also tested in FIA_X509_EXT.1 SFRs. Throughout testing of these SFRs, the evaluator observed that proceeding from a control test using a valid certificate (yielding a successful connection) to a negative case utilizing a certificate that is invalid for the failures defined in each respective SFR, that the certificates were rejected as expected and that they were not 'automatically accepted'. Test case FCS_TLSS_EXT.2.3 covers the case of a failed reference identifier match. Testing throughout FIA_X509_EXT.1 SFRs cover failed validation of the certificate path, and failed validation of the expiration date. Revocation checking is not supported on this channel, so the failed determination of revocation status test is not applicable. An override mechanism is not defined.

Test 8: Not applicable, as there is no override mechanism available for certificate validation on the TOE TLS server.

2.3.18.2 NDcPP22e:FCS_TLSS_EXT.2.2

TSS Assurance Activities: The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

The evaluator shall verify the TSS describes how the TSF uses certificates to authenticate the TLS client. The evaluator shall verify the TSS describes if the TSF supports any fallback authentication functions (e.g. username/password, challenge response) the TSF uses to authenticate TLS clients that do not present a certificate. If fallback authentication functions are supported, the evaluator shall verify the TSS describes whether the fallback authentication functions can be disabled.

See FCS_TLSS_EXT.2.1 where this activity is addressed.

Guidance Assurance Activities: If the TSS indicates that mutual authentication using X.509v3 certificates is used, the evaluator shall verify that the AGD guidance includes instructions for configuring the client-side certificates for TLS mutual authentication.

The evaluator shall verify the guidance describes how to configure the TLS client certificate authentication function. If the TSF supports fallback authentication functions, the evaluator shall verify the guidance provides instructions for configuring the fallback authentication functions. If fallback authentication functions can be disabled, the evaluator shall verify the guidance provides instructions for disabling the fallback authentication functions.

See FCS_TLSS_EXT.2.1 where this activity is addressed.

Testing Assurance Activities: Test 1a [conditional]: If the TOE requires or can be configured to require a client certificate, the evaluator shall configure the TOE to require a client certificate and send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 1b [conditional]: If the TOE supports fallback authentication functions and these functions cannot be disabled. The evaluator shall configure the fallback authentication functions on the TOE and configure the TOE to send a

Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify the TOE authenticates the connection using the fallback authentication functions as described in the TSS.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

Test 2 [conditional]: If TLS 1.2 is claimed for the TOE, the evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the connection is denied.

Test 3: The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.

Test 4: The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.

Test 5: The evaluator shall perform the following modifications to the traffic:

a) Configure the server to require mutual authentication and then connect to the server with a client configured to send a client certificate that is signed by a Certificate Authority trusted by the TOE. The evaluator shall verify that the server accepts the connection.

b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message (see RFC5246 Sec 7.4.8). The evaluator shall verify that the server rejects the connection.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

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

Test 6: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds, and a trusted channel can be established.

Test 7: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 8 [conditional]: The purpose of this test is to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

See the previous SFR, FCS_TLSS_EXT.2.1 where the assurance activity is duplicated, and where all of the required testing was performed.

2.3.18.3 NDcPP22e:FCS_TLSS_EXT.2.3

TSS Assurance Activities: The evaluator shall verify that the TSS describes which types of identifiers are supported during client authentication (e.g. Fully Qualified Domain Name (FQDN)). If FQDNs are supported, the evaluator shall verify that the TSS describes that corresponding identifiers are matched according to RFC6125. For all other types of identifiers, the evaluator shall verify that the TSS describes how these identifiers are parsed from the certificate, what the expected identifiers are and how the parsed identifiers from the certificate are matched against the expected identifiers.

Section 6.1, FCS_TLSS_EXT.2 of the TSS states that RFC 5280 is supported for the TLS connection between the distributed TOE components (FMC and FTD) and the attribute type “id-at-title” is used by the TOE client to match the presented identifier with the configured identifier). If verification fails, the TLS connection will not be established.

FQDNs are not supported in this channel.

Guidance Assurance Activities: The evaluator shall ensure that the AGD guidance describes the configuration of expected identifier(s) for X.509 certificate-based authentication of TLS clients. The evaluator ensures this description includes all types of identifiers described in the TSS and, if claimed, configuration of the TOE to use a directory server.

Section "Device Registration" states that the FMC and FTD identify themselves using the 'id-at-title' attribute as their reference identifier. In this case, this applies to the FMC acting as a TLSS authenticating an FTD as a TLSC on the ITT channel. No TOE configuration is necessary.

Testing Assurance Activities: The evaluator shall send a client certificate with an identifier that does not match an expected identifier and verify that the server denies the connection.

The evaluator configured a test client to connect to the FMC TOE using TLS.

The evaluator configured the TLSC to present a valid client certificate and a certificate with an identifier not configured on the TOE but is otherwise valid (e.g., chains to the root configured on the TOE) so that the reference identifier is the only mismatched configuration. The TOE rejected the TLS connection attempt as a result of the unexpected identifier, and the connection failed to yield a usable user session.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: For all tests in this chapter the TLS client used for testing of the TOE shall support mutual authentication.

The FMC component's ITT channel is the only TLS Server on the TOE with support for mutual authentication. As such, that channel was tested throughout FCS_TLSS_EXT.2.*.

The evaluator utilized a TLS client for testing throughout the FCS_TLSC_EXT.2.* SFRs that supports mutual authentication, as demonstrated by testing in the following SFR, FCS_TLSS_EXT.2.1.

2.4 User data protection (FDP)

2.4.1 Full Residual Information Protection (STFFW14e:FDP_RIP.2)

2.4.1.1 STFFW14e:FDP_RIP.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 'Resources' in the context of this requirement are network packets being sent through (as opposed to 'to', as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet. The evaluator shall check to ensure that the TSS describes packet processing to the extent that they can determine that no data will be reused when processing network packets. The evaluator shall ensure that this description at a minimum describes how the previous data are zeroized/overwritten, and at what point in the buffer processing this occurs.

Section 6.1 (FDP_RIP.2) in the ST states that for FTD, the TOE ensures that packets transmitted through the TOE do not contain residual information from previous packets. Packets that are not the required length (for the temporary memory storage location, or for the minimum transmission unit size on the egress interface) use zeros for padding so residual data is never transmitted from the TOE. Packet handling within memory buffers ensures new packets cannot contain portions of previous packets by ensuring that when packets are written to memory locations those memory locations are padded with zeros as necessary to fill the allocated memory size, so no residual data exists within that memory range when the packet is read for transmission. This applies to data plane traffic and even administrative session traffic.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5 Firewall (FFW)

2.5.1 Stateful Traffic Filtering (STFFW14e:FFW_RUL_EXT.1)

2.5.1.1 STFFW14e:FFW_RUL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.1.2 STFFW14e:FFW_RUL_EXT.1.2

TSS Assurance Activities: The evaluator shall verify that the TSS describes a stateful packet filtering policy and the following attributes are identified as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

The evaluator shall verify that each rule can identify the following actions: permit or drop with the option to log the operation. The evaluator shall verify that the TSS identifies all interface types subject to the stateful packet filtering policy and explains how rules are associated with distinct network interfaces.

Section 6.1 (FFW_RUL_EXT.1.1/FFW_RUL_EXT.1.2) in the ST states that the TOE (FTD) provides stateful traffic filtering of IPv4 and IPv6 network traffic. Administratively-defined traffic filter rules (access-lists or Objects > Object Management > Access Control Lists > Extended) can be applied to any interface to filter traffic based on IP parameters including source and destination address, transport layer protocol, type and code, TCP and UDP port numbers. The TOE allows establishment of communications between remote endpoints, and tracks the state of each session (e.g. initiating, established, and tear-down), and will clear established sessions after proper tear-down is completed as defined by each protocol, or when session timeouts are reached.

To track the statefulness of sessions to/from and through the firewall, the TOE maintains a table of connections in various connection states and connection flags. The TOE updates the table (adding, and removing connections, and modifying states as appropriate) based on configurable connection timeout limits, and by inspecting fields within the packet headers. Connection states are further explained in Section 7.2 of the ST.

The proper session establishment and termination followed by the TOE is as defined in the following RFCs:

• RFC 792 (ICMPv4)

• RFC 4443 (ICMPv6)

• RFC 791 (IPv4)

• RFC 2460 (IPv6)

• TCP, RFC 793, section 2.7 Connection Establishment and Clearing

• UDP, RFC 768 (not applicable, UDP is a “stateless” protocol)

Section 6.1 (FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4) in the ST states that each traffic flow control rule on the TOE is defined as either a “permit” rule, or a “deny” rule, and any rule can also contain the keyword “log” which will cause a log message to be generated when a new session is established because it matched the rule. The TOE can be configured to generate a log message for the session establishment or attempt at session establishment of any permitted or denied traffic. When a rule is created to explicitly allow a protocol which is implicitly allowed to spawn additional sessions, the establishment of spawned sessions is logged as well.

Access Control Lists (ACLs) are only enforced after they’ve been applied to a network interface. Any network interface can have an ACL applied to it. Interfaces can be referred to by their identifier (e.g. GigabitEthernet 0/1).

The interface types that can be assigned to an interface are:

- Physical interfaces
    - Ethernet
    - GigabitEthernet
    - TenGigabitEthernet
    - Management
- Port-channel interfaces (designated by a port-channel number)
- Subinterface (designated by the subinterface number)

The default state of an interface depends on the type and the context mode:

- For the “system” context in single mode or multiple context mode, interfaces have the following default states:
    - Physical interfaces = Disabled
    - Subinterfaces = Enabled. However, for traffic to pass through the subinterface, the physical interface must also be enabled.
- For any non-system context (in multiple context mode): All allocated interfaces (allocated to the context by the system context) are enabled by default, no matter what the state of the interface is in the system context. However, for traffic to pass through the interface, the interface also has to be enabled in the system context. If you shut down an interface in the system context, then that interface is down in all contexts to which that interface has been allocated.

In interface configuration mode, the administrator can configure hardware settings (for physical interfaces), assign a name, assign a VLAN, assign an IP address, and configure many other settings, depending on the type of interface and the security context mode.

For an enabled interface to pass traffic, the following interface configuration mode commands must be used (in addition to explicitly permitting traffic flow by applying an access-group to the interface): “**nameif”**, and, for routed mode, “**ip address”**. For subinterfaces, also configure the “**vlan”** command.

Guidance Assurance Activities: The evaluators shall verify that the guidance documentation identifies the following attributes as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

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

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

Section "Passive vs Inline Mode and Default Traffic Flows" in the Supplement states that traffic policies are defined in terms of network zones which in turn are associated with FTD interfaces. So, a policy may be defined to allow traffic from "zone0" to "zone1", though those zones may be mapped to the "outside" and "inside" interfaces on one FTD and also be mapped to "int1" and "int2" interfaces of another FTD which enforces the same policy. No FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it's explicitly permitted by at least one policy rule, thus an implicit "deny-all" rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section "Configure Access Control Policy" in the Supplement states that for each rule, the administrator can specify a rule action, to trust, block or inspect matching traffic with an intrusion policy.

Section "Stateful Session Behaviors" of the Supplement contains a table specifying the types of packets that are decoded and can be configured upon which includes IPv4, IPv6, ICMPv4, ICMPv6, TCP and UDP packets.

Section "Intrusion Rules Editor" in the Supplement contains a table that outlines each protocol type, the required header field inspection and the specific keywords that can be used to configure a rule. For IPv4 and IPv6 the required header fields include the source and destination address and transport protocol. For ICMPv4 and ICMPv6 it includes the type and code, and for TCP and UDP it includes the source and destination port.

Testing Assurance Activities: Test 1: The evaluator shall use the instructions in the guidance documentation to test that stateful packet filter firewall rules can be created that permit, drop, and log packets for each of the following attributes:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

Test 2: Repeat the test evaluation activity above to ensure that stateful traffic filtering rules can be defined for each distinct network interface type supported by the TOE.

Note that these test activities should be performed in conjunction with those of FFW_RUL_EXT.1.9 where the effectiveness of the rules is tested. The test activities for FFW_RUL_EXT.1.9 define the protocol/attribute combinations required to be tested. If those combinations are configured manually, that will fulfil the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

Test 1: This test was performed as part of STFFW14e:FFW_RUL_EXT.1.9 where a firewall rule was defined using each of the protocol attributes defined in this test and packets were sent either matching or not matching the rules. To create these rules the evaluator followed the administrative guidance and found all necessary

instructions were provided accurately. In each case the TOE demonstrated the appropriate permit/deny behavior.

Test 2: This test was performed as part of STFFW14e:FFW_RUL_EXT.1.9 where a firewall rule was defined using each of the protocol attributes defined in this test and packets were sent either matching or not matching the rules. To create these rules the evaluator followed the administrative guidance and found all necessary instructions were provided accurately. The evaluator found that these rules can be applied to all types of supported network interface.

2.5.1.3 STFFW14e:FFW_RUL_EXT.1.3

TSS Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Guidance Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Testing Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

2.5.1.4 STFFW14e:FFW_RUL_EXT.1.4

TSS Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Guidance Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Testing Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

2.5.1.5 STFFW14E:FFW_RUL_EXT.1.5

TSS Assurance Activities: The evaluator shall verify that the TSS identifies the protocols that support stateful session handling. The TSS shall identify TCP, UDP, and, if selected by the ST author, also ICMP.

The evaluator shall verify that the TSS describes how stateful sessions are established (including handshake processing) and maintained.

The evaluator shall verify that for TCP, the TSS identifies and describes the use of the following attributes in session determination: source and destination addresses, source and destination ports, sequence number, and individual flags.

The evaluator shall verify that for UDP, the TSS identifies and describes the following attributes in session determination: source and destination addresses, source and destination ports.

The evaluator shall verify that for ICMP (if selected), the TSS identifies and describes the following attributes in session determination: source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5.

The evaluator shall verify that the TSS describes how established stateful sessions are removed. The TSS shall describe how connections are removed for each protocol based on normal completion and/or timeout conditions. The TSS shall also indicate when session removal becomes effective (e.g., before the next packet that might match the session is processed).

Section 6.1 (FFW_RUL_EXT.1.1/FFW_RUL_EXT.1.2) in the ST states that the TOE (FTD) provides stateful traffic filtering of IPv4 and IPv6 network traffic. Administratively-defined traffic filter rules can be applied to any interface to filter traffic based on IP parameters including source and destination address, transport layer protocol, type and code, TCP and UDP port numbers. The TOE allows establishment of communications between remote endpoints, and tracks the state of each session (e.g. initiating, established, and tear-down), and will clear established sessions after proper tear-down is completed as defined by each protocol, or when session timeouts are reached.

To track the statefulness of sessions to/from and through the firewall, the TOE maintains a table of connections in various connection states and connection flags. The TOE updates the table (adding, and removing connections, and modifying states as appropriate) based on configurable connection timeout limits, and by inspecting fields within the packet headers.

Section 7.2 in the ST describes the establishment and maintenance of stateful TCP and UDP connections. As network traffic enters an interface of the TOE, the TOE inspects the packet header information to determine whether the packet is allowed by access control lists, and whether an established connection already exists for that specific traffic flow. The TOE maintains and continuously updates connection state tables to keep track of establishment, teardown, and open sessions. To help determine whether a packet can be part of a new session or an established session, the TOE uses information in the packet header and protocol header fields to determine the session state to which the packet applies as defined by the RFC for each protocol. Administrators can view the connection state using the “show conn” command to display the number of active TCP and UDP connections.

Section 6.1 (FFW_RUL_EXT.1.5) in the ST states that if it is a new connection the TOE has to check the packet against access control lists and perform other tasks to determine if the packet is allowed or denied. To perform this check, the first packet of the session goes through the "session management path," and depending on the type of traffic, it might also pass through the "control plane path."

The TOE creates forward and reverse flows in the fast path for TCP traffic; the TOE also creates connection state information for connectionless protocols like UDP and ICMP (when you enable ICMP inspection), so that they can also use the fast path.

If the connection is already established, the TOE does not need to re-check packets against the ACL; matching packets can go through the "fast" path based on attributes identified in FFW_RUL_EXT.1.5. For TCP, FFW_RUL_EXT.1.5 identifies the attributes: source and destination addresses, source and destination ports, sequence number, and individual flags. For UDP, FFW_RUL_EXT.1.5 identifies the attributes: source and destination addresses and source and destination ports.

Existing traffic flows are removed from the set of established traffic flows when the session inactivity timeout hits or the completion of the expected information flow.

The proper session establishment and termination followed by the TOE is as defined in the following RFCs:

• RFC 792 (ICMPv4)

• RFC 4443 (ICMPv6)

• RFC 791 (IPv4)

• RFC 8200 (IPv6)

• TCP, RFC 793, section 2.7 Connection Establishment and Clearing

• UDP, RFC 768 (not applicable, UDP is a “stateless” protocol)

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes stateful session behaviours. For example, a TOE might not log packets that are permitted as part of an existing session.

Section "Stateful Session Behaviors" in the Supplement specifically outlines Stateful Session Behaviors and how they can be configured. After the packets are decoded through the first three TCP/IP layers, they are sent to preprocessors which normalize traffic at the application layer and detect protocol anomalies. The following three preprocessors must be enabled and configured in the evaluated configuration (by default, all three preprocessors are enabled):

• TCP Streaming Preprocessor - Administrators can configure the system so that the preprocessor detects any TCP traffic that cannot be identified as part of an established TCP session. Stateful inspection allows administrators to ignore these packets because they are not part of an established TCP session and do not provide meaningful information.

• UDP Streaming Preprocessor - UDP data streams are not typically thought of in terms of sessions. However, the stream preprocessor uses the source and destination IP address fields in the encapsulating IP datagram header and the port fields in the UDP header to determine the direction of flow and identify a session.

• IP Defragmentation Preprocessor - When an IP datagram is broken into two or more smaller IP datagrams because it is larger than the maximum transmission unit (MTU), it is fragmented. A single IP datagram fragment may not contain enough information to identify a hidden attack. Attackers may attempt to evade detection by transmitting attack data in fragmented packets or attempt to crash the system when reassembling the fragmented packets. The IP defragmentation preprocessor reassembles fragmented IP datagrams, and if fragmented datagrams cannot be reassembled, it will be rejected (i.e., dropped) and logged with certain intrusion rules enabled

Testing Assurance Activities: Test 1: The evaluator shall configure the TOE to permit and log TCP traffic. The evaluator shall initiate a TCP session. While the TCP session is being established, the evaluator shall introduce session establishment packets with incorrect flags to determine that the altered traffic is not accepted as part of the session (i.e., a log event is generated to show the ruleset was applied). After a TCP session is successfully established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports, sequence number, flags) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 2: The evaluator shall terminate the TCP session established per Test 1 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 3: The evaluator shall expire (i.e., reach timeout) the TCP session established per Test 1 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 4: The evaluator shall configure the TOE to permit and log UDP traffic. The evaluator shall establish a UDP session. Once a UDP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 5: The evaluator shall expire (i.e., reach timeout) the UDP session established per Test 4 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 6: If ICMP is selected, the evaluator shall configure the TOE to permit and log ICMP traffic. The evaluator shall establish a session for ICMP as defined in the TSS. Once an ICMP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 7: If applicable, the evaluator shall terminate the ICMP session established per Test 6 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 8: The evaluator shall expire (i.e., reach timeout) the ICMP session established per Test 6 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

The evaluation team performed the tests specified in the assurance activity and confirmed that the traffic filter firewall rules operate as expected in each test scenario.

Test 1: The evaluator configured the TOE to allow and log network traffic for a valid TCP session. The evaluator started transmitting packets from a test server to establish a valid TCP session. The packets being sent were constructed such that they would be permitted to pass through the TOE. During the TCP session negotiation, packets with invalid flags are injected at various points in the exchange. The evaluator ensured by examining the TOE logs and viewing captured packets, that the packets with incorrect flags were discarded as not being part of the session. The evaluator also sent packets with incorrect source and destination addresses, incorrect source and destination ports, and incorrect flags and sequence numbers. The evaluator confirmed (through logs and packet captures) that all non-matching packets are not treated as part of the established session.

Test 2: The evaluator configured the TOE to allow and log network traffic for a valid TCP session. The evaluator started transmitting packets from a test server to establish a valid TCP session. The evaluator then terminated the TCP session, and attempted to send additional packets using the same TCP session information. The evaluator ensured by examining the TOE logs and viewing captured packets, that the packet sent after the session termination, was not passed by the TOE as part of the session.

Test 3: Repeated Test 2, expiring the session rather than explicitly terminating the session. The evaluator observed that no packets sent after a session expiration were treated by the TOE as part of the original TCP session.

Test 4: The evaluator configured the TOE to allow and log network traffic for a valid UDP session. The evaluator started transmitting packets from a test server to establish a valid UDP session. The packets being sent were constructed such that they would be permitted to pass through the TOE. The evaluator also sent packets with incorrect source and destination addresses, incorrect source and destination ports, and incorrect flags and sequence numbers. The evaluator confirmed (through logs and packet captures) that all nonmatching packets are not treated as part of the established session.

Test 5: The evaluator performed test 3 (using an expired session) using a UDP session rather than a TCP session. The same passing behavior was observed as in test 3, with the UDP traffic sent after the session expired were not passed by the TOE as part of the session.

Test 6: Not applicable. ICMP was not selected.

Test 7: Not applicable. ICMP was not selected.

Test 8: Not applicable. ICMP was not selected.

2.5.1.6 STFFW14E:FFW_RUL_EXT.1.6

TSS Assurance Activities: The evaluator shall verify that the TSS identifies the following as packets that will be automatically dropped and are counted or logged:

a) Packets which are invalid fragments, including a description of what constitutes an invalid fragment

b) Fragments that cannot be completely re-assembled

c) Packets where the source address is defined as being on a broadcast network

d) Packets where the source address is defined as being on a multicast network

e) Packets where the source address is defined as being a loopback address

f) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address 'reserved for future use' (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4;

g) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as an 'unspecified address' or an address 'reserved for future definition and use' (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;

h) Packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified

i) Other packets defined in FFW_RUL_EXT.1.6 (if any).

Section 6.1 (FFW_RUL_EXT.1.6/FFW_RUL_EXT.1.7) in the ST states that the TOE can be configured to implement default denial of various mal-formed packets/fragments, and other illegitimate network traffic, and can be configured to log that such packets/frames were dropped. This section specifies all the traffic that will be denied and logged consistent with the traffic specified in FFW_RUL_EXT.1.6 and FFW_RUL_EXT.1.7. An exception to this is invalid fragments and fragments which cannot be reassembled completely, which are blocked and counted, rather than logged as specified in the SFR. This section includes specification of packets which are invalid fragments and a description of what constitutes an invalid fragment. The evaluator verified that all traffic listed in this assurance activity is identified.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes packets that are discarded and potentially logged by default. If applicable protocols are identified, their descriptions need to be consistent with the TSS. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

Section “Required Default Settings for the CC Evaluated Configuration” in the Supplement provides instructions for configuring the traffic flow control as required for the CC evaluated configuration.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that no FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it's explicitly permitted by at least one policy rule, thus an implicit "deny-all" rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section "Access Control Rule" in the Supplement provides instructions for creating and editing access control rules including using the logging option.

Section "Managing Intrusion Policies" in the Supplement states that an enabled rule causes the system to generate intrusion events for (and optionally block) traffic matching the rule.

Section "Create Intrusion Policy" in the Supplement outlines the process for the administrator to create a new intrusion policy and includes the settings for specifying if rules should drop packets and log or simply log events that trigger the policy. Section "Intrusion Rule Actions" provides instructions for enabling or disabling a rule within an individual intrusion policy as well as specifying the action the system takes if monitored conditions trigger the rule. The rule's state can be set to Alert (which only generates events) or Block (which blocks the packet and generates an event).

Testing Assurance Activities: Both IPv4 and IPv6 shall be tested for items a), b), c), d), and e) of the SFR element. Both IPv4 and IPv6 shall be tested for item i) unless the rule definition is specific to IPv4 or IPv6. Note: f), g), and h) are specific to IPv4 or IPv6 and shall be tested accordingly.

Test 1: The evaluator shall test each of the conditions for automatic packet rejection in turn. In each case, the TOE should be configured to allow all network traffic and the evaluator shall generate a packet or packet fragment that is to be rejected. The evaluator shall use packet captures to ensure that the unallowable packet or packet fragment is not passed through the TOE.

Test 2: For each of the cases above, the evaluator shall use any applicable guidance to enable dropped packet logging or counting. In each case above, the evaluator shall ensure that the rejected packet or packet fragment was recorded (either logged or an appropriate counter incremented).

The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured firewall rules to allow network traffic and enabled rejected packet logging and counting (specifically for invalid fragments and fragments that cannot be re-assembled completely). The evaluator generated the following types of packets which the TOE rejected and logged. For the fragments, the evaluator generated the packets which the TOE rejected and counted.

Test 1-1, Invalid Fragment IPv4
Test 1-2, Invalid Fragment IPv6
Test 2-1, Incomplete Fragment IPv4
Test 2-2, Incomplete Fragment IPv6
Test 5-1, Invalid Broadcast Source Address IPv4
Test 6-1, Invalid Multicast Source Address IPv4
Test 6-2, Invalid Multicast Source Address IPv6
Test 7-1, Invalid Loopback Source Address IPv4
Test 7-2, Invalid Loopback Source Address IPv6
Test 10-1, Invalid Future Source Address IPv4
Test 10-2, Invalid Future Destination Address IPv4
Test 11-1, Invalid Future Source Address IPv6
Test 11-2, Invalid Future Destination Address IPv6
Test 12-1, Invalid IP Options Loose Source Routing IPv4
Test 12-2, Invalid IP Options Strict Source Routing IPv4
Test 12-3, Invalid IP Options Record Route IPv4

In addition to the PP required default rules, the security target identifies several additional rules which were also tested.

1. The evaluator ensured that the TOE rejected and was capable of logging IPv4 packets with destination IP address equal to 0.0.0.0.

2. The evaluator ensured that the TOE rejected and was capable of logging IPv4 packets with source IP address equal to 0.0.0.0.

3. The evaluator ensured that the TOE rejected and was capable of logging packets with the first octet of the source IP address equal to zero.

4. The evaluator ensured that the TOE rejected and was capable of logging packets with the network part of the source IP address equal to all 0's.

5. The evaluator ensured that the TOE rejected and was capable of logging packets with the network part of the source IP address equal to all 1's.

6. The evaluator ensured that the TOE rejected and was capable of logging packets with the host part of the source IP address equal to all 1's.

7. The evaluator ensured that the TOE rejected and was capable of logging packets with the host part of the source IP address equal to all 0's.

8. The evaluator ensured that the TOE rejected and was capable of logging ICMP error packets when the ICMP error messages are not related to any session already established in the TOE. This was also tested with ICMPv6.

9. The evaluator ensured that the TOE rejected and was capable of logging network packets when the appliance is not able to find any established connection related to the frame embedded in the ICMPv6 error message.

10. The evaluator ensured that the TOE rejected and was capable of logging network packets when an ICMP echo request/reply packet was received with a malformed code (non-zero).

In addition to the PP required default rules, the security target identifies several additional rules, which were also tested. These map to the assignments in part h.) of the FFW_RUL_EXT.1.6 SFR in the ST.

- Reject & Log L2 broadcast packet (MAC address FF:FF:FF:FF:FF:FF)

- Reject & log IPv4 dest address equal to 0.0.0.0

- Reject & log IPv4 source address equal to 0.0.0.0

- Reject & log First octet of the source IP address equal to zero

- Reject & log Network part of the source IP address equal to all zero

- Reject & log Network part of the source IP address equal to all one

- Reject & log Source IP address host part equal to all one

- Reject & log Source IP address host part equal to all zero

- ICMP Error Inspect: error not related to session not allowed.

- ICMPv6 Error Inspect: error not related to session not allowed.

- ICMP Inspect bad icmp code: non-zero code not allowed in ICMP Echo.

- LAND Attack: packets with same source & dest addr/port are not allowed

The evaluator generated traffic based on each condition above and verified that the TOE successfully blocked the traffic.

2.5.1.7 STFFW14E:FFW_RUL_EXT.1.7

TSS Assurance Activities: The evaluator shall verify that the TSS explains how the following traffic can be dropped and counted or logged:

a) Packets where the source address is equal to the address of the network interface where the network packet was received

b) Packets where the source or destination address of the network packet is a link-local address

c) Packets where the source address does not belong to the networks associated with the network interface where the network packet was received, including a description of how the TOE determines whether a source address belongs to a network associated with a given network interface

Section 6.1 (FFW_RUL_EXT.1.6/FFW_RUL_EXT.1.7) in the ST states that the TOE can be configured to implement default denial of various mal-formed packets/fragments, and other illegitimate network traffic, and can be configured to log that such packets/frames were dropped.

This section specifies all the traffic that will be denied and logged consistent with the traffic specified in FFW_RUL_EXT.1.6 and FFW_RUL_EXT.1.7. The traffic is denied/dropped by the default action (deny/drop) of each Access Control Policy, and if logging is enabled for the default action the TOE will generate audit messages when the action occurs. The evaluator verified that all traffic listed in this assurance activity is identified.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes how the TOE can be configured to implement the required rules. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

Section “Required Default Settings for the CC Evaluated Configuration” in the Supplement provides instructions for configuring the traffic flow control as required for the CC evaluated configuration. For the required rules in FFW_RUL_EXT.1.7, the traffic will be dropped by default, and auditing of these events can be enabled by enabling logging on the Default Action of the Access Control Policy.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that no FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it’s explicitly permitted by at least one policy rule, thus an implicit “deny-all” rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section “Access Control Rule” in the Supplement provides instructions for creating and editing access control rules including using the logging option.

Section “Managing Intrusion Policies” in the Supplement states that an enabled rule causes the system to generate intrusion events for (and optionally block) traffic matching the rule.

Section “Create Intrusion Policy” in the Supplement outlines the process for the administrator to create a new intrusion policy and includes the settings for specifying if rules should drop packets and log or simply log events that trigger the policy. Section “Intrusion Rule Actions” provides instructions for enabling or disabling a rule within an individual intrusion policy as well as specifying the action the system takes if monitored conditions trigger the rule. The rule’s state can be set to Alert (which only generates events) or Block (which blocks the packet and generates an event).

Testing Assurance Activities: Test 1: The evaluator shall configure the TOE to drop and log network traffic where the source address of the packet matches that of the TOE network interface upon which the traffic was received. The evaluator shall generate suitable network traffic to match the configured rule and verify that the traffic is dropped and a log message generated.

Test 2: The evaluator shall configure the TOE to drop and log network traffic where the source IP address of the packet fails to match the network reachability information of the interface to which it is targeted, e.g. if the TOE believes that network 192.168.1.0/24 is reachable through interface 2, network traffic with a source address from the 192.168.1.0/24 network should be generated and sent to an interface other than interface 2. The evaluator shall verify that the network traffic is dropped and a log message generated.

These tests were performed for both IPv4 and IPv6 traffic.

Test 1: The evaluator configured the TOE to drop and log network traffic where the source address of the packet matches that of the TOE network interface upon which the traffic was received. The evaluator generated suitable traffic to match the configured rule and confirmed via packet capture and logs that the traffic is dropped and a log message is generated.

Test 2: The evaluator configured the TOE to drop and log network traffic where the source IP address of the packet fails to match the network reachability information of the interface to which it is targeted. The evaluator generated suitable traffic to match the configured rule and confirmed via packet capture and logs that the traffic is dropped and a log message is generated. The evaluator then configured the TOE to drop and log network traffic where the source or destination address of the network packet is a link-local address and confirmed that when traffic matching that rule was generated, the TOE dropped the traffic and generated a log message.

2.5.1.8 STFFW14E:FFW_RUL_EXT.1.8

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

Section 6.1 (FFW_RUL_EXT.1.1/FFW_RUL_EXT.1.2) in the ST states that the TOE (FTD) provides stateful traffic filtering of IPv4 and IPv6 network traffic. Administratively-defined traffic filter rules can be applied to any interface to filter traffic based on IP parameters including source and destination address, transport layer protocol, type and code, TCP and UDP port numbers. To track the statefulness of sessions to/from and through the firewall, the TOE maintains a table of connections in various connection states and connection flags. The TOE updates the table (adding, and removing connections, and modifying states as appropriate) based on configurable connection timeout limits, and by inspecting fields within the packet headers.

Section 6.1 (FFW_RUL_EXT.1.5) in the ST states that all traffic that goes through the TOE is inspected using the Adaptive Security Algorithm and either is allowed through or dropped. If it is a new connection the TOE has to check the packet against access control lists and perform other tasks to determine if the packet is allowed or denied. If the connection is already established, the TOE does not need to re-check packets against the ACL; matching packets can go through the "fast" path based on attributes identified in FFW_RUL_EXT.1.5

Section 6.1 (FFW_RUL_EXT.1.8) in the ST states that the TOE (FTD) administrators have control over the sequencing of access control entries (ACEs) within an access control list (ACL) to be able to set the sequence in which ACEs are applied within any ACL. The entries within an ACL are always applied in a top-down sequence, and the first entry that matches the traffic is the one that's applied, regardless of whether there may be a more precise match for the traffic further down in the ACL. By changing the ordering/numbering of entries within an ACL, the administrator changes the sequence in which the entries are compared to network traffic flows. Thus, while the TOE does not implement a mechanism to ensure that no conflicting rules can be configured, the ACE sequence ensures that the first rule is enforced regardless of the specificity of the rule.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes how the order of stateful traffic filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

Section "Configure Access Control Policy" in the Supplement provides instructions for configuring the access control policy and access control rules. It states that the system matches traffic to access control rules in order; the first matched rule handles the traffic.

Section "Access Control Rule" in the Supplement states that within an access control policy, the system matches traffic to rules in top-down order by rule number. Section "Creating and Editing Access Control Rules' provides instructions for adding new rules or editing existing rules including specifying the rule position.

Testing Assurance Activities: Test1: If the TOE implements a mechanism that ensures that no conflicting rules can be configured, the evaluator shall try to configure two conflicting rules and verify that the TOE rejects the conflicting rule(s). It is important to verify that the mechanism is implemented in the TOE but not in the non-TOE environment. If the TOE does not implement a mechanism that ensures that no conflicting rules can be configured, the evaluator shall devise two equal stateful traffic filtering rules with alternate operations - permit and drop. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation. (TD0545 applied)

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

Test 1: There is no mechanism that forcibly prevents an administrator from configuring conflicting rules. The evaluator first configured a permit rule to permit traffic to a specific destination and a second rule to deny traffic to the same destination. The evaluator confirmed that the traffic was permitted. Subsequently, the evaluator configured a deny rule first with the second rule (permit). The evaluator observed that the traffic was blocked. The original firewall rules were implemented using IPv4. The test was then repeated using IPv6. Both test cases behaved as expected.

Test 2: Continuing test 1, the evaluator repeated the procedure above, except the evaluator changed the rules to make one a subset of the other, and then tested both orders. The evaluator confirmed that the first is enforced regardless of the specificity of the rule. The original firewall rules were implemented using IPv4. The test was then repeated using IPv6. Both test cases behaved as expected.

2.5.1.9 STFFW14e:FFW_RUL_EXT.1.9

TSS Assurance Activities: The evaluator shall verify that the TSS describes the process for applying stateful traffic filtering rules and also that the behavior (either by default, or as configured by the administrator) is to deny packets when there is no rule match unless another required conditions allows the network traffic (i.e., FFW_RUL_EXT.1.5 or FFW_RUL_EXT.2.1).

Section 6.1 (FFW_RUL_EXT.1.9) in the ST states that an implicit “deny-all” rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. “access-list sample-acl deny ip any any log”.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the guidance documentation provides the appropriate instructions to configure the behavior to deny packets with no matching rules.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that no FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it’s explicitly permitted by at least one policy rule, thus an implicit “deny-all” rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section “Access Control Rule” in the Supplement provides instructions for creating and editing access control rules including using the logging option and specifying the rule’s actions which can be allow or block.

Section “Managing Intrusion Policies” in the Supplement states that an enabled rule causes the system to generate intrusion events for (and optionally block) traffic matching the rule. Section “Create Intrusion Policy” outlines the process for the administrator to create a new intrusion policy and includes the settings for specifying if rules should drop packets and log or simply log events that trigger the policy. Section “Intrusion Rule Actions” provides instructions for enabling or disabling a rule within an individual intrusion policy as well as specifying the action the system takes if monitored conditions trigger the rule. The rule’s state can be set to Alert (which only generates events) or Block (which blocks the packet and generates an event).

Testing Assurance Activities: For each attribute in FFW_RUL_EXT.1.2, the evaluator shall construct a test to demonstrate that the TOE can correctly compare the attribute from the packet header to the ruleset, and shall demonstrate both the permit and deny for each case. It shall also be verified that a packet is dropped if no matching rule can be identified for the packet. The evaluator shall check the log in each case to confirm that the relevant rule was applied. The evaluator shall record a packet capture for each test to demonstrate the correct TOE behaviour.

The evaluator defined several tests variations to exercise the attributes and rules from FFW_RUL_EXT.1.2. The evaluator generated traffic to match specific aspects of the configured firewall rule set and confirmed that all attributes demonstrated permit, deny and log for each test case.

2.5.1.10 STFFW14E:FFW_RUL_EXT.1.10

TSS Assurance Activities: The evaluator shall verify that the TSS describes how the TOE tracks and maintains information relating to the number of half-open TCP connections. The TSS should identify how the TOE behaves when the administratively defined limit is reached and should describe under what circumstances stale half-open connections are removed (e.g. after a timer expires).

Section 6.1 (FFW_RUL_EXT.1.10) in the ST states that the TOE (FTD) administrators can configure the maximum number of half-open TCP connections allowed by configuring a Network Analysis Policy to include SYN Attack

Prevention with desired limits. After the configured limit is reached, the TOE will act as a proxy for the server and generates a SYN-ACK response to new client SYN request. When the TOE receives an ACK back from the client, it can then authenticate that the client is real and allow the connection to the server. If an ACK is not received in the configurable time frame, the session is closed, resource is returned to the free pool, and it will be counted. The default idle time until a TCP half-open connection closes is 10 minutes.

Section 6.1 (FFW_RUL_EXT.1.1/FFW_RUL_EXT.1.2) states that to track the statefulness of sessions to/from and through the firewall, the TOE maintains a table of connections in various connection states and connection flags. The TOE updates the table (adding, and removing connections, and modifying states as appropriate) based on configurable connection timeout limits, and by inspecting fields within the packet headers.

It goes on to state that when traffic exceeds the maximum rate the TOE can handle, the TOE drops the excess traffic and ensures that no traffic that wouldn't pass stateful traffic filtering rules will be passed through.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes the behaviour of imposing TCP half-open connection limits and its default state if unconfigured. The evaluator shall verify that the guidance clearly indicates the conditions under which new connections will be dropped e.g. per destination or per-client.

Section "Rate-Based Attack Prevention" in the Supplement describes the SYN attack prevention option which helps protect network hosts against SYN floods which are indicated by any traffic containing excessive incomplete connections to hosts on the network. Instructions are provided for configuring Syn Attack Prevention which prevents half-open 'embryonic' connections. The administrator can specify the number of SYN packets per number of seconds. After a timeout period elapses, if the rate condition has stopped, the event generation and packet dropping stops.

Testing Assurance Activities: Test 1: The evaluator shall define a TCP half-open connection limit on the TOE. The evaluator shall generate TCP SYN requests to pass through the TOE to the target system using a randomised source IP address and common destination IP address. The number of SYN requests should exceed the TCP half-open threshold defined on the TOE. TCP SYN-ACK messages should not be acknowledged. The evaluator shall verify through packet capture that once the defined TCP half-open threshold has been reached, subsequent TCP SYN packets are not transmitted to the target system. The evaluator shall verify that when the configured threshold is reached that, depending upon the selection, either a log entry is generated or a counter is incremented.

The evaluator configured a TCP half-open connection limit on the TOE. The evaluator then generated TCP SYN requests which would pass through the TOE to a target system using a randomized source IP address and common destination IP address. The number of SYN requests sent exceeded the TCP half-open threshold defined on the

TOE. TCP SYN ACK messages were not acknowledged. Using a packet capture, the evaluator verified that once the defined TCP half-open threshold had been reached, subsequent TCP SYN packets are not transmitted to the target system. The evaluator verified that when the configured threshold is reached the TOE behaved as claimed in the Security Target.

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

The evaluator shall verify that the TSS also include a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describe the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets. The description shall also include a description how the TOE behaves in the situation where the traffic exceeds the amount of traffic the TOE can handle and how it is ensured that also in this condition stateful traffic filtering rules are still applied so that traffic does not pass that shouldn't pass according to the specified rules.

Section 6.1 (FFW_RUL_EXT.1.1/FFW_RUL_EXT.1.2) in the ST states that the TOE (FTD) provides stateful traffic filtering of IPv4 and IPv6 network traffic. Administratively-defined traffic filter rules (access-lists or Objects > Object Management > Access Control Lists > Extended) can be applied to any interface to filter traffic based on IP parameters including source and destination address, transport layer protocol, type and code, TCP and UDP port numbers. The TOE allows establishment of communications between remote endpoints, and tracks the state of each session (e.g. initiating, established, and tear-down), and will clear established sessions after proper tear-down is completed as defined by each protocol, or when session timeouts are reached.

To track the statefulness of sessions to/from and through the firewall, the TOE maintains a table of connections in various connection states and connection flags. The TOE updates the table (adding, and removing connections, and modifying states as appropriate) based on configurable connection timeout limits, and by inspecting fields within the packet headers. Connection states are further explained in Section 7.2 of the ST.

The proper session establishment and termination followed by the TOE is as defined in the following RFCs:

• RFC 792 (ICMPv4)

• RFC 4443 (ICMPv6)

• RFC 791 (IPv4)

• RFC 8200 (IPv6)

• TCP, RFC 793, section 2.7 Connection Establishment and Clearing

• UDP, RFC 768 (not applicable, UDP is a “stateless” protocol)

Section 6.1 (FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4) in the ST states that each traffic flow control rule on the TOE is defined as either a “permit” rule, or a “deny” rule, and any rule can also contain the keyword “log” which will cause a log message to be generated when a new session is established because it matched the rule. The TOE can be configured to generate a log message for the session establishment or attempt at session establishment of any permitted or denied traffic. When a rule is created to explicitly allow a protocol which is implicitly allowed to spawn additional sessions, the establishment of spawned sessions is logged as well.

Access Control Lists (ACLs) are only enforced after they’ve been applied to a network interface. Any network interface can have an ACL applied to it. Interfaces can be referred to by their identifier (e.g. GigabitEthernet 0/1).

The interface types that can be assigned to an interface are:

• Physical interfaces

o Ethernet

o GigabitEthernet

o TenGigabitEthernet

o Management

• Port-channel interfaces (designated by a port-channel number)

• Subinterface (designated by the subinterface number)

The default state of an interface depends on the type and the context mode:

• For the “system” context in single mode or multiple context mode, interfaces have the following default states:

o Physical interfaces = Disabled

o Subinterfaces = Enabled. However, for traffic to pass through the subinterface, the physical interface must also be enabled.

• For any non-system context (in multiple context mode): All allocated interfaces (allocated to the context by the system context) are enabled by default, no matter what the state of the interface is in the system context. However, for traffic to pass through the interface, the interface also has to be enabled in the system context. If you shut down an interface in the system context, then that interface is down in all contexts to which that interface has been allocated.

In interface configuration mode, the administrator can configure hardware settings (for physical interfaces), assign a name, assign a VLAN, assign an IP address, and configure many other settings, depending on the type of interface and the security context mode.

For an enabled interface to pass traffic, the following interface configuration mode commands must be used (in addition to explicitly permitting traffic flow by applying an access-group to the interface): "**nameif"**, and, for routed mode, "**ip address"**. For subinterfaces, also configure the "**vlan"** command.

Component Guidance Assurance Activities: The guidance documentation associated with this requirement is assessed in the subsequent test evaluation activities.

See subsequent test evaluation activities.

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

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

Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test evaluation activities.

Test 1: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to block network traffic directed at a specific destination. The evaluator started transmitting packets from a test server. The packets being sent were constructed such that they should be blocked by the configure rule. The evaluator rebooted the TOE and monitored traffic on both sides of the TOE during the reboot process. The evaluator observed that while packets were being delivered to the TOE ingress side, no matching packets were being passed from the egress side.

Test 2: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit network traffic directed at a specific destination. The evaluator started transmitting packets from a test server. The packets being sent were constructed such that they should be accepted by the configured rule. The evaluator rebooted the TOE and monitored traffic on both sides of the TOE during the reboot process. The evaluator observed that there is a gap during which no traffic is passed shortly after the reboot is started until just prior to the login prompt being presented by the TOE. This demonstrates that none of the generated network traffic is permitted through the firewall during initialization and is only permitted once initialization is complete.

2.5.2 Stateful Filtering of Dynamic Protocols (STFFW14e:FFW_RUL_EXT.2)

2.5.2.1 STFFW14e:FFW_RUL_EXT.2.1

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 identifies the protocols that can cause the automatic creation of dynamic packet filtering rules. In some cases rather than creating dynamic rules, the TOE might establish stateful sessions to support some identified protocol behaviors.

The evaluator shall verify that the TSS explains the dynamic nature of session establishment and removal. The TSS also shall explain any logging ramifications.

The evaluator shall verify that for each of the protocols selected, the TSS explains the dynamic nature of session establishment and removal specific to the protocol.

Section 6.1 (FFW_RUL_EXT.2) in the ST states that the TOE (FTD) supports numerous TCP and UDP protocols that require dynamic establishment of secondary network sessions like FTP. The establishment of sessions along with the dynamical definition of the rule are treated as auditable events. The TOE will manage establishment and teardown of the following protocols in accordance with the RFC for each protocol:

• FTP (File Transfer Protocol) is a TCP protocol supported in either active or passive mode:

o In active mode the client initiates the control session, and the server initiates the data session to a client port provided by the client;

o For active FTP to be allowed through the TOE, the firewall rules must explicitly permit the control session from the client to the server, and “inspect ftp” must be enabled. The TOE will then explicitly permit a control session to be initiated from the client to the server, and implicitly permit data sessions to be initiated from the server to the client while the control session is active.

o In passive (PASV) mode, the client initiates the control session, and the client also initiates the data session to a secondary port provided to the client by the server.

For passive FTP to be permitted through the TOE, the firewall rules must explicitly permit the control session from the client to the server, and “inspect ftp” must be enabled with the “match passive-ftp” option enabled. That feature will cause the TOE to look for the PASV or EPSV commands in the FTP control traffic and for the server’s destination port, and dynamically permit the data session.

Component Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes dynamic session establishment capabilities.

The evaluator shall verify that the guidance documentation describes the logging of dynamic sessions consistent with the TSS.

Section “Dynamic Session Establishment” in the FTD Admin Guide describes the logging of dynamic sessions consistent with the TSS.

Section “Access Control Rules” in the FTD Admin Guide provides instructions for adding new rules and modifying existing rules. It states that to support the dynamic session establishment capability for FTP, an access control rule that allows traffic to destination port FTP must be created with logging. This will enable the FTP application detector to allow the FTP data connections without an additional explicit rule.

Component Testing Assurance Activities: Test 1: The evaluator shall define stateful traffic filtering rules to permit and log traffic for each of the supported protocols and drop and log TCP and UDP ports above 1024. Subsequently, the evaluator shall establish a connection for each of the selected protocols in order to ensure that it succeeds. The evaluator shall examine the generated logs to verify they are consistent with the guidance documentation.

Test 2: Continuing from Test 1, the evaluator shall determine (e.g., using a packet sniffer) which port above 1024 opened by the control protocol, terminate the connection session, and then verify that TCP or UDP (depending on the protocol selection) packets cannot be sent through the TOE using the same source and destination addresses and ports.

Test 3: For each additionally supported protocol, the evaluator shall repeat the procedure above for the protocol. In each case the evaluator must use the applicable RFC or standard in order to determine what range of ports to block in order to ensure the dynamic rules are created and effective.

Test 1: The evaluator first configured the TOE and created two firewall rules: the first rule allows FTP traffic (TCP, port 21) for a specific destination, and a second rule (to block TCP traffic for ports above 1024). The evaluator then caused the test server to attempt to establish an FTP connection with an FTP server. The test server successfully established an FTP session and identified the data ports specified by the FTP server. The TOE did not block any of the FTP related connections, even though the TOE was configured with a rule to block TCP packets with a destination port great than 1024 (put another way, the TOE correctly, dynamically allowed establishment of FTP data sessions).

Test 2: The evaluator attempted to reuse the previous information from the terminated session to send FTP data.

The evaluator verified that the attempt failed.

Test 3: The TOE does not support any other protocols

2.6 Identification and authentication (FIA)

2.6.1 Authentication Failure Management (NDcPP22e:FIA_AFL.1)

2.6.1.1 NDcPP22e:FIA_AFL.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.1.2 NDcPP22e: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 6.1 (FIA_AFL.1) in the ST indicates that for the FMC provides the administrator the ability to specify the maximum number (can be set differently per account on FMC) of unsuccessful authentication attempts via SSH or WebUI (default is five attempts, configurable from 1-999) before the offending account is locked. The configured limit is the maximum number of allowed consecutive failures, thus the defined number of unsuccessful consecutive authentication attempts that results in locking of accounts is one more than the maximum number of allowed consecutive failures. Only an authorized administrator (with the 'administrator' role) can unlock a locked account. By default, the predefined 'admin' account is exempt from becoming locked, but that default is overridden when CC mode is enabled. If all admin accounts become locked for any reason, FMC can be accessed locally using password recovery procedures.

The FTD CLI provides the administrator the ability to specify the maximum number of unsuccessful authentication attempts (configurable from 1-9999) before the offending account is locked. Only an authorized administrator (with the 'administrator' role) can unlock a locked account. An account can also be configured to be unlocked after an administrator-defined time period without the intervention of another administrator with 'admin' role. If all admin accounts become locked for any reason, FTD can be accessed locally using password recovery procedures. The FTD CLI also provides the administrator the ability to specify the time duration for which a locked account remains locked (configurable from 1 to 9999 minutes). Once an account is locked, it automatically unlocks after the time-period that the administrator has configured, has elapsed.

FXOS will allow a maximum number (same value applies to all FXOS accounts) of consecutive failed login attempts via SSH or WebUI before the offending account becomes locked ('lock-status' set to 'locked'). When an account is locked, it can be unlocked by another administrator who has the 'admin' role (not just 'read-only'). An account can also be configured to be unlocked after an administrator-defined time period without the intervention of another administrator with 'admin' role. If all admin accounts become locked for any reason, FXOS can be accessed locally using password recovery procedures.

- All types of user accounts (including account type 'admin') are locked out of the system after exceeding the maximum number of login attempts.
- The default maximum number of unsuccessful login attempts is '3' (configurable from 1-10).

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 “Adding New User Accounts” in the FTD Admin Guide provides instructions for configuring the maximum number of failed login attempts during account creation via the FMC. The default setting is 5. This setting can be modified as described in section “Modifying and Deleting User Accounts”. Section “User and Role Management” in the FTD Admin Guide states that on FMC, the WebUI accounts are separate from the CLI accounts. Even though they both have a default account called “admin” those accounts and their credentials are separate. If the ‘admin’ account for WebUI becomes locked (e.g., due to consecutive failed login attempts), the ‘admin’ account for CLI will remain unlocked, and vice versa. To avoid risk of lockout on FMC, restrict the source IP addresses that can initiate SSH, or disable SSH.

Section “Unlocking FMC Accounts” in the FTD Admin Guide provides instructions for unlocking an FMC account that has become locked due to exceeding the configured Maximum Number of Failed Logins.

Section “Configure Authentication” in the FTD Admin Guide provides the command for configuring the maximum number of failed login attempts before a user account will be locked. It also provides the command for unlocking the FTD account. Section “User and Role Management” in the FTD Admin Guide describes how to avoid risk of lockout on FMC, by restricting the source IP addresses that can initiate SSH.

Section “Configuring Unlock Time” in the FTD Admin Guide contains instructions for the administrator to configure the amount of time an account is locked out for. After this time period, an account is automatically unlocked. Alternatively, an administrator can manually unlock the account.

Section “Password Recovery Procedures” in the FTD Admin Guide provides references to online guidance for instructions on resetting the password on the FMC GUI and FMC CLI accounts.

Section *Set the Maximum Number of Login Attempts* in the FXOS Admin Guide details configuration of maximum login attempts and unlock time

set max-login-attempts *max_login*

set user-account-unlock-time *unlock_time*

The section also details the command used to unlock an administrator account manually.

clear lock status

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: The evaluator configured the limit for the maximum number of failed login attempts at 3 for FTD (this value was 5 for FXOS & FMC). The evaluator then logged in to the TOE using incorrect credentials three times (five for FXOS & FMC). The evaluator then attempted to use correct credentials and found that the account had been locked out in each case, with valid credentials no longer working. This test was repeated on both the FTD, FMC, and FXOS.

Since both FTD and FXOS alternately support a time-based account unlock, the evaluator also attempted the same procedure with the time-based implementation. The values for FTD were 2 failed logins and a 2-minute lockout. The values for FXOS were 3 failed logins and a 10-minute lockout. In both cases, once the configured failed login threshold value was reached, the TOE no longer allowed access to the account even when valid credentials were provided, confirming that the account was locked.

Test 2: Continuing Test 1, the evaluator then successfully unlocked the account in each case using the procedures found in the guidance. This test was repeated on the FTD, FMC and FXOS. After the account was manually unlocked by an administrator, the user could then log in again with a valid password.

After the account was locked in the time-based case, the evaluator also tried a valid password just before and after the configured time period had elapsed. Just before the time elapsed, valid credentials still did not work. After the configured time had elapsed, the account lock was cleared and the valid credentials worked again, resulting in successful access.

2.6.2 Password Management - per TD0792 (NDcPP22e:FIA_PMG_EXT.1)

2.6.2.1 NDcPP22e: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 lists the supported special character(s) for the composition of administrator passwords.

The evaluator shall check the TSS to ensure that the minimum_password_length parameter is configurable by a Security Administrator.

The evaluator shall check that the TSS lists the range of values supported for the minimum_password_length parameter. The listed range shall include the value of 15.

(TD0792 applied)

Section 6.1 of the ST, FIA_PMG_EXT.1 states that 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 as listed in the SFR (FXOS passwords do not support "=" or "$" characters). Minimum password length is settable by the Authorized Administrator, and support passwords of 8 to 32 characters (FTD and FMC) and 8 to 127 characters (FXOS) when "enforce-strong-password" option is enabled in security scope. Password composition rules specifying the types and number of required characters that comprise the password are settable by the Authorized Administrator. Passwords can be configured with a maximum lifetime, configurable by the Authorized Administrator. New passwords can be required to contain a minimum of 4-character changes from the previous password.

The value of 15 is within the configurable range of the minimum password length parameter.

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 "Adding New User Accounts" in the FTD Admin Guide provides instructions for configuring the minimum password length and also provides guidance on strong password composition using the identified characters consistent with the ST. Minimum password length can be set to be 8-127 characters with the default being 8 characters. Passwords must be at least 8 alphanumeric characters of mixed case and must include at least one numeric character and one special character. It cannot be a word that appears in a dictionary or include consecutive repeating characters. It is also recommended that the "Check Password Strength" checkbox be selected.

Section "Configure Authentication" in the FTD Admin Guide also provides instructions for configuring the minimum password length and strength check directly via the FTD CLI for FTD accounts.

Sections *Enable Password Strength Check* **and** *Configure the Minimum Password Length* in the FXOS Admin Guide describe the constraints enforced upon passwords used to authenticate through FXOS interfaces. These sections provide guidance and instructions to allow an administrator to set the minimum password and support the guidelines presented for a strong password.

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

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.

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: The evaluator successfully set valid user passwords on each TOE component with compositions including the following:

-Min configurable length

-demonstrating special character set

-demonstrating number set

-demonstrating uppercase set

-demonstrating lowercase set

-Max length

Test 2: The evaluator attempted to set invalid user passwords on each TOE component with compositions including the following. All invalid password attempts were rejected.

-short password

-long password

-no numbers

-no special character

-no uppercase

-no lowercase

The evaluator also found that the minimum and maximum length were accurate with regard to the Security Target claims. The subset chosen for testing covers characters a-z, A-Z, 1-9, and all of the special characters selected in element 1.1. This is sufficient as it covers the composition of passwords that could be used by administrators in a real-world scenario.

2.6.3 Protected Authentication Feedback (NDcPP22e:FIA_UAU.7)

2.6.3.1 NDcPP22e: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. The ST states that when logging in, the TOE will not echo passwords such that passwords are not inadvertently displayed to the user and any other users that might be able to view the login display. The TOE replaces the entered password character with a "*" character or does not show any characters at all.

Section "Logging into the Appliance" in the FTD Admin Guide describes and provides instructions for logging in remotely via Web UI and CLI as well as logging in locally via the serial console. It also provides an example screenshot of the login via the Web UI that shows that the password is not displayed and provides a notice to the user to observe that the password is not displayed.

Section "Logging into the Appliance" in the FXOS Admin Guide similarly describes and provides instructions for logging in remotely via Web UI and CLI, as well as locally via the serial console. This section notes to the reader that it will be observed that the password is not displayed while being 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 FIA_UIA_EXT.1 where the evaluator observed that passwords are obscured on the console logins.

2.6.4 Password-based Authentication Mechanism (NDcPP22e:FIA_UAU_EXT.2)

2.6.4.1 NDcPP22e:FIA_UAU_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.1 (FIA_UAU_EXT.2) in the ST states that the TOE provides local password-based authentication mechanisms to FMC, FTD and FXOS. The process for authentication is the same for administrative access whether administration is occurring via a directly connected console cable or remotely via SSHv2 (password-based or public key-based) or TLS. At initial login in the administrative user is prompted to provide a username. After the user provides the username, the user is prompted to provide the administrative password associated with the user account. The TOE then either grants administrative access (if the combination of username and password is correct) or indicates that the login was unsuccessful. The TOE does not provide indication of whether the username or password was the reason for an authentication failure. FXOS also provides support for RADIUS and TACAC+ authentication mechanisms. For SSH connections, an administrative user can also present an SSH private key along with a username, rather than a username/password combination. The SSH private key must correspond to an SSH public key that is loaded on the TOE by following the instructions found in the AGD. After the user provides the username and SSH private key, the TOE then either grants administrative access (if the private key corresponds to a public key known to the TOE and the username is correct) or will prompt for a password, indicating the SSH public/private keypair login was unsuccessful. Similarly in this case, the TOE does not provide an indication whether the username or SSH private key was the reason for authentication failure.

See also FIA_UIA_EXT.1.

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

See FIA_UIA_EXT.1

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

See FIA_UIA_EXT.1.

2.6.5 User Identification and Authentication (NDcPP22e:FIA_UIA_EXT.1)

2.6.5.1 NDcPP22e:FIA_UIA_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.5.2 NDcPP22e:FIA_UIA_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 describes the logon process for each logon method (local, remote (HTTPS, SSH, etc.)) supported for the product. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a 'successful logon'.

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

For distributed TOEs the evaluator shall examine that the TSS details how Security Administrators are authenticated and identified by all TOE components. If not all TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the TSS shall describe how the overall TOE functionality is split between TOE components including how it is ensured that no unauthorized access to any TOE component can occur.

For distributed TOEs, the evaluator shall examine the TSS to determine that it describes for each TOE component which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration. For each TOE component that does not support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 the TSS shall describe any unauthenticated services/services that are supported by the component.

Section 6.1 (FIA_UIA_EXT.1) in the ST states that the TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed. All the TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2. Administrative access to the TOE is facilitated through the TOE's CLI (SSH (password-based and public key-based) or local console in FTD, FMC and FXOS), or web GUI in FMC and FXOS. The TOE mediates all administrative actions through the CLI and GUI. The TOE presents a warning banner in accordance with FTA_TAB.1 requirement prior to initiating the identification authentication mechanism for those attempting to access the TOE. Once a potential administrative user attempts to access an administrative interface either locally or remotely, the TOE prompts the user for a username and password. For remote SSH connections, an administrative user can alternately use a username and an SSH public/private keypair for authentication rather than a username and password combination. Only after the administrative user presents the correct authentication credentials will access to the TOE administrative functionality be granted. No access is allowed to the administrative functionality of the TOE until an administrator is successfully identified and authenticated.

Section 6.1 (FIA_UAU_EXT.2) in the ST states that the TOE provides local password-based authentication mechanisms to FMC, FTD and FXOS. The process for authentication is the same for administrative access whether administration is occurring via a directly connected console cable or remotely via SSHv2 (password-based or public key-based) or TLS. At initial login in the administrative user is prompted to provide a username. After the user provides the username, the user is prompted to provide the administrative password associated with the user account. The TOE then either grants administrative access (if the combination of username and password is correct) or indicates that the login was unsuccessful. The TOE does not provide indication of whether the username or password was the reason for an authentication failure. FXOS also provides support for RADIUS and TACAC+ authentication mechanisms. For SSH connections, an administrative user can also present an SSH private key along with a username, rather than a username/password combination. The SSH private key must correspond to an SSH public key that is loaded on the TOE by following the instructions found in the AGD. After the user provides the username and SSH private key, the TOE then either grants administrative access (if the private key corresponds to a public key known to the TOE and the username is correct) or will prompt for a password, indicating the SSH public/private keypair login was unsuccessful. Similarly in this case, the TOE does not provide an indication whether the username or SSH private key was the reason for authentication failure.

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 “Restrict Access and Enable CC Mode” in the FTD Admin Guide states that the system by default only supports SSH and HTTPS security protocols for management. The system is required to support only the cipher suites, version, and protocols claimed in the Security Target. HTTPS, TLS, and SSH connection settings are configured automatically when CC mode is enabled.

Section “Logging into the Appliance” in the FTD Admin Guide provides instructions for logging into the appliance via WebUI, Remote CLI connection, and Local CLI connection. The remote CLI connection includes instructions for both SSH password and public-key authentication methods.

Section “Generating an HTTPS Server Certificate Signing Request” in the FTD Admin Guide provides instructions for generating a certificate request through the local configuration HTTPS Certificate page. Section “Importing HTTPS Server Certificate” provides instructions for importing the HTTPS certificate.

Section “Common Criteria (CC) Mode” in the FTD Admin Guide provides instructions for configuring CC and FIPS mode which limits the algorithms used for HTTPS/TLS and SSH on FTD to those claimed in the ST. Section “Configure Authentication” in the FTD Admin Guide provides instructions for logging into the FTD appliance via the CLI.

Section “FTD Remote Access VPN” in the Supplement provides instructions for configuring an IPSec VPN client connection to the FTD for remote administration. It further states that if a Remote Access VPN session is being used to tunnel SSH for remote administrative access to FTD (SSH over IPsec) and the IPsec connection between from the VPN client (AnyConnect) and FTD is unintentionally broken the AnyConnect client will automatically attempt to re-initiate the IPsec connection. If AnyConnect is able to automatically reestablish the IPsec session the tunneled SSH session, if previously established, may have remained active though it may have timed out and thus would need to be reinitiated from the client. In most cases no administrative action is required (on the AnyConnect client nor on FTD) though any connectivity issues will need to be resolved in the networks between AnyConnect and FTD. If IPsec connectivity has been lost for an extended period of time AnyConnect will discontinue automatic attempts to reestablish the tunnel, in which case the local administrative user of AnyConnect must re-initiate the IPsec tunnel, and reinitiate the SSH session through IPsec to FTD.

The section entitled *Configuring HTTPS* in the FXOS Admin Guide describes how to configure the FXOS Web UI.

Sections *Configure SSH Access* and *Generate the SSH Host Key* in the FXOS Admin Guide provides instructions to set up inbound SSH session to the FXOS SSH Server interface.

The *Enable Password Strength Check* section in the FXOS Admin Guide describes the management of password constraints for FXOS.

Sections entitled *Configure RADIUS via CLI* and Configure *TACACS+ via CLI* of the FXOS Admin Guide describe configuration of RADIUS and TACACS+ for the FXOS interfaces.

Section “Logging into the Appliance” in the FTD Admin Guide contains instructions on logging into the local administrative interface for both FMC and FTD.

Section *Login to CLI Locally* in the FXOS Admin Guide describes how to connect to the FXOS CLI using a terminal plugged into the console port. This is sufficient to inform administrators how to ensure that the interface is local.

Section Creating the Pre-Login Banner in the FXOS Admin Guide provides instructions for configuring login banners for the CLI (console and SSH access) as well as for the Web UI.

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 that credential/login method, the evaluator shall show that providing correct I&A information results in the ability to access the system, while providing incorrect information results in denial of access.

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 and FIA_UAU_EXT.2, the evaluator shall test that the components authenticate Security Administrators as described in the TSS.

Test 1:

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

FTD:

• Logon locally to the TOE with passwords.

• Logon to the TOE via SSH with passwords.

• Logon using SSH with public/private key pairs.

FMC:

• Logon locally to the TOE with passwords.

• Logon to the TOE via SSH with passwords.

• Logon using SSH with public/private key pairs.

• Logon to the TOE via the Web UI with passwords.

FXOS:

• Logon locally to the TOE with passwords.

• Logon to the TOE via SSH with passwords.

• Logon using SSH with public/private key pairs.

• Logon to the TOE via the Web UI with passwords.

• Logon to the TOE via SSH with RADIUS authentication.

• Logon to the TOE via SSH with TACACS+ authentication.

For each of these interfaces, the evaluator conducted testing which showed successful and failed login attempts, and protected authentication feedback.

Test 2: After TOE configuration, the evaluator performed an nmap scan of the TOE looking for additional network services offered by the TOE that were not identified in the Security Target. The scan results did not show any ports open that offered services that were not identified in the ST.

Test 3: The evaluator successfully determined that the TOE's only service prior to login at the local console is that the TOE displays the TOE login banner.

Test 4: The evaluator determined that each of the TOE's components require authentication before TSF actions can be performed. As demonstrated through testing of test 1 through 3 above, the TOE requires all Security Administrators to be authenticated as described in the TSS.

2.6.6 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/ITT)

2.6.6.1 NDcPP22e:FIA_X509_EXT.1.1/ITT

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. It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the device. The evaluator shall perform the following tests for FIA_X509_EXT.1.1/ITT:

These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols.:

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function, and shall use this chain to demonstrate that the function succeeds.

Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store).

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

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

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the TOE certificate and revocation of the TOE intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. No testing is required if no revocation method is selected. 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.

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

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

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

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

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

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

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

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.

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.

(TD0527 12/2020 update applied)

These tests were performed on all ITT TOE components (FTD TLS Client and FMC/FMCv TLS server).

Test 1a: The successful validation of the cert chain with trusted CA certificates was demonstrated in FPT_ITT.1 test 1 which demonstrates a valid connection between the TOE components (FTD & FMC).

Test 1b: For the FTD TLSC, the evaluator configured the test peer with a certificate that is invalid, not chaining back to a root CA that is trusted by the FTD. The evaluator found that the FTD rejected this connection. For the FMC TLSS, this test was performed in FCS_TLSS_EXT.2.1 test 3. In this test, an imposter CA certificate is sent that does not chain back to a root CA that is trusted by the FMC. The evaluator found that the FMC rejected the connection for this reason.

Test 2: For this test, the evaluator alternately configured a test server to send an authentication certificate 1) that is valid and 2) that is expired and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and verified that the connection succeeded only if there are no expired certificates.

Test 3: Not applicable. The TOE does not support revocation checking for distributed TOE communication.

Test 4: Not applicable. The TOE does not support revocation checking for distributed TOE communication.

Test 5: For this test, the evaluator alternately configured a test server 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 TLSC TOE client to the test server and verified that the connection succeeded only if the certificate is not modified/corrupted.

Test 6: This test has been performed as part of Test 5.

Test 7: This test has been performed as part of Test 5.

Test 8: Not applicable. The TOE does not support EC certificates in the ITT channel.

2.6.6.2 NDcPP22e:FIA_X509_EXT.1.2/ITT

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/ITT. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/ITT. 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 two certificates: a self-signed root 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 one CA in the chain does not contain the basicConstraints extension. The evaluator confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

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

Test 1: For this test, the evaluator alternately configured a test server 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 then attempted to connect the TLSC TOE client to the test server and verified that the connection was rejected in each case.

Test 2: The test was performed as part of Test 1.

Component TSS Assurance Activities: The evaluator shall examine the TSS to ensure it 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). If selected, the TSS shall describe how certificate revocation checking is performed. It is not sufficient to verify the status of a X.509 certificate only when it's loaded onto the device.

Section 6.1 (FIA_X509_EXT.1/ITT) in the ST states that the validity check for the certificates takes place at session establishment and/or at time of import depending on the certificate type. For example, server certificate is checked at session establishment while CA certificate is checked at both. The TOE conforms to standard RFC 5280 for certificate and path validation (i.e., peer certificate checked for expiration, peer certificate checked if signed by a trusted CA in the trust chain, peer certificate checked for unauthorized modification, peer certificate checked for revocation). The TOE does not use certificate revocation checking for communication between TOE components.

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 describe how certificate revocation checking is performed.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide describes the TOE as a TLS client and indicates that the check of validity of the certificate takes place at session establishment. This information also applies to the TOE ITT channel. It states that the connection will fail if the server certificate does not meet any of the following criteria:

• The certificate is not signed by the CA with cA flag set to TRUE.

• The certificate is not signed by a trusted CA in the certificate chain.

• The certificate has been revoked or modified.

Note that the TOE does not use certificate revocation checking for communication between TOE components.

Component Testing Assurance Activities: None Defined

2.6.7 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/Rev)

2.6.7.1 NDcPP22e: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 not sufficient to verify the status of a X.509 certificate only when it is loaded onto the TOE. It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected). The evaluator shall perform the following tests for FIA_X509_EXT.1.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

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

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

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

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore the revoked certificate(s) used for testing shall not be a trust anchor.

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

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

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

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

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

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

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

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.

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.

(TD0527 12/2020 update applied)

Test 1a and 1b: For this test, the evaluator alternately configured the TOE to not have and then to have the trusted root CA used by the test peer to anchor all of its certificates. In each case, the evaluator then attempted a connection between the test peer and the TOE and confirmed that the connection only succeeded when the trusted root CA was properly configured forming a valid certificate chain. These tests were repeated on the FTD, and FMC and covered both the IPsec and TLS protocols.

Test 2: For this test, the evaluator alternately configured a test server to send an authentication certificate 1) that is valid and 2) that is expired and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the TOE to the test server. The connection succeeded only if there were no expired certificates. These tests were repeated on the FTD and FMC and covered both the IPsec and TLS protocols.

Test 3: For this test, the evaluator alternately configured a test server 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 TOE to the test server and confirmed that the connection only succeeded if there were no revoked certificates. This test was executed using CRL (all components) and OCSP revocation checking (FTD). These tests were repeated on the FTD and FMC and covered both the IPsec and TLS protocols.

Test 4: For this test, the evaluator alternately configured a test server 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 TOE to the test server and confirmed that the connection only succeeded if all retrieved CRLs were signed using certificates with cRLSign. This test was executed using CRL (all components) and OCSP revocation checking (FTD). OCSP was tested with an OCSP responder certificate missing the OCSP signing purpose. These tests were repeated on the FTD and FMC and covered both the IPsec and TLS protocols.

Test 5: For this test, the evaluator alternately configured a test server 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 attempted to connect the TOE to the test server and verified that the connection only succeeded if the certificate was not modified/corrupted. These tests were repeated on the FTD and FMC and covered both the IPsec and TLS protocols.

Test 6: This test was performed as part of Test 5.

Test 7: This test was performed as part of Test 5.

Test 8a and b: Only the FTD supports ECDSA certificates for IPsec and FTD TLS client. The evaluator executed a control test case with a valid ECDSA certificate chain and verified that the connection was successful. The evaluator then replaced an intermediate CA with one that has explicit curves defined. The TOE successfully rejected the invalid certificate chain.

Test 8c: The evaluator first attempted to import a certificate where the elliptic curve parameters are specified as a named curve, and signed by a trusted EC rootCA. The evaluator observed that the TOE accepted this certificate into the truststore. The evaluator then attempted to import a certificate that utilizes an explicit format version of the elliptic curve parameters, and signed by a trusted EC rootCA. The evaluator observed the TOE rejected the certificate import attempt and did not save the certificate to the truststore.

2.6.7.2 NDcPP22e: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 confirms that the TOE rejects such a certificate at one (or both) of the following points: (i) as part of the validation of the leaf certificate belonging to this chain; (ii) when attempting to add a CA certificate without the basicConstraints extension to the TOE's trust store (i.e. when attempting to install the CA certificate as one which will be retrieved from the TOE itself when validating future certificate chains).

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

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).

These tests were repeated for IPsec on FTD and for TLS on FTD and FMC.

Test 1: The evaluator configured the TOE with a certificate that did not contain the basicConstraints extension. The evaluator then attempted to make a connection between the peer devices. The connection attempt failed.

Test 2: The evaluator configured the TOE with a certificate that had the cA flag in the basicConstraints extension set to FALSE. The evaluator then attempted to make a connection between the peer devices. The connection attempt failed.

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 6.1, (FIA_X509_EXT.1/Rev) states that the validity check for the certificates takes place at session establishment and/or at time of import depending on the certificate type. For example, server certificate is checked at session establishment while CA certificate is checked at both. The TOE conforms to standard RFC 5280 for certificate and path validation (i.e., peer certificate checked for expiration, peer certificate checked if signed by a trusted CA in the trust chain, peer certificate checked for unauthorized modification, peer certificate checked for revocation).

The TOE can generate a RSA key pair that can be embedded in a Certificate Signing Request (CSR) created by the TOE. The CSR can be generated at the UI. The TOE can then send the CSR manually to a Certificate Authority (CA) for the CA to sign and issue a certificate. Once the certificate has been issued, the administrator can import the X.509v3 certificate into the TOE. Integrity of the CSR and certificate during transit are assured through the use of digital signature (signing the hash of the TOE's public key contained in the CSR and certificate). CRL is configurable and can be used for certificate revocation check (for FTP_ITC only, thus relevant only to FIA_X509_EXT.1/Rev, not relevant to FIA_X509_EXT.1/ITT as no revocation checking is used for communications between TOE components). Checking is also done for the 'basicConstraints' extension and the 'cA' flag to determine whether they are present and set to TRUE. If they are not, the CA certificate is not accepted as a trust anchor.

FMC and FXOS only support CRL, while FTD supports use of both CRL and OCSP (including verification of the OCSP signing purpose in the certificate that signs the OCSP response). FTD supports CRL for other purposes, i.e., for validation of syslog server certificates for both the TLS connections to TLS servers.

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 "Configure Syslog over TLS for FMC and FTD" in the FTD Admin Guide describes the TOE as a TLS client and indicates that the check of validity of the certificate takes place at session establishment. It states that the connection will fail if the server certificate does not meet either one of the following criteria:

- The certificate is not signed by the CA with cA flag set to TRUE.
- The certificate is not signed by a trusted CA in the certificate chain.
- The certificate Subject Alternative Name (SAN) does not match the expected hostname (i.e., reference identifier).
- The certificate has been revoked or modified.

If you choose CRLs, the system uses the same CRLs to validate both audit client certificates and HTTPS certificate to secure the HTTPS connection between the system and a web browser.

The section entitled *Certificates and Trust Points* in the FXOS Admin Guide indicates that Certificate revocation checking occurs on all certificates except self-signed Root Certificate Authorities when CRL revocation-check has been defined for a trustpoint. This section indicates that X509v3 certificates can belong to a Certificate Authority or can be an identity certificate. It explains that identity certificates are used to authenticate network peers within the TLS and IPsec protocols. This section also explains the extended key usage and key usage fields that are supported by the TOE and further states that FXOS does not enforce any other EKU.

Component Testing Assurance Activities: None Defined

2.6.8 X.509 Certificate Validation (VPNGW13:FIA_X509_EXT.1/Rev)

2.6.8.1 VPNGW13:FIA_X509_EXT.1.1/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

There are no additional evaluation activities specified for this SFR.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.6.9 X.509 Certificate Authentication (NDcPP22e:FIA_X509_EXT.2)

2.6.9.1 NDcPP22e:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.9.2 NDcPP22e: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 6.1 (FIA_X509_EXT.2(1)/FIA_X509_EXT.2(2)) in the ST states that the administrators can configure a trust chain by importing the CA certificate(s) that signed and issued the server (syslog) certificate. This will tell the TOE which CA certificate(s) to use during the validation process. If the TOE does not find the trusted root CA, the TLS connections (FTD TLS client and FTD OS TLS client) to the syslog server will fail. When the TOE cannot establish a connection for the validity check using CRL or the OCSP responder for verification, the FTD and FMC will not accept the certificate. When communicating with peers, the TOE uses the default certificate that is configured through the FMC and one that matches the peer's request. For more information, please refer to the CC Supplemental User Guide. The FXOS does not accept the certificate when the validity check using CRL cannot be completed.

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 "Configure Syslog over TLS for FMC and FTD" in the FTD Admin Guide provides instructions for the administrator to generate a Certificate Signing Request (CSR) and import the client certificate to the FTD or FMC. It also instructs the administrator to configure the system to load one or more certificate revocation lists (CRLs) in order to verify certificate status. The connection will fail if the server certificate does not meet either one of the following criteria:

• The certificate is not signed by the CA with cA flag set to TRUE.

• The certificate is not signed by a trusted CA in the certificate chain.

• The certificate Subject Alternative Name (SAN) does not match the expected hostname (i.e., reference identifier).

• The certificate has been revoked or modified.

Section "Configure Syslog over TLS for FMC and FTD" in the FTD Admin Guide states that if CRLs are chosen, the system uses the same CRLs to validate both audit client certificates and HTTPS certificate to secure the HTTPS connection between the system and a web browser. When the TOE cannot establish a connection for the validity check using CRL or the OCSP responder for verification, all FTD and FMC TLS and IPsec connections will not accept the certificate and the trusted channel will not be established. If TLS sessions fail due to inability to contact the CRL server or OCSP server (FTD only), the connectivity to the CRL or OCSP server should be restored before reattempting to establish the TLS sessions.

Section “Generating an HTTPS Server Certificate Signing Request” in the FTD Admin Guide provides instructions for the administrator to generate a certificate request through the local configuration HTTPS Certificate page. Section “Importing HTTPS Server Certificate” provides instructions for importing the HTTPS certificate.

Section “FTD Certificate Based Authentication” in the Supplement provides instructions for the administrator to install a certificate using manual enrollment. Section “FTD Certificate Map Object” provides instructions for creating a Certificate Map Object which is used to provide an association between a received certificate and a Remote Access VPN connection profile. Connection Profiles and Certificate Map objects are both part of a remote access VPN policy. If a received certificate matches the rules contained in the certificate map, the connection is "mapped", or associated with the specified connection profile. If IPsec sessions fail due to inability to contact the CRL or OCSP server, restore connectivity to the CRL or OCSP server before reattempting to establish the IPsec sessions.

Section “Certificates and Trust Points” and its many subsections in the FXOS Admin Guide contains instructions to configure the TOE to use certificates for authentication.

Section “Configure IPsec Secure Channel” in the FXOS Admin Guide suggests restoring the connection to any CRL distribution point where the validity check cannot be completed before attempting reattempting to establish the IPsec sessions. In this case, FXOS will attempt to restore the connection automatically.

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

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.

This test was performed on all TOE components and was iterated for IPsec (FTD only) and TLS.

Test: 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(1) and FIA_X509_EXT.2.2(2) in the ST.

2.6.10 X.509 Certificate Authentication (VPNGW13:FIA_X509_EXT.2)

2.6.10.1 VPNGW13:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.10.2 VPNGW13:FIA_X509_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to support its use for IPsec at a minimum. The evaluator shall ensure that all evaluation of this SFR is performed against its use in IPsec communications as well as any other supported usage.

There are no further evaluation activities specified for this SFR. Testing for FIA_X509_EXT.2 was performed against the FTD's IPsec channel.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.6.11 X.509 Certificate Requests (NDcPP22e:FIA_X509_EXT.3)

2.6.11.1 NDcPP22e:FIA_X509_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.11.2 NDcPP22e: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 “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for creating a certificate request which include establishing the fields: Country, Organization, Organization Unit and Common Name.

Section “Generating an HTTPS Server Certificate Signing Request” in the FTD Admin Guide provides instructions for creating a certificate request which include establishing the fields: Country, Organization, Organization Unit and Common Name.

Section “Adding Certificate Enrollment Objects” in the Supplement provides instructions for creating a certificate request which includes establishing the fields: Country, Organization, Organization Unit and Common Name.

The *Certificates and Trust Points* section of the FXOS Admin Guide states that FXOS supports X.509v3 certificates as defined by RFC 5280 for use in authentication of a network peer using IPsec. It also explains that FXOS can generate a certificate signing request (CSR) using FXOS. This section explains the KeyUsage and ExtendedKeyUsage fields that are required by FXOS in certificates which FXOS validates.

The *Creating a Key Ring* section in the FXOS Admin Guide describes the step-by-step procedures to create a key ring for an RSA certificate. The following section, *Creating a Certificate Request for a Key Ring* section in the FXOS Admin Guide describes the step-by-step procedures to specify the fields of a CSR using that key ring.

The *Certificates and Trust Points* section of the FXOS Admin Guide states that the administrator can configure FXOS to make CRL checks mandatory when authenticating a certificate by using the revocation-check crl command. This configuration is required to be in a Common Criteria certified configuration

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.

These tests were performed on the FTD and FMC.

Test 1- The evaluator followed operational guidance to log into the TOE and then generate the CSR. 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.

2.6.12 X.509 Certificate Requests (VPNGW13:FIA_X509_EXT.3)

2.6.12.1 VPNGW13:FIA_X509_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.12.2 VPNGW13:FIA_X509_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

There are no further evaluation activities specified for this SFR.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7 Security management (FMT)

2.7.1 Management of security functions behaviour (NDcPP22e:FMT_MOF.1/ManualUpdate)

2.7.1.1 NDcPP22e: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 it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

There are no specific requirements for non-distributed TOEs.

All security management functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement and components to which they apply.

Section 6.1 (FMT_MOF.1/ManualUpdate) in the ST states that the TOE restricts the ability to enable, disable, determine and modify the behavior of all of the security functions of the TOE to authorized administrators. The TOE provides the ability for authorized administrators to initiate TOE updates and access TOE data, such as audit data, configuration data, security attributes, information flow rules, and session thresholds.

For the FMC, only accounts with 'administrator' privilege can upload patches to FMC and initiate installation of patches to FMC or FTD devices (the FMC WebUI is used to configure FMC and FTD).

For FXOS, only accounts with 'admin' role can upload software updates to FXOS and initiate updates of FXOS.

Section 6.1 (FMT_SMF.1) in the ST identifies the TOE security functions necessary to administer the TOE locally and remotely via the administrative interfaces of the FTD (SSH CLI, local console) and FMC (WebUI, SSH CLI, local console) and FXOS (SSH CLI, local console, Web UI).

Section 9 "Annex B: SFR TOE Components Mapping" in the ST provides a mapping of the distributed TOE components to the SFRs in this ST.

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 "Product Upgrade" in the FTD Admin Guide provides instructions for manually updating the TOE components (FMC, FTD). It states that the FMC must be updated before the managed devices (i.e. FTD). It also provides warnings regarding capabilities that might be affected when an administrator installs an update from a managed device. The following capabilities may be affected: Traffic inspection and connection logging, Traffic flow including switching, routing, and related functionality and Link state.

Section *Image Management* in the FXOS Admin Guide describes upgrading the platform bundle, downloading the evaluated software image file from Cisco.com onto a trusted computer system and verifying the downloaded image using the "verify" command.

Component Testing Assurance Activities: 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.

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 - There is no interface to perform an update prior to authentication. The interfaces to perform an update are only available to administrators that have successfully authenticated to the TOE. During test of FIA_UIA_EXT.1, the evaluator observed that prior to authentication, there are no services available to a user aside from viewing the login banner. This includes the lack of any ability to update the TOE. Therefore, any attempt to update the TOE fails prior to authentication as a Security Administrator.

Test 2 - A successful update with prior authentication as a Security Administrator was performed as part of FPT_TUD_EXT.1.

2.7.2 Management of Security Functions Behaviour (NDcPP22e:FMT_MOF.1/Services)

2.7.2.1 NDcPP22e:FMT_MOF.1.1/Services

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

From chapter 2.4.1.1): For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

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

See FMT_MOF.1.1/ManualUpdate.

The TOE is distributed.

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

From chapter 2.4.1.1): For distributed TOEs it is required to verify the Guidance Documentation to describe management of each TOE component. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

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

See FMT_MOF.1.1/ManualUpdate.

The TOE is distributed.

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

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

Test 1 - There is no interface to start or stop any services prior to authentication. The interfaces to start and stop services are only available to administrators that have successfully authenticated to the TOE. In testing of FMT_MOF.1/ManualUpdate, the evaluator logged in as a user without administrator privileges and confirmed that, in addition to there being no ability to update the TOE, there are also no operations available to start or stop services.

Test 2 – Testing throughout FCO_CPC_EXT.1 was performed while authenticated as a security administrator. In FCO_CPC_EXT.1.1 the evaluator attempted to enable ITT services allowing the FTD and FMC to communicate and found that this was successful. In FCO_CPC_EXT.1.2, the evaluator tested disabling the same services and found that this was also successful. For FXOS, the enabling or disabling of services as a security administrator was demonstrated by the audits collected in FAU_GEN.1 for stopping and starting the auditing services.

2.7.3 Management of TSF Data (NDcPP22e:FMT_MTD.1/CoreData)

2.7.3.1 NDcPP22e:FMT_MTD.1.1/CoreData

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

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.

All security management functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

Section 6.1 (FMT_MTD.1/CoreData) in the ST states that for FTD and FMC, the TOE provides a web-based GUI (using HTTPS) management interface and CLI or shell (using SSH or serial connection) for all TOE administration, including the policy rule sets, user accounts and roles, and audit functions. The FMC provides the Web GUI and CLI, while FTD provides a CLI. The ability to manage various security attributes, system parameters and all TSF data is controlled and limited to those users who have been assigned the appropriate administrative role and privileges associated with those roles. The TOE supports the predefined Administrator role. Users with the Administrator role have access to all TOE security functions including handling of X.509v3 certificates and management of the trust store.

User accounts are used to access the FXOS system through the FXOS WebUI and CLI. Up to 48 local user accounts can be configured. Each user account must have a unique username and password. The ‘admin’ account is a default user account and cannot be modified or deleted. This account is the system administrator or superuser account and has full privileges. The term “authorized administrator” or “Security Administrator” applies to this account and other accounts assigned to the Administrator role.

Section 9 “Annex B: SFR TOE Components Mapping” in the ST provides a mapping of the distributed TOE components to the SFRs in this ST.

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.

The TSF data manipulating functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

Section “User and Role Management” in the FTD Admin Guide provides instructions for creating user accounts and states that all users created are administrators. User accounts without the Administrator role are restricted from accessing user management functions. Similarly, section “Configure Authentication” in the FTD Admin Guide describes the Administrator role which has complete read and write access to the entire system. This is consistent with the ST which indicates that the TOE supports the predefined Administrator role. Users with the Administrator role have access to all TOE security functions including importing X.509v3 certificates to the TOE’s trust store.

The section *Changing the Management IP Address* in the FXOS Admin Guide details configuration of a management interface and disabling appropriate services in order to securely manage the TOE.

See NDcPP22e:FIA_X509_EXT.2.2 which identifies the sections in the Guide(s) which describe how the administrator role can configure and maintain the trust store including loading of CA certificates.

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.

All management functions are exercised under other SFRs.

2.7.4 Management of TSF Data (NDcPP22e:FMT_MTD.1/CryptoKeys)

2.7.4.1 NDcPP22e: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 chapter 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 how that how those operations are performed.

All security management functions for each TOE component are identified and described in the TSS Assurance Activities throughout this AAR with the requirement to which they apply. The evaluator confirmed that all relevant aspects of each TOE component are covered by the FMT SFRs.

Section 6.1 (FMT_MTD.1/CryptoKeys) in the ST states that the TOE only provides the ability for authorized administrators to access TOE data, such as audit data, configuration data, security attributes (such as cryptographic keys and certificates used in VPN), routing tables, and session thresholds.

Section 6.1 (FCS_CKM.4) in the ST states that the TOE is designed to zeroize secret and private keys when they are no longer required by the TOE. The table in section 7.3 identifies the applicable secret and private keys and summarizes how they are deleted.

Section 9 “Annex B: SFR TOE Components Mapping” in the ST provides a mapping of the distributed TOE components to the SFRs in this ST.

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

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

All security management functions and the corresponding configuration information for each TOE component are identified in the Guidance Assurance Activities throughout this AAR with the requirement to which they apply. The evaluator confirmed that all relevant aspects of each TOE component are covered by the FMT SFRs.

Section “Configure Syslog over TLS for FMC and FTD” in the FTD Admin Guide provides instructions for configuring TLS communications between FMC and an external syslog server and FTD and an external syslog server.

Section “Device Registration” in the FTD Admin Guide provides instructions for configuring the communication between the FMC and FTD. The same ciphersuites are used by the TLS client and TLS server during device registration as are used during subsequent inter-device communications. These ciphersuites are listed in this section and are consistent with those specified in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 in the ST.

Section “Custom Web Server Certificate” in the FTD Admin Guide provides instructions for generating an HTTPS Server Certificate signing request and importing an HTTPS certificate on the FMC.

Section “Configure SSH Public-Key authentication” in the FTD Admin Guide provides the steps for generating an SSH keypair to be used for SSH public key authentication for both the FMC and the FTD device.

Section “Configure SSH Rekey Configuration” in the FTD Admin Guide provides instructions for configuring the SSH rekey parameters.

Section “PKI Infrastructure” in the Supplement describes the PKI framework for FTD IPsec communication and indicates that an RSA or ECDSA key pair can be generated and used for both signing and encryption, or separate keys can be generated for each purpose. TLS uses a key for encryption but not signing, however, IKE uses a key for signing but not encryption.

Section “Adding Certificate Enrollment Objects” in the Supplement provides steps for enrolling a certificate and specifying the certificate contents including the Key Type: RSA or ECDSA for FTD certificate-based authentication.

Section “VPN Basics” in the Supplement states that the secret keys used for symmetric encryption, private keys, and CSPs used to generate keys, are zeroized immediately after use (for IPsec VPN functions, within FTD only), or on system shutdown (for all other functions).

The section entitled *Generate the SSH Host Key* in the FXOS Admin Guide describes how to generate and zeroize a secret RSA key for used by the FXOS SSH Server.

Sections *Deleting a Key Ring* and *Deleting a Trust Point* in the FXOS Admin Guide provide instructions for deleting a key or a trust point.

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

The evaluator attempted to modify the certificate store while logged in as a user with no administrative rights and verified that the attempts failed. The evaluator successfully modified the certificate store while logged in as an administrator in FIA_X509_EXT.3.

2.7.5 Management of TSF Data (VPNGW13:FMT_MTD.1/CryptoKeys)

2.7.5.1 VPNGW13:FMT_MTD.1.1/CryptoKeys

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

There are no evaluation activities specified. Refer to NDcPP22e:FMT_MTD.1/CryptoKeys where the NDcPP22e activity has been performed and covers the VPN cryptographic data.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.6 Specification of Management Functions - per TD0631 (NDcPP22e:FMT_SMF.1)

2.7.6.1 NDcPP22e: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.

See the other requirements in this AAR as referenced. All security management functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

Section 6.1 (FMT_SMF.1) in the ST identifies the TOE security functions necessary to administer the TOE locally and remotely and maps them to the administrative interfaces through which they are available including the FTD (SSH CLI, local console), FMC (WebUI, SSH CLI, local console) and FXOS (WebUI, SSH CLI and local console). All the management functions that are available to be performed on the TOE local console can also be performed remotely via SSH. No access or service is provided prior to identification and authentication, beyond viewing the login banner.

The following sections in the referenced guidance documentation describe the TOE’s local administrative interfaces (local serial console) and clearly indicate how the console is configured to ensure that it is accessed locally:

Section “Logging into the Appliance” in the FTD Admin Guide contains instructions on logging into the local administrative interface for both FMC and FTD.

Section *Login to CLI Locally* in the FXOS Admin Guide describes how to connect to the FXOS CLI using a terminal plugged into the console port. This is sufficient to inform administrators how to ensure that the interface is local.

Refer to NDcPP22e:FCO_CPC_EXT.1.3 where the ability to configure the interaction between TOE components is described. The evaluator confirmed during testing that the TOE behavior is as described in the TSS and Guidance Documentation.

Component Guidance Assurance Activities: See TSS Assurance Activities

See TSS Assurance Activities.

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

All of the management functions were demonstrated throughout the course of testing.

2.7.7 Specification of Management Functions (STFFW14e:FMT_SMF.1/FFW)

2.7.7.1 STFFW14e:FMT_SMF.1.1/FFW

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, 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.

Section 6.1 (FMT_SMF.1/FFW) in the ST identifies the TOE security functions necessary to administer the TOE locally and remotely and maps them to the administrative interfaces through which they are available. This includes the management functions specified in FMT_SMF.1/FFW which are managed via the FMC interfaces.

Refer to NDcPP22e:FMT_SMF.1 for the guidance documentation reference describing the local administrative interface and the configuration of the interaction between TOE components.

Refer to NDcPP22e:FCO_CPC_EXT.1.3 where the ability to configure the interaction between TOE components is described. The evaluator confirmed during testing that the TOE behavior is as described in the TSS and Guidance Documentation.

Component Guidance Assurance Activities: See TSS Activity.

See TSS Activity.

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

All of the management functions were demonstrated throughout the course of testing.

2.7.8 Specification of Management Functions (IPS) (IPS10:FMT_SMF.1/IPS)

2.7.8.1 IPS10:FMT_SMF.1.1/IPS

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 describes how the IPS data analysis and reactions can be configured. This may be performed in conjunction with the evaluation of IPS_ABD_EXT.1, IPS_IPB_EXT.1, and IPS_SBD_EXT.1.

Section 6.1 (FMT_SMF.1/IPS) in the ST states the Administrators can deploy intrusion policy with intrusion rules to any interface. An interface, however, can only have one policy applied to that interface. The Administrators can also import vendor-defined signatures from Cisco, create their own intrusion rules, create rules to define which traffic is inspected and analyzed, enable anomaly rules/detections, modify thresholds and threshold duration, and configure Block List/Do Not Block List. The IPS Analysts (Intrusion Admins) Administrators can create, modify, or delete intrusion policies but only the IPS Administrators can deploy the policies.

For additional details, see IPS_ABD_EXT.1, IPS_IPB_EXT.1 and IPS_ABD_EXT.1.

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance describes the instructions for each function defined in the SFR, describes how to configure the IPS data analysis and reactions, including how to set any configurable defaults and how to configure each of the applicable analysis pattern matching methods and reaction modes

The requirement defines the following functions for which instructions can be found in the sections identified in the Supplement:

• Enable, disable signatures applied to sensor interfaces, and determine the behavior of IPS functionality - section “Intrusion Rule Actions”.

• Modify these parameters that define the network traffic to be collected and analyzed: Section “Access Control Policy”, section “Intrusion Rules Editor” and section “Intrusion Rules Import”.

o Source IP addresses (host address and network address)

o Destination IP addresses (host address and network address)

o Source port (TCP and UDP)

o Destination port (TCP and UDP)

o Protocol (IPv4 and IPv6)

o ICMP type and code

• Update (import) signatures - section “Intrusion Rules Import”.

• Create custom signatures - section “Writing New Rules”.

• Configure anomaly detection - section “Rate-Based Attack Prevention”.

• Enable and disable actions to be taken when signature or anomaly matches are detected - section “Intrusion Rule Actions”.

• Modify thresholds that trigger IPS reactions - section “Adding and Modifying Intrusion Event Thresholds”.

• Modify the duration of traffic blocking actions - section “Create Intrusion Policy”, section “Adding and Modifying Intrusion Event Thresholds”.

• Modify the known-good and known-bad lists (of IP addresses or address ranges) - Section “Configure Security Intelligence”.

• Configure the known-good and known-bad lists to override signature-based IPS policies - Section “Configure Security Intelligence”.

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

Test 1: The evaluator shall use the operational guidance to create a signature and enable it on an interface. The evaluator shall then generate traffic that would be successfully triggered by the signature. The evaluator should observe the TOE applying the corresponding reaction in the signature.

Test 2: The evaluator shall then disable the signature and attempt to regenerate the same traffic and ensure that the TOE allows the traffic to pass with no reaction.

Test 3: The evaluator shall use the operational guidance to import signatures and repeat the test conducted in Test 1.

Other testing for this SFR is performed in conjunction with the EAs for IPS_ABD_EXT.1 and IPS_SBD_EXT.1.

Test 1: This test was performed in conjunction with part of IPS10:IPS_SBD_EXT.1.2-t1 testing. The evaluator created a custom rule duplicating an imported rule and observed that matching traffic triggered both of the rules, with the TOE enforcing the configured reaction to the rule (i.e. generating an event and blocking the traffic).

Test 2: The evaluator disabled the “String” rules used during testing of IPS_SBD_EXT.1.2-t1 and re-ran IPS_SBD_EXT.1.2-t1 Part 3 (ICMPv4 string). Test results show IPS_SBD_EXT.1.2-t1

Part 3 failing because the blocking of traffic did not occur.

Test 3: This test was performed as part of Test 1.

2.7.9 Specification of Management Functions (VPNGW13:FMT_SMF.1/VPN)

2.7.9.1 VPNGW13:FMT_SMF.1.1/VPN

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

Section 6.1 of the ST, FMT_SMF.1/VPN identifies the TOE security functions necessary to administer the TOE locally and remotely and maps them to the administrative interfaces through which they are available. This includes the management functions specified in FMT_SMF.1/VPN which are managed via the FMC interfaces.

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

This activity has been performed in VPNGW13:FPF_RUL_EXT.1.4, VPNGW13:FPF_RUL_EXT.1.5 and NDcPP2e:FMT_SMF.1 where the evaluator verified that the guidance describes and provides instructions for configuring all management functions specified in FMT_SMF.1/VPN.

Component Testing Assurance Activities: The evaluator tests management functions as part of performing other test EAs. No separate testing for FMT_SMF.1/VPN is required unless one of the management functions in FMT_SMF.1.1/VPN has not already been exercised under any other SFR.

This requirement is met throughout the testing of all other VPNGW13 requirements. All of the management functions have been exercised in other SFRs.

2.7.10 Restrictions on Security Roles (NDcPP22e:FMT_SMR.2)

2.7.10.1 NDcPP22e:FMT_SMR.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.10.2 NDcPP22e:FMT_SMR.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.10.3 NDcPP22e: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.

Section 6.1 (FMT_SMR.2) in the ST states that the TOE (FTD and FMC) includes one evaluated role which corresponds to the required 'Security Administrator' role described in the requirement.

The ST further states that FXOS contains only the Administrator role which also corresponds to the required 'Security Administrator'

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 in the FTD & FXOS Admin Guides 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; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

All administrative interfaces were used throughout the course of testing including console for all devices, SSH and HTTPS for FMC and FMCv, and SSH for the FTD/FTDv. The evaluator also tested these communication channels by applying the FCS_SSHS and FCS_TLSS tests to the relevant channels on the FMC/FMCv and FTD/FTDv. The FTD can be set up to optionally support SSH tunneling over IPSec. The evaluator applied the FCS_IPSEC testing to the FTD.

2.8 Packet Filtering (FPF)

2.8.1 Packet Filtering Rules (VPNGW13:FPF_RUL_EXT.1)

2.8.1.1 VPNGW13:FPF_RUL_EXT.1.1

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

The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Section 6.1 (FPF_RUL_EXT.1) in the ST states that during the boot cycle, the TOE first powers on hardware, loads the image, and executes the power on self-tests. Until the power on self-tests successfully complete, the interfaces to the TOE are deactivated. Once the tests complete, the interfaces become active and the rules associated with the interface become immediately operational. There is no state during initialization/ startup that the access lists are not enforced on an interface.

During initialization/startup (while the TOE is booting) the configuration has yet to be loaded, and no traffic can flow through any of its interfaces. No traffic can flow through the TOE interfaces until the POST has completed, and the configuration has been loaded. If any aspect of the POST fails during boot, the TOE will reload without forwarding traffic. If a critical component of the TOE, such as the clock or cryptographic modules, fails while the TOE is in an operational state, the TOE will reload, which stops the flow of traffic. If a component such as a network interface, which is not critical to the operation of the TOE, but may be critical to one or more traffic flows, fails while the TOE is operational, the TOE will continue to function, though all traffic flows through the failed network interface(s) will be dropped.

Guidance Assurance Activities: The operational guidance associated with this requirement is assessed in the subsequent test EAs.

See subsequent test evaluation activities.

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

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

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

Test 1: Because the TOE acts as a router, packets are sent to a TOE interface and forwarded. Thus, traffic used during STFFW14e:FFW_RUL_EXT.1-t1 testing is directed at TOE interfaces (per VPNGW13:FPF_RUL_EXT.1.1-t1) to be forwarded as appropriate to the ultimate destination. Therefore, the tests for Packet Filtering from the VPNGW module are a subset of those identified by the FW module and thus are covered by tests described under STFFW14e:FFW_RUL_EXT.1.

Test 2: This was performed as part of testing for STFFW14e:FFW_RUL_EXT.1-t1.

2.8.1.2 VPNGW13:FPF_RUL_EXT.1.2

TSS Assurance Activities: There are no EAs specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

There are no evaluation activities specified for this element. See FPF_RUL_EXT.1.4.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.8.1.3 VPNGW13:FPF_RUL_EXT.1.3

TSS Assurance Activities: There are no EAs specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

There are no evaluation activities specified for this element. See FPF_RUL_EXT.1.4.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.8.1.4 VPNGW13:FPF_RUL_EXT.1.4

TSS Assurance Activities: The evaluator shall verify that the TSS describes a packet filtering policy that can use the following fields for each identified protocol, and that the RFCs identified for each protocol are supported:

- IPv4 (RFC 791)

o source address

o destination address

o Protocol

- IPv6 (RFC 8200)

o source address

o destination address

o next header (protocol)

- TCP (RFC 793)

o source port

o destination port

- UDP (RFC 768)

o source port

o destination port

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

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

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

Section 6.1 (FPF_RUL_EXT.1) in the ST states that an authorized administrator can define the traffic that needs to be protected by configuring access lists (permit, deny, log) and applying these access lists to interfaces using access and crypto map sets. Therefore, traffic may be selected on the basis of the source and destination address, and optionally the Layer 4 protocol and port.

The TOE (FTD) enforces information flow policies on network packets that are received by TOE interfaces and leave the TOE through other TOE interfaces. When network packets are received on a TOE interface, the TOE verifies whether the network traffic is allowed or not and performs one of the following actions, pass/not pass information, as well as optional logging.

The TOE implements rules that define the permitted flow of traffic between interfaces of the TOE for unauthenticated traffic. These rules control whether a packet is transferred from one interface to another based on:

1. Presumed address of source
2. Presumed address of destination
3. Transport layer protocol (or next header in IPv6)
4. Service used (UDP or TCP ports, both source and destination)
5. Network interface on which the connection request occurs

These rules are supported for the following protocols: RFC 791(IPv4); RFC 2460 (IPv6); RFC 793 (TCP); RFC 768 (UDP). FTD compliance with these protocols is verified via regular quality assurance, regression, and interoperability testing.

Section 6.1 (FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4) in the ST further describes the interface types subject to the Access Control Lists (ACLs). ACLs are only enforced after they've been applied to a network interface. Any network interface can have an ACL applied to it. Interfaces can be referred to by their identifier (e.g. GigabitEthernet 0/1).

The interface types that can be assigned to an interface are:

• Physical interfaces

 o Ethernet

   o GigabitEthernet

   o TenGigabitEthernet

   o Management

- Port-channel interfaces (designated by a port-channel number)

- Subinterface (designated by the subinterface number)

Guidance Assurance Activities: The evaluators shall verify that the operational guidance identifies the following protocols as being supported and the following attributes as being configurable within packet filtering rules for the associated protocols:

- IPv4 (RFC 791)

o source address

o destination address

o Protocol

- IPv6 (RFC 8200)

o source address

o destination address

o next header (protocol)

- TCP (RFC 793)

o source port

o destination port

- UDP (RFC 768)

o source port

o destination port

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

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

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

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that traffic policies are defined in terms of network zones which in turn are associated with FTD interfaces. So, a policy may be defined to allow traffic from “zone0” to “zone1”, though those zones may be mapped to the “outside” and “inside” interfaces on one FTD and also be mapped to “int1” and “int2” interfaces of another FTD which enforces the same policy. No FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it’s explicitly permitted by at least one policy rule, thus an implicit “deny-all” rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section “Configure Access Control Policy” in the Supplement states that for each rule, the administrator can specify a rule action, to trust, block or inspect matching traffic with an intrusion policy.

Section “Stateful Session Behaviors” of the Supplement contains a table specifying the types of packets that are decoded and can be configured upon which includes IPv4, IPv6, ICMPv4, ICMPv6, TCP and UDP packets.

Section “Intrusion Rules Editor” in the Supplement contains a table that outlines each protocol type, the required header field inspection and the specific keywords that can be used to configure a rule. For IPv4 and IPv6 the required header fields include the source and destination address and transport protocol. For ICMPv4 and ICMPv6 it includes the type and code, and for TCP and UDP it includes the source and destination port.

The “Scope of Evaluation” section in the FTD Admin Guide indicates that any features not associated with SFRs in the claimed NDcPP and PP modules are not considered part of the TOE evaluation.

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

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

- IPv4

o Source address

o Destination Address

o Protocol

- IPv6

o Source Address

o Destination Address

o Next Header (Protocol)

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

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

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

The VPNGW13:FPF_RUL_EXT.1 testing is a subset of testing required for the STFFW14e:FFW_RUL_EXT.1 testing. Refer to the STFFW14e:FFW_RUL_EXT.1.2-t1 & STFFW14e:FFW_RUL_EXT.1.2-t2 for actual results. Also, refer to VPNGW13:FPF_RUL_EXT.1.6 for test results on various packet filtering scenarios.

2.8.1.5 VPNGW13:FPF_RUL_EXT.1.5

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

Section 6.1 (FPF_RUL_EXT.1) in the ST states that the TOE (FTD) enforces information flow policies on network packets that are received by TOE interfaces and leave the TOE through other TOE interfaces. When network packets are received on a TOE interface, the TOE verifies whether the network traffic is allowed or not and performs one of the following actions, pass/not pass information, as well as optional logging.

The TOE implements rules that define the permitted flow of traffic between interfaces of the TOE for unauthenticated traffic. These rules control whether a packet is transferred from one interface to another based on:

1. Presumed address of source

2. Presumed address of destination

3. Transport layer protocol (or next header in IPv6)

4. Service used (UDP or TCP ports, both source and destination)

5. Network interface on which the connection request occurs

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

Packets will be dropped unless a specific rule has been set up to allow the packet to pass (where the attributes of the packet match the attributes in the rule and the action associated with the rule is to pass traffic). Rules are enforced on a first match basis from the top down. As soon as a match is found the action associated with the rule is applied.

These rules are entered in the form of access lists at the CLI (via 'access list' and 'access group' commands). These interfaces reject traffic when the traffic arrives on an external TOE interface, and the source address is an external IT entity on an internal network;

These interfaces reject traffic when the traffic arrives on an internal TOE interface, and the source address is an external IT entity on the external network;

These interfaces reject traffic when the traffic arrives on either an internal or external TOE interface, and the source address is an external IT entity on a broadcast network;

These interfaces reject traffic when the traffic arrives on either an internal or external TOE interface, and the source address is an external IT entity on the loopback network;

These interfaces reject requests in which the subject specifies the route for information to flow when it is in route to its destination; and

For application protocols supported by the TOE (e.g., DNS, HTTP, SMTP, and POP3), these interfaces deny any access or service requests that do not conform to its associated published protocol specification (e.g., RFC). This is accomplished through protocol filtering proxies that are designed for that purpose.

Otherwise, these interfaces pass traffic only when its source address matches the network interface originating the traffic to the network interface corresponding to the traffic's destination address.

Section 6.1 (FFW_RUL_EXT.1.5) in the ST states that if it is a new connection the TOE has to check the packet against access control lists and perform other tasks to determine if the packet is allowed or denied. If the connection is already established, the TOE does not need to re-check packets against the ACL; matching packets can go through the "fast" path based on attributes identified in FFW_RUL_EXT.1.5. For TCP, FFW_RUL_EXT.1.5 identifies the attributes: source and destination addresses, source and destination ports, sequence number, and individual flags. For UDP, FFW_RUL_EXT.1.5 identifies the attributes: source and destination addresses and source and destination ports.

The proper session establishment and termination followed by the TOE is as defined in the following RFCs:

• RFC 792 (ICMPv4)

• RFC 4443 (ICMPv6)

• RFC 791 (IPv4)

• RFC 8200 (IPv6)

• TCP, RFC 793, section 2.7 Connection Establishment and Clearing

• UDP, RFC 768 (not applicable, UDP is a “stateless” protocol)

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

Section “Configure Access Control Policy” in the Supplement provides instructions for configuring the access control policy and access control rules. It states that the system matches traffic to access control rules in order; the first matched rule handles the traffic.

Section “Access Control Rule” in the Supplement states that within an access control policy, the system matches traffic to rules in top-down order by rule number. Section “Creating and Editing Access Control Rules’ provides instructions for adding new rules or editing existing rules including specifying the rule position.

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

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

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

The VPNGW13:FPF_RUL_EXT.1 testing is a subset of testing required for the STFFW14:FFW_RUL_EXT.1 testing. Refer to the STFFW14:FFW_RUL_EXT.1.8-t1 & STFFW14:FFW_RUL_EXT.1.8-t2 for actual results.

2.8.1.6 VPNGW13:FPF_RUL_EXT.1.6

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

Section 6.1 (FPF_RUL_EXT.1) in the ST states that the TOE (FTD) enforces information flow policies on network packets that are received by TOE interfaces and leave the TOE through other TOE interfaces. When network packets are received on a TOE interface, the TOE verifies whether the network traffic is allowed or not and performs one of the following actions, pass/not pass information, as well as optional logging.

Packets will be dropped unless a specific rule has been set up to allow the packet to pass (where the attributes of the packet match the attributes in the rule and the action associated with the rule is to pass traffic). Rules are enforced on a first match basis from the top down. As soon as a match is found the action associated with the rule is applied.

The TOE supports all IPv4 protocols excluding Protocol 2 (IGMP) which is not routable and thus will not be forwarded by the TOE.

The TOE supports the following 17 IPv6 protocols:

- Transport Layer Protocol 4 - IPv4 encapsulation
- Transport Layer Protocol 6 - Transmission Control
- Transport Layer Protocol 8 - Exterior Gateway Protocol
- Transport Layer Protocol 9 - any private interior gateway
- Transport Layer Protocol 17 - User Datagram
- Transport Layer Protocol 41 - IPv6 encapsulation
- Transport Layer Protocol 46 - Reservation Protocol
- Transport Layer Protocol 47 - General Routing Encapsulation

- Transport Layer Protocol 49 - BNA
- Transport Layer Protocol 58 - ICMP for IPv6
- Transport Layer Protocol 59 - No Next Header for IPv6
- Transport Layer Protocol 88 - TCF
- Transport Layer Protocol 89 - EIGRP
- Transport Layer Protocol 103 - Protocol Independent Multicast
- Transport Layer Protocol 105 - SCPS Transport Layer Protocol
- Transport Layer Protocol 112 - Virtual Router Redundancy Protocol
- Transport Layer Protocol 132 - Stream Control Transmission Protocol

The TOE supports all IPv6 protocols excluding Protocol 2 (IGMP) which is not routable and thus will not be forwarded by the TOE. All other IPv6 protocols from the RFC Values for IPv4 and IPv6 table in the MOD VPNGW SD v1.3 are dropped by default by the TOE.

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

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that no FTD interface will forward traffic until policies have been configured and applied to that interface. Traffic will not be forwarded unless it’s explicitly permitted by at least one policy rule, thus an implicit “deny-all” rule is applied to all interfaces to which any traffic filtering rule has been applied. The implicit deny-all rule is executed after all admin-defined rules have been executed, and will result in dropping all traffic that has not been explicitly permitted, or explicitly denied. If an administrator wants to log all denied traffic, a rule entry should be added that denies all traffic and logs it, e.g. by either adding a rule at the end of a policy to explicitly drop and log all traffic, or by setting the Default Action for the policy to block all traffic, and enabling logging for the default rule.

Section “Access Control Rule” in the Supplement provides instructions for creating and editing access control rules including using the logging option and specifying the rule’s actions which can be allow or block.

Section “Managing Intrusion Policies” in the Supplement states that an enabled rule causes the system to generate intrusion events for (and optionally block) traffic matching the rule. Section “Create Intrusion Policy” outlines the process for the administrator to create a new intrusion policy and includes the settings for specifying if rules should drop packets and log or simply log events that trigger the policy.

Section “Intrusion Rule Actions” provides instructions for enabling or disabling a rule within an individual intrusion policy as well as specifying the action the system takes if monitored conditions trigger the rule. The rule’s state can be set to Alert (which only generates events) or Block (which blocks the packet and generates an event).

Section “Access Control Policies (ACP)” in the FTD Admin Guide describes the range of IPv4/IPv6 protocols supported by the TOE.

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

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

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

Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

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

Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

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

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

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

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

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

The following table identifies the RFC defined values for the protocol fields for IPv4 and IPv6 to be used in configuring and otherwise testing Packet Filtering rule definition and enforcement:

Test 1: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log each defined IPv4 Transport Layer Protocol in multiple parts as described below:

- Test 1-1: IPv4 Protocol permitted and logged based on specific source and destination addresses.
- Test 1-2: IPv4 Protocol permitted and logged based on specific destination addresses.
- Test 1-3: IPv4 Protocol permitted and logged based on specific source addresses.
- Test 1-4: IPv4 Protocol permitted and logged based on wildcard addresses.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 2: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit all traffic except to deny and log each defined IPv4 Transport Layer Protocol in multiple parts as described below:

- Test 2-1: IPv4 Protocol all permitted with some denied and logged based on specific source and destination addresses.

- Test 2-2: IPv4 Protocol all permitted with some denied and logged based on specific destination addresses.
- Test 2-3: IPv4 Protocol all permitted with some denied and logged based on specific source addresses.
- Test 2-4: IPv4 Protocol all permitted with some denied and logged based on wildcard addresses.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 3: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log and to deny and log each defined IPv4 Transport Layer Protocol. Then the evaluator generated packets using these protocols which do not match any configured rule (i.e., which have source and destination address outside the scope of the configured rules) and thus match the default deny rule.

- Test 3-1: IPv4 Protocol some permitted and logged and some denied and logged based on addresses outside the scope of the configured rules (matching the default deny rule)

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 4: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log and to deny and log each defined IPv6 Transport Layer Protocol in multiple parts as described below:

- Test 4-1: IPv6 Protocol permitted and logged based on specific source and destination addresses.
- Test 4-2: IPv6 Protocol permitted and logged based on specific destination addresses.
- Test 4-3: IPv6 Protocol permitted and logged based on specific source addresses.
- Test 4-4: IPv6 Protocol permitted and logged based on wildcard addresses.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 5: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit all traffic except to deny and log each defined IPv6 Transport Layer Protocol in multiple parts as described below:

- Test 5-1: IPv6 Protocol all permitted with some denied and logged based on specific source and destination addresses.
- Test 5-2: IPv6 Protocol all permitted with some denied and logged based on specific destination addresses.
- Test 5-3: IPv6 Protocol all permitted with some denied and logged based on specific source addresses.
- Test 5-4: IPv6 Protocol all permitted with some denied and logged based on wildcard addresses.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 6: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log and to deny and log each defined IPv6 Transport Layer Protocol. Then the evaluator generated packets using these protocols which do not match any configured rule (i.e., which have source and destination address outside the scope of the configured rules) and thus match the default deny rule.

- Test 6-1: IPv6 Protocol some permitted and logged and some denied and logged based on addresses outside the scope of the configured rules (matching the default deny rule)

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 7: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log traffic in multiple parts as described below:

- Test 7-1: TCP (IPv4) permitted and logged based on source port.
- Test 7-2: TCP (IPv4) permitted and logged based on destination port.
- Test 7-3: TCP (IPv4) permitted and logged based on source and destination port.
- Test 7-4: TCP (IPv6) permitted and logged based on source port.
- Test 7-5: TCP (IPv6) permitted and logged based on destination port.
- Test 7-6: TCP (IPv6) permitted and logged based on source and destination port.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 8: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to deny and log traffic in multiple parts as described below:

- Test 8-1: TCP (IPv4) denied and logged based on source port.
- Test 8-2: TCP (IPv4) denied and logged based on destination port.
- Test 8-3: TCP (IPv4) denied and logged based on source and destination port.
- Test 8-4: TCP (IPv6) denied and logged based on source port.
- Test 8-5: TCP (IPv6) denied and logged based on destination port.
- Test 8-6: TCP (IPv6) denied and logged based on source and destination port.

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 9: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to permit and log traffic in multiple parts as described below:

- Test 9-1: UDP (IPv4) permitted and logged based on source port.
- Test 9-2: UDP (IPv4) permitted and logged based on destination port.
- Test 9-3: UDP (IPv4) permitted and logged based on source and destination port (includes UDP port 500)
- Test 9-4: UDP (IPv6) permitted and logged based on source port.
- Test 9-5: UDP (IPv6) permitted and logged based on destination port.
- Test 9-6: UDP (IPv6) permitted and logged based on source and destination port (includes UDP port 500)

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Test 10: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured the TOE to deny and log traffic in multiple parts as described below:

- Test 10-1: UDP (IPv4) denied and logged based on source port.

- Test 10-2: UDP (IPv4) denied and logged based on destination port.
- Test 10-3: UDP (IPv4) denied and logged based on source and destination port (includes UDP port 500)
- Test 10-4: UDP (IPv6) denied and logged based on source port.
- Test 10-5: UDP (IPv6) denied and logged based on destination port.
- Test 10-6: UDP (IPv6) denied and logged based on source and destination port (includes UDP port 500)

The evaluator verified that the TOE was able to filter the packets as expected and described by each of the specified tests.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9 Protection of the TSF (FPT)

2.9.1 Protection of Administrator Passwords (NDcPP22e:FPT_APW_EXT.1)

2.9.1.1 NDcPP22e:FPT_APW_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.1.2 NDcPP22e: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 6.1 (FPT_APW_EXT.1) in the ST states that the TOE is designed to not disclose or store plaintext passwords (e.g., passwords are never recorded in the audit records or displayed during the authentication process). The passwords are stored hashed using SHA-512 with a 32-bit salt value (FTD and FMC). Only 'root' user account with access to the shell can view the hashed passwords and this is prohibited in the evaluated configuration.

The ST further states that for FXOS, passwords are stored in hashed form using SHA-512. Only admin users can load a debugging plugin (which is NOT given to customers) to have a file system based access to key files.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.2 Failure with Preservation of Secure State (Self-Test Failures) (VPNGW13:FPT_FLS.1/SelfTest)

2.9.2.1 VPNGW13:FPT_FLS.1.1/SelfTest

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.1 (FPT_FLS.1/SelfTest) in the ST indicates that on the TOE (FTD), noise source health tests are run both periodically and at start-up to determine the functional health of the noise source. These tests are specifically designed to catch catastrophic losses in the overall entropy associated with the noise source. Tests are run on the raw noise output, before the application of any conditioners. If a noise source fails the health test either at start-up or after the device is operational, the platform will be shut down.

Whenever a failure (e.g., POST or integrity test fails) occurs within the TOE (FTD) that results in the FTD ceasing operation, the FTD securely disables its interfaces to prevent the unintentional flow of any information to or from the FTD and reloads. So long as the failures persist, the FTD will continue to reload. This functionally prevents any failure from causing an unauthorized information flow. There are no failures that circumvent this protection.

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

Section “Self-Tests” of the FTD Admin Guide outlines possible self-test errors that could take place and what steps to follow to attempt to fix them. The guidance documentation specifies that Cisco products perform a suite of FIPS 140-2 self-tests during power-up and re-boot. If any of the self-test fails, the product will not enter operational state and an error message indicating a self-test failure will be displayed via the serial console CLI. If this occurs, the appliance should be rebooted. If the product still does not enter operational state, Cisco Support should be contacted. The following possible errors that can occur during the self-tests are:

• Known Answer Test (KAT) failures

• Zeroization Test failure

• Software integrity failure

Component Testing Assurance Activities: None Defined

2.9.3 Basic internal TSF data transfer protection - per TD0639 (NDcPP22e:FPT_ITT.1)

2.9.3.1 NDcPP22e:FPT_ITT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the TOE is not a distributed TOE, then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall examine the TSS to determine that, for all communications between components of a distributed TOE, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator shall also confirm that all protocols listed in the TSS for these inter-component communications are specified and included in the requirements in the ST.

Section 6.1 (FPT_ITT.1, FPT_ITT.1/Join) in the ST states that the communication between the FMC and FTD is protected by TLSv1.2. TLS provides authentication, key exchange, encryption and integrity protection of all data transmitted between the TOE components. As noted throughout this AAR, the evaluator has confirmed that all protocols listed in the TSS are specified and included in the requirements in the ST.

The evaluator also notes the inclusion of the iteration “FPT_ITT.1/Join” in the ST. This is consistent with Application Note 48 in the NDcPP which states that “If the ST author selects the FTP_ITC.1/FPT_ITT.1 channel type in the main selection in FCO_CPC_EXT.1.2 then the ST identifies the registration channel as a separate iteration of FTP_ITC.1 or FPT_ITT.1 and gives the iteration identifier (e.g. “FPT_ITT.1/Join”) …”

Component Guidance Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall confirm that the guidance documentation contains instructions for establishing the relevant allowed communication channels and protocols between each pair of authorized TOE components, and that it contains recovery instructions should a connection be unintentionally broken.

See FCO_CPC_EXT.1 which describes the instructions for establishing communications between the FTD and FMC.

Section “Device Registration” in the FTD Admin Guide states that if the connection between FMC and FTD is broken during device registration, the FMC and FTD will continue to attempt to reconnect and retry registration for up to two minutes. If the registration has not completed within two minutes, restore connectivity between the FMC and FTD and reinitiate the registration from FMC. If connectivity is lost between FMC and FTD after device registration each endpoint will automatically attempt to re-initiate connection to the other until connectivity is restored, no administrative action is required other than resolving any connectivity issues in the networks between the FMC and FTD.

Component Testing Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall ensure that communications using each protocol between each pair of authorized TOE components 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 with an authorized IT entity, the channel data is not sent in plaintext.

c) Test3: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route between distributed components.

The evaluator shall ensure that, for each different pair of non-equivalent component types, the connection is physically interrupted for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration that is shorter than the application layer timeout but is of sufficient length to interrupt the network link layer.

The evaluator shall ensure that when physical connectivity is restored, either communications are appropriately protected, or the secure channel is terminated and the registration process (as described in the FTP_TRP.1/Join) re-initiated, with the TOE generating adequate warnings to alert the Security Administrator.

In the case that 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 components. 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.

Test 1: This test was performed as part of Test 3 below where testing demonstrated that the TLS handshake completed successfully and application data is seen between the FMC and the managed device.

Test 2: This test was performed as part of Test 3 below where testing demonstrated that the channel data is TLS encrypted and not sent in plaintext.

Test 3: The evaluator first noted that the session establishment between the FMC and FTD was successful because the TLS handshake completed and application data was passed between the two devices. The application data is encrypted, so no channel data was sent in plaintext. The exact timeout durations are documented in the Test Report.

In the MAC layer timeout test, the evaluator disconnected the connection. The TOE components attempt to pass data on the channel, but the packets are lost. After a short period of time (enough to cause a network link layer disruption), the evaluator reconnected the cable between the two devices and observed the continuation of application data between the devices. The evaluator successfully verified that the communications were adequately protected after coming back online.

In the App layer timeout test, the evaluator disconnected the connection. After enough time to cause the application layer disconnect, the evaluator reconnected the cable and observed that a new TLS handshake was initiated. The channel data was protected after the disconnect.

Both disconnects were performed in a physical manner, where a core switch was disconnected to interrupt the connection. At no time during the test did the evaluator observe that channel data was not adequately protected via TLS.

2.9.4 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys) (NDcPP22e:FPT_SKP_EXT.1)

2.9.4.1 NDcPP22e: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 an interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Section 6.1 (FPT_SKP_EXT.1) in the ST indicates that for FTD and FMC, the TOE is designed to not disclose or store cryptographic keys such as encryption symmetric keys and private keys. The public keys can be viewed but cannot be modified without detection. Note that access to public keys is restricted to administrators.

The ST further states for FXOS that all keys are stored on volatile memory without encryption. Only admin users can load a debugging plugin (which is NOT given to customers) to have a file system based access to key files.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.5 Reliable Time Stamps - per TD0632 (NDcPP22e:FPT_STM_EXT.1)

2.9.5.1 NDcPP22e:FPT_STM_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.5.2 NDcPP22e: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) in the ST states that the FMC and FXOS provides a source of date and time information for the TOE, used in audit timestamps, in validating service requests, determining the validity of certificates, and for tracking time-based actions related to session management including timeouts for inactive administrative or remote VPN sessions (FTA_SSL*), and renegotiating SAs for IPsec tunnels (FCS_IPSEC_EXT.1(1)). This function can only be accessed from within the configuration exec mode via the privileged mode of operation or using the appropriate role. The clock function is reliant on the system clock provided by the underlying hardware. The clock's date and time can be adjusted by authorized administrators. FMC and FMCv's clock can be configured manually by the administrators and can also synchronize time with a NTP server. FXOS can either set its time manually or sync with an NTP server. The FTD automatically synchronizes its clock with the FXOS clock.

Component Guidance Assurance Activities: The evaluator examines the guidance documentation to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the guidance documentation instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

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 NTP (FMC/FMCv) in the FTD Admin Guide provides instructions on how to configure NTP."

Section *Setting the Time Manually via …* in the FTD Admin Guide provides instructions on how the admin can set the system's (FMC) time. There are two versions of this section, one for CLI and one for GUI that both contain instructions to configure the time via the different administrative user interfaces.

Section "Configure the Clock" in the FTD Admin Guide indicates that the FTD must be configured to synchronize its clock with the FMC and provides the steps for doing so.

Manually via CLI and *Set the Date and Time Manually via GUI* in the FXOS Admin Guide describe how to configure the clock and time zone manually.

The *Section Setting the Date and Time Using NTP* in the FXOS Admin Guide describes the commands, windows and actions necessary to configure FXOS to use NTP to set its date and time.

The TOE does not obtain its time from an 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 uses 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 will observe that the NTP server has set the time to what is expected. If the TOE supports multiple protocols for establishing a connection with the NTP server, the evaluator shall perform this test using each supported protocol claimed in the guidance documentation.

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.

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.

Test 1: The evaluator set the local clock from the FMC web UI and observed the time change and the synchronization of the time change on the FTD. The corresponding audits were collated in evidence for FAU_GEN.1. This process was repeated for FXOS where the same result was observed.

Test 2: The evaluator followed guidance to configure an external NTP server on the TOE using NTPv4 and NTP authentication with SHA-1. This process was repeated for FXOS which, in contrast, uses NTPv3 and does not claim support for NTP authentication. This was performed in conjunction with FCS_NTP_EXT.1 testing.

Test 3: Not applicable. The TOE does not obtain time from an underlying VS.

2.9.6 TSF testing (NDcPP22e:FPT_TST_EXT.1)

2.9.6.1 NDcPP22e:FPT_TST_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 ensure that it details the self-tests that are run by the TSF; this description should include an outline of what the tests are actually doing (e.g., rather than saying 'memory is tested', a description similar to 'memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written' shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

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.

Section 6.1 (FPT_TST_EXT.1) in the ST states that the FTD, FMC and FXOS run a suite of self-tests during initial start-up (power-on-self-tests or POST) to verify its correct operation. When CC mode is enabled on the FMC, FTD and FXOS, additional cryptographic tests and software integrity test will be run during start-up. The self-testing includes cryptographic algorithm tests (known-answer tests) that feed pre-defined data to cryptographic modules and confirm the resulting output from the modules match expected values, and firmware integrity tests that verify the digital signature of the code image using RSA-2048 with SHA-512. The cryptographic algorithm testing verifies proper operation of encryption functions, decryption functions, signature padding functions, signature hashing functions, and random number generation. The firmware integrity testing verifies the FTD, FMC and FXOS images have not been tampered with or corrupted. If any of these self-tests fails, the TOE will cease operation.

Noise source health tests are run both periodically and at start-up on the FTD to determine the functional health of the noise source. These tests are specifically designed to catch catastrophic losses in the overall entropy associated with the noise source. Tests are run on the raw noise output, before the application of any conditioners. If a noise source fails the health test either at start-up or after the device is operational, the platform will be shut down.

Whenever a failure (e.g., POST or integrity test fails) occurs within the FTD that results in the FTD ceasing operation, the FTD securely disables its interfaces to prevent the unintentional flow of any information to or from the FTD and reloads. So long as the failures persist, the FTD will continue to reload. This functionally prevents any failure from causing an unauthorized information flow. There are no failures that circumvent this protection.

The sufficiency argument from the TSS states that the tests are sufficient to ensure the correct operation of the security features because they address integrity of the TOE executing firmware and verify the correctness of the cryptographic operations (including noise source) underlying the security features described by the Security Target.

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 "Self-Tests" of the FTD Admin Guide outlines possible self-test errors that could take place and what steps to follow to attempt to fix them. The guidance documentation specifies that Cisco products perform a suite of FIPS 140-2 self-tests during power-up and re-boot. If any of the self-test fails, the product will not enter operational state, and an error message indicating a self-test failure will be displayed via the serial console CLI. If this occurs, the appliance should be rebooted. If the product still does not enter operational state, Cisco Support should be contacted. The following possible errors that can occur during the self-tests are:

• Known Answer Test (KAT) failures

• Zeroization Test failure

• Software integrity failure

Section *Self-Tests* in the FXOS Admin Guide indicates that FXOS also performs FIPS 140-2 self-tests during power-up and reboot, and that if any of these tests fail, the product will not enter operational state. The following possible errors that can occur during this self-test are:

o Known Answer Test (KAT) failures

o Zeroization Test failure

o Software integrity failure

This section further states that if any of the self-tests fail, the appliance should be rebooted. If the product still does not enter the operational state, Cisco Support should be contacted.

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, chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software. Note that the testing is not restricted to the cryptographic functions of the TOE.

b) FIPS 140-2, chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

The evaluator shall either verify that the self tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

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.

The evaluator initiated a reboot from each device and confirmed that the status messages show that the FIPS self-tests were executed successfully.

2.9.7 TSF Testing - per TD0824 (VPNGW13:FPT_TST_EXT.1)

2.9.7.1 VPNGW13:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.7.2 VPNGW13:FPT_TST_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module requires a particular self-test to be performed, but this self-test is still evaluated using the same methods specified in the Supporting Document.

See NDcPP22e:FPT_TST_EXT.1 and FPT_FLS.1.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.8 Self-Test with Defined Methods (VPNGW13:FPT_TST_EXT.3)

2.9.8.1 VPNGW13:FPT_TST_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.8.2 VPNGW13:FPT_TST_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

See NDcPP22e:FPT_TST_EXT.1 and FPT_FLS.1.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.9 Trusted update (NDcPP22e:FPT_TUD_EXT.1)

2.9.9.1 NDcPP22e:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.9.2 NDcPP22e:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.9.3 NDcPP22e: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 needs to 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. Alternatively an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

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.

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

Section 6.1 (FPT_TUD_EXT.1) in the ST states that the TOE components (FMC, FTD and FXOS) have specific versions that can be queried by an administrator. When updates are made available by Cisco, an administrator can obtain and manually install those updates.

Digital signatures (RSA), with key sizes of 2048 and 3072 bits, are used to verify software/firmware update files (to ensure they have not been modified from the originals distributed by Cisco) before they are used to update the applicable TOE components. The update process will fail if the digital signature verification process fails. Updates can be downloaded from http://www.cisco.com/go/firepower9300-software or http://www.cisco.com/go/firepower4100-software or https://software.cisco.com with a Cisco.com account. The appropriate software image is then downloaded to the administrator's workstation, then uploaded to FMC, or FXOS (FTD updates are uploaded to FMC then pushed from FMC to FTD). Software update files are verified using digital signatures (RSA) automatically at the time they are uploaded to FMC or FXOS. Update files will fail to be stored on the device if they fail validation. Images stored on FXOS can be re-verified with the command – "verify platform-pack version *version_number*".

On FMC, the FMC and FTD updates can uploaded and installed by navigating to System > Updates. Several upload files can remain stored locally on FMC and installed to FMC or FTD at a later time. When updates are initiated they are applied immediately, and the FMC or FTD will reload automatically with the new software version. That same page also shows the currently running version on FMC. To view the currently running version of any FTD, navigate to Devices > Device Management > then select the device > click on the 'Device' tab.

On FXOS, FXOS updates can be uploaded by navigating to System > Updates. Several upload files can remain stored locally on FXOS and installed at a later time. When updates are initiated they are applied immediately, and the FXOS will reload automatically with the new software version. To view the currently running version of FXOS, click on the 'Overview' tab.

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.

If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

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

Section “Product Upgrade” of the FTD Admin Guide states that an administrator can view the version and date of each update by viewing the (System > Updates) page of the WebUI. It also indicates that viewing the (Help > About) tab will show the currently installed version of software. Also, the (Devices > Device Management) tab will allow the admin to view the version numbers of all managed devices.

Section “Product Upgrade” in the FTD Admin Guide provides instructions for manually updating the TOE components (FMC, FTD). It states that the FMC must be updated before the managed devices (i.e. FTD). It also provides warnings regarding capabilities that might be affected when an administrator installs an update from a managed device. The following capabilities may be affected: Traffic inspection and connection logging, Traffic flow including switching, routing, and related functionality and Link state.

As FMC and FTD updates are uploaded to FMC the FMC automatically verifies their integrity using RSA digital signature verification. If any file fails the signature verification an “Upload failed” message will be displayed at the top-center of the page, the file will not be stored on FMC, and the file will not be listed on the Product Updates page so it cannot be installed. If the error message indicates a lack of storage space, remove unneeded update files and repeat the upload. If any other reason for failure is indicated in the Upload Failed error message, redownload the update file from software.cisco.com, and re-attempt the upload. If the upload (including image integrity verification) is successful, the uploaded file will be listed on the Product Updates page. If uploads continue to fail, contact Cisco TAC for assistance. When stored update files are installed their integrity is verified again using RSA digital signature verification; the FMC will re-verify integrity of FMC updates, and the FTD will verify integrity of FTD updates.

FXOS can also verify the image via the Firepower Chassis Manager WebUI by navigating to System, then Updates, then clicking “Check Image Integrity” button next to the ASA image.

The digital signature uses 2048-bit RSA with SHA-512. If the image verification fails, the system will not boot into operational mode. If the update succeeds, the system will boot to the new image. The *Update Application Image via CLI* and *Delete a ASA Logical Device via CLI* section in the FXOS Admin Guide describes how the “set startup-version” command is used to set the boot image version and how the “show app-instance” command is used to view the boot image version.

The *Image Management* section in the FXOS Admin Guide contains several sections which describe the process of obtaining updates from Cisco, loading them into the FXOS, verifying the image and updating the software using the image. The *Verifying the Integrity of an Image* section in the FXOS Admin Guide describes how to use the “show package” command to list the images, and how to use the “verify platform-pack version” command to optionally re-verify the images. The *Update the Platform Bundle Image via CLI* section in the FXOS Admin Guide describes how to use the “install platform platform-vers” command to set the boot image.

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:

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

2) An image that has not been signed

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

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

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

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

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

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

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

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

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

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

Test 1: The FTD is updated via the FMC. First, the evaluator checked the currently installed version of the FTD (which is also the most recently installed version), then updated the FTD using the FMC. The evaluator then checked the new version number and verified that the installation was successful.

For the FMC, first, the evaluator checked the current version of the FMC, as well as the reported most recently installed version. The evaluator then uploaded the update. The success is marked by the message displayed to the screen in the WebUI. The evaluator then clicked on the install/upgrade button in order to upgrade the FMC version. The version changed to the updated version after the installation succeeded.

In regard to delayed activation for, each TOE component does not have a separate activation step, as the TSS states. Therefore, in each case the most recently installed version was updated along with the currently running version.

The process for test 1 was repeated for FXOS. The evaluator checked and noted the currently running version of the TOE as well as the most recently installed version. The evaluator then initiated the update. Once the update completed successfully, the evaluator noted that both the currently running version and most recently installed versions matched and were updated to the version identified by the update image. Similar to other TOE devices, FXOS does not have a separate step to 'activate' the image. As a result, the most recently installed version and currently running version will be updated at the same time.

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

1. Bytes in the update file are modified via a hex editor

2. Update is missing a signature

3. Update's signature is corrupted

All TOE components, including FTD devices (with the exception of FXOS, which handles its own updates) are updated via the FMC components. The update image is verified at the time of upload to the FMC component. Attempts to update with each of these modified update files outlined above failed.

For the case of delayed activation, the evaluator found that the FMC did not update the "most recently installed version" when rejecting these invalid updates. This indicates that the TOE rejects invalid updates before updating the most recently installed version.

Test 2 was repeated for FXOS. The image is also verified at the time of upload, and in each case outlined above the TOE's image validation successfully detected the invalid image. As expected, the FXOS also did not update any of its version indicators as a result of attempting to upload the invalid images.

Test 3: Not applicable. Published hash is not used to verify the integrity of the TOE updates.

2.9.10 Trusted Update - per TD0824 (VPNGW13:FPT_TUD_EXT.1)

2.9.10.1 VPNGW13:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.10.2 VPNGW13:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.10.3 VPNGW13:FPT_TUD_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to mandate that a particular selection be chosen, but this selection is part of the original definition of the SFR so no new behavior is defined by the PP-Module.

There are no evaluation activities specified. See the NDcPP22e:FPT_TUD_EXT.1.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.10 TOE ACCESS (FTA)

2.10.1 TSF-INITIATED TERMINATION (NDCPP22E:FTA_SSL.3)

2.10.1.1 NDCPP22E: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 6.1 (FTA_SSL.3) in the ST states that an administrator can configure maximum inactivity times for both local and remote administrative sessions. When a session is inactive (i.e., no session input) for the configured period of time the TOE will terminate the session, requiring the administrator to log in again to establish a new session when needed. The inactivity times are set at a default of 60 minutes, but an Administrator can configure the inactivity time for the FMC and FTD through the FMC WebUI and FXOS through the FXOS CLI.

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 "Inactivity Timeout Setting" in the FTD Admin Guide states by default, all user sessions (web-based and CLI) automatically log out after 60 minutes (1 hour) of inactivity. Users with the Administrator Role can change the inactivity timeout value in the system policy to meet their security needs. It then provides the steps that can be taken to set this timeout.

Section "Configure Inactivity Timeout Settings" in the FTD Admin Guide provides instructions for configuring the console timeout for FTD which applies to all CLI access including serial console and SSH.

Section *Selecting the Default Authentication Service via CLI* of the FXOS Admin Guide details timeout configuration for FXOS.

Firepower-chassis /security/default-auth # **set session-timeout** *seconds*

Firepower-chassis /security/default-auth # **set con-session-timeout** *seconds*

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

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

For each method of remote administration on the FXOS, FTD and FMC, the evaluator configured two different idle timeout values. The evaluator then logged into the device, then leaving the session inactive. The evaluator observed that after the configured amount of time, the TOE terminated the evaluator's session.

2.10.2 TSF-Initiated Termination (VPN Headend) (VPNGW13:FTA_SSL.3/VPN)

2.10.2.1 VPNGW13:FTA_SSL.3.1/VPN

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 verify that it describes the ability of the TSF to terminate an inactive VPN client session.

Section 6.1 (FTA_SSL.3/VPN) in the ST states that when a remote VPN client session reaches a period of inactivity, its connection is terminated, and it must re-establish the connection with new authentication to resume operation. This period of inactivity is set by the administrator using Objects > Object Management > Group Policy > Session Settings > Idle Timeout in the VPN configuration. The Group Policy is then tied to a Connection Profile.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to verify that it provides instructions to the administrator on how to configure the time limit for termination of an active VPN client session.

Section "Configure Group Policy Object" in the Supplement provides instructions for setting the 'Idle Timeout/Alert Interval' in the VPN configuration which specifies the VPN client user's idle timeout period in minutes.

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

Test 1: The evaluator shall follow the steps provided in the operational guidance to set the inactivity timer for five minutes. The evaluator shall then connect a VPN client to the TOE, let it sit idle for four minutes and fifty seconds, and observe that the VPN client is still connected at this time by performing an action that would require VPN access. The evaluator shall then disconnect the client, reconnect it, wait five minutes and ten seconds, attempt the same action, and observe that it does not succeed. The evaluator shall then verify using audit log data that the VPN client session lasted for exactly five minutes.

Test 2: The evaluator shall configure the inactivity timer to ten minutes and repeat Test 1, adjusting the waiting periods and expected audit log data accordingly.

Test 1: The evaluator followed the appropriate steps outlined in the test-case above and found that the second action attempt (after 5 minutes 10 seconds) failed as expected.

Test 2: Results for this test mirrored test 1 for this requirement, adjusting the time period accordingly.

2.10.3 User-initiated Termination (NDcPP22e:FTA_SSL.4)

2.10.3.1 NDcPP22e: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 local and remote administrative sessions are terminated.

Section 6.1 (FTA_SSL.4) in the ST states that an administrator is able to exit out of both local and remote administrative sessions of the FMC, FTD and FXOS, effectively terminating the session so it cannot be re-used and will require authentication to establish a new session.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Section “Logout” in the FTD Admin Guide provides instructions for the administrator to log out and/or end their session with the TOE for the FMC GUI and FMC CLI.

Section “FTD Logout” in the FTD Admin Guide provides instructions for logging out of the FTD CLI.

Section *Logout* in the FXOS Admin Guide explains how an administrator can terminate their interactive session with FXOS.

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

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

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

Test 1: This test was performed as part of FIA_UIA_EXT.1-t1 where the evaluator performed a logout for each interactive console session after a successful authentication attempt.

Test 2: This test was performed as part of FIA_UIA_EXT.1-t1 where the evaluator performed a logout for each remote interactive session after a successful authentication attempt.

2.10.4 TSF-initiated Session Locking (NDcPP22e:FTA_SSL_EXT.1)

2.10.4.1 NDcPP22e: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 6.1 (FTA_SSL_EXT.1) in the ST states that an administrator can configure maximum inactivity times for both local and remote administrative sessions. When a session is inactive (i.e., no session input) for the configured period of time the TOE will terminate the session, requiring the administrator to log in again to establish a new session when needed. The inactivity times are set at a default of 60 minutes, but an Administrator can configure the inactivity time for the FMC and FTD through the FMC WebUI and FXOS through the FXOS CLI.

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 "Inactivity Timeout Setting" in the FTD Admin Guide states by default, all user sessions (web-based and CLI) automatically log out after 60 minutes (1 hour) of inactivity. Users with the Administrator Role can change the inactivity timeout value in the system policy to meet their security needs. It then provides the steps that can be taken to set this timeout.

Section "Configure Inactivity Timeout Settings" in the FTD Admin Guide provides instructions for configuring the console timeout for FTD which applies to all CLI access including serial console and SSH.

Section *Selecting the Default Authentication Service via CLI* of the FXOS Admin Guide details timeout configuration for FXOS.

Firepower-chassis /security/default-auth # **set session-timeout** *seconds*

Firepower-chassis /security/default-auth # **set con-session-timeout** *seconds*

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.

Test 1: The evaluator followed operational guidance to set the console timeout on the FTD to both 5 and 7 minutes and observed that the user was logged out automatically in 5 and 7 minutes, respectively. The evaluator repeated this test on the FMC with the same time values, and observed the same behavior. For the FXOS, this value was 2 minutes and 5 minutes and the same behavior was observed.

2.10.5 Default TOE Access Banners (NDcPP22e:FTA_TAB.1)

2.10.5.1 NDcPP22e: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 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 6.1 (FTA_TAB.1) in the ST states that The TOE provides administrators with the capability to configure advisory banner or warning message(s) that will be displayed prior to completion of the logon process at the local console or via any remote connection (e.g., SSH or HTTPS). The TOE displays an advisory notice and a consent warning message for each administrative method of access:

- FMC/FMCv: Console, SSH, and WebUI
- FXOS: Console, SSH, and WebUI
- FTD: The FTD CLI (SSH), which provides the login banner.

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

Section “Configure Login Banner” in the FTD Admin Guide provides instructions for administrators to configure the Login Banner on the FMC or FTD (managed device).

Section *Creating the Pre-Login Banner* in the FXOS Admin Guide details banner configuration for FXOS.

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

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

Test 1: The evaluator followed the AGD to configure the login banner as part of testing for FTA_TAB.1. The evaluator demonstrated that the configured banner was presented during authentication at each administrative interface in FIA_UIA_EXT.1 Test 1.

2.10.6 TOE Session Establishment (VPNGW13:FTA_TSE.1)

2.10.6.1 VPNGW13:FTA_TSE.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 verify that it describes the methods by which the TSF can deny the establishment of an otherwise valid remote VPN client session (e.g., client credential is valid, not expired, not revoked, etc.), including day, time, and IP address at a minimum.

Section 6.1 (FTA_TSE.1) in the ST states that the TOE allows for creation of ACLs that restrict VPN connectivity-based client's IP address (location). These ACLs allow customization of all of these properties to allow or deny access (Objects > Object Management > Group Policy > Traffic Filter Fields > Access List Filter). In addition, the administrator can create Group Policy tied to Connection Profile (Objects > Object Management > Group Policy > Session Settings > Access Hours) which can be used to restrict access based on date and time.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance to determine that it provides instructions for how to enable an access restriction that will deny VPN client session establishment for each attribute described in the TSS.

Section "Configure Group Policy Object" in the Supplement provides instructions for setting the 'Idle Timeout/Alert Interval' in the VPN configuration which specifies the VPN client user's idle timeout period in minutes and the 'Access Hours' in the VPN configuration which specifies the range of time the group policy is available to be applied to a remote access user.

Section "Restrict VPN Client Connections by Source IP" in the Supplement provides instructions to restrict which source IP addresses are permitted to attempt to negotiate IKE connections with the FTD.

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

Test 1: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it, noting the IP address from which the client connected. The evaluator shall follow the steps described in the operational guidance to prohibit that IP address from connecting, attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 2: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it. The evaluator shall follow the steps described in the operational guidance to prohibit the VPN client from connecting on a certain day (whether this is a day of the week or specific calendar date), attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 3: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it. The evaluator shall follow the steps described in the operational guidance to prohibit the VPN client during a range of times that includes the time period during which the test occurs, attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 4: [conditional] If any other attributes are identified in FTA_TSE.1, the evaluator shall conduct a test similar to tests 1 through 3 to demonstrate the enforcement of each of these attributes. The evaluator shall demonstrate a successful remote client VPN connection, configure the TSF to deny that connection based on the attribute, and demonstrate that a subsequent connection attempt is unsuccessful.

Test 1: A successful connection was tested in VPNGW13:FTA_SSL.3/VPN. The evaluator configured the TOE to restrict VPN access based on IP address and confirmed that the TOE denied the VPN access based on IP address.

Test 2: A successful connection was tested in VPNGW13:FTA_SSL.3/VPN. The evaluator configured the TOE to restrict VPN access for a whole day and confirmed that the TOE successfully restricted VPN access during the time period specified.

Test 3: The evaluator configured VPN client restriction based on a time range. The evaluator then attempted a connection outside of the configured times. The TOE successfully blocked the connection attempt.

Test 4: Not applicable. There are no other attributes claimed in FTA_TSE.1.

2.10.7 VPN Client Management (VPNGW13:FTA_VCM_EXT.1)

2.10.7.1 VPNGW13:FTA_VCM_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 the TSS to verify that it asserts the ability of the TSF to assign a private IP address to a connected VPN client.

Section 6.1 (FTA_VCM_EXT.1) in the ST states that the TOE provides the option to assign the remotely connecting VPN client an internal network IP address. The Objects > Object Management > Address Pools can be used to define the range of IP and IPv6 addresses to be available for use.

Component Guidance Assurance Activities: There are no guidance EAs for this component.

There are no guidance EAs for this component.

Component Testing Assurance Activities: The evaluator shall connect a remote VPN client to the TOE and record its IP address as well as the internal IP address of the TOE. The evaluator shall verify that the two IP addresses belong to the same network. The evaluator shall disconnect the remote VPN client and verify that the IP address of its underlying platform is no longer part of the private network identified in the previous step.

The evaluator first recorded the internal IP address of the TOE. After connecting the VPN client to the gateway (TOE), the evaluator checked the client's IP address and ensured that the TOE successfully assigned a private IP address to the client. These addresses belonged to the same network. The evaluator then disconnected the VPN client and once again checked the client to ensure the IP address was no longer assigned. The evaluator found this to be the case upon disconnecting.

2.11 Trusted path/channels (FTP)

2.11.1 Inter-TSF trusted channel - per TD0639 (NDcPP22e:FTP_ITC.1)

2.11.1.1 NDcPP22e:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.1.2 NDcPP22e:FTP_ITC.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.1.3 NDcPP22e: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) in the ST states that the TOE uses IPsec and/or TLS to protect communications between itself and remote entities for the following purposes:

- The TOE protects transmission of audit records when sending syslog message to a remote audit server by transmitting the messages:
  - From FMC/FMCv as a TLS client, using X.509v3 certificates for assured identification of the syslog server and with mutual authentication supported.
  - From FXOS over IPsec, using X.509v3 certificates for assured identification of the syslog server.

  - From FTD as a TLS client (FTD TLS Client), that is configured by the FMC and is the main audit system for audits generated by FTD. It sends audit events such as IPsec and login messages to the external syslog server and Mutual authentication is not supported.
  - From FTD as a TLS client (FTD OS TLS Client), that is configured through the FTD's command line and sends audit events to an external syslog server such as SSH login, console login, etc. and Mutual authentication is not supported.
  - From FTD to an external syslog server over IPsec.

- The TOE protects communication with a NTP server:
  - From FXOS to a NTP server over IPsec, using RSA for peer authentication that X509v3 certificates.

- The TOE (FTD only) protects peer-to-peer VPN connections between itself and VPN peers (connections can be initiated by the TOE or by the peer) using IPsec, using X.509v3 certificates for assured identification of the peer.
- The TOE (FTD Only) protects VPN connections inbound from VPN clients using IPsec, using X.509v3 certificates for assured identification of the VPN client. Note that the remote VPN client is in the operational environment.
- Connections to authentication servers (AAA servers) can be protected via IPsec tunnels. Connections with AAA servers can be configured for authentication of TOE administrators.
  - RADIUS over IPsec (FXOS)
  - TACACS+ over IPsec (FXOS)

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.

Refer to FCS_TLSC_EXT.1 and FCS_TLSC_EXT.2 which describe the instructions for establishing TLS between the FMC and an external syslog server, between the FTD and an external syslog server and between the TOE components.

Section "Configure the external audit server (i.e. syslog-ng-daemon)" in the FTD Admin Guide states that the administrator is responsible for maintaining the connection between the system and audit server. If the connection is unintentionally broken, the administrator should perform the following steps to diagnose and fix the problem:

• Check the physical network cables.

• Check that the audit server is still running.

• Reconfigure the audit log settings.

• If all else fail, reboot the system and audit server.

Section "FTD Remote Access VPN" in the Supplement indicates that if a Remote Access VPN session is being used to tunnel SSH for remote administrative access to FTD (SSH over IPsec) and the IPsec connection between from the VPN client (AnyConnect) and FTD is unintentionally broken the AnyConnect client will automatically attempt to reinitiate the IPsec connection. If AnyConnect is able to automatically reestablish the IPsec session the tunneled SSH session, if previously established, may have remained active though it may have timed out and thus would need to be reinitiated from the client. In most cases no administrative action is required (on the AnyConnect client nor on FTD) though any connectivity issues will need to be resolved in the networks between AnyConnect and FTD. If IPsec connectivity has been lost for an extended period of time AnyConnect will discontinue automatic attempts to reestablish the tunnel, in which case the local administrative user of AnyConnect must re-initiate the IPsec tunnel, and reinitiate the SSH session through IPsec to FTD.

Since FTD and ASA utilize the same physical networking, these instructions apply to both FXOS and FTD equally.

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.

Test 1: Refer to NDcPP22e:FCS_IPSEC_EXT.1 where the Syslog IPsec Channel was fully tested. Refer to NDcPP22e:FCS_TLSC_EXT.1 where the FMC, FTD, and FTD OS syslog TLS channel were fully tested.

Test 2: Refer to NDcPP22e:FTP_ITC.1-t4 where the TOE initiating the trusted channel has been demonstrated.

Test 3: Refer to NDcPP22e:FTP_ITC.1-t4 where the TOE using an encrypted trusted channel has been demonstrated.

Test 4: These tests were repeated for the FMC TLS and FTD TLS and IPsec connections with the test peer. The test was also performed with FXOS IPsec connections. The exact amount of time for each disconnect is described in the Test Report:

MAC Layer: The evaluator connected the TOE to the test peer/server and began to send some traffic between the devices. The connection was physically disrupted and after enough time to cause a network link layer disruption, the connection was restored. After the restoration, the evaluator observed the IPsec and TLS connections remained active and data remained protected.

APP Layer: The evaluator connected the TOE to the test peer/server and began to send some traffic between the devices. The connection was physically disrupted and enough time to cause the application layer disruption, the connection was restored. After the restoration, the evaluator observed that the IPsec and TLS connections had to be reestablished and data remained protected.

Both disconnects were performed in a physical manner, where a core switch was disconnected to interrupt the connection. At no time during the test did the evaluator observe that channel data was not adequately protected via TLS.

2.11.2 Inter-TSF Trusted Channel (VPN Communications) (VPNGW13:FTP_ITC.1/VPN)

2.11.2.1 VPNGW13:FTP_ITC.1.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.2.2 VPNGW13:FTP_ITC.1.2/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.2.3 VPNGW13:FTP_ITC.1.3/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

See NDCPP22e:FTP_ITC.1 where the evaluation activities have also been applied to IPsec VPN communications.

Component Guidance Assurance Activities: The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

See NDCPP22e:FTP_ITC.1 where the evaluation activities have also been applied to IPsec VPN communications.

Component Testing Assurance Activities: TheEAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications. Additional evaluation testing for IPsec is covered in FCS_IPSEC_EXT.1.

See NDcPP22e:FTP_ITC.1.3 and FCS_IPSEC_EXT.1 where the evaluation activities have also been applied to IPsec VPN communications.

2.11.3 Trusted Path - per TD0639 (NDcPP22e:FTP_TRP.1/Admin)

2.11.3.1 NDcPP22e:FTP_TRP.1.1/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.3.2 NDcPP22e:FTP_TRP.1.2/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.3.3 NDcPP22e: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 6.1 (FTP_TRP.1/Admin) in the ST states that the TOE uses SSHv2 or HTTPS to provide the trusted path (with protection from disclosure and modification) for all remote administration sessions. Optionally, the FXOS and FTD support tunneling the SSH and HTTPS connections in IPsec VPN tunnels (remote VPN client). Remote administration of FMC can be performed using SSH or TLS/HTTPS. Remote administration of FXOS can be performed using SSH or TLS/HTTPS. Remote administration through the CLI of FTD is via SSH.

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

See FIA_UIA_EXT.1 which documents the areas in the guides which provide instructions for establishing the remote administrative sessions for each supported method.

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.

Test 1 and Test 2: The different remote administration methods were tested throughout the course of the evaluation. The successful testing of these channels and the demonstration of their encryption can be found in FCS_SSHS_EXT.1 for SSH on all devices and FCS_TLSS_EXT.1 for HTTPS on the FMC/FMCv and FXOS devices.

2.12 Intrusion Prevention (IPS)

2.12.1 Anomaly-Based IPS Functionality (IPS10:IPS_ABD_EXT.1)

2.12.1.1 IPS10:IPS_ABD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.12.1.2 IPS10:IPS_ABD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.12.1.3 IPS10:IPS_ABD_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 describes the composition, construction, and application of baselines or anomaly-based attributes specified in IPS_ABD_EXT.1.1.

The evaluator shall verify that the TSS provides a description of how baselines are defined and implemented by the TOE, or a description of how anomaly-based rules are defined and configured by the administrator.

If 'frequency' is selected in IPS_ABD_EXT.1.1, the TSS shall include an explanation of how frequencies can be defined on the TOE.

If 'thresholds' is selected in IPS_ABD_EXT.1.1, the TSS shall include an explanation of how the thresholds can be defined on the TOE.

The evaluator shall verify that each baseline or anomaly-based rule can be associated with a reaction specified in IPS_ABD_EXT.1.3.

The evaluator shall verify that the TSS identifies all interface types capable of applying baseline or anomaly-based rules and explains how they are associated with distinct network interfaces. Where interfaces can be grouped into a common interface type (e.g., where the same internal logical path is used, perhaps where a common device driver is used) they can be treated collectively as a distinct network interface.

Section 6.1 (IPS_ABD_EXT.1) in the ST describes how in an intrusion detection/prevention deployment, the TOE (FTD) examines packets based on network analysis policies and intrusion policies. The detection of anomalies is performed through rules configured by the administrator. The operations associated with the anomaly-based IPS policies are to allow the traffic flow for any sensor interface in any mode and allow the traffic flow and block/drop the traffic flow in inline mode. Administrators can define strings to match URLs/URIs, and web page content for pattern-matching. Several preprocessors allow administrators to detect specific threats, such as IP/TCP/UDP/ICMP portscans, ICMP/TCP flooding, DoS attacks and other rate-based attacks (“frequency”). The administrator can configure thresholds that mimic normal expected frequency and configure the TOE to detect and drop events exceeding the configured thresholds. (The ST notes that the TOE’s definition of the term “threshold” matches the definition of the term “frequency” in the IPS PP module, thus ‘frequency’ rather than ‘threshold’ is selected in the IPS_ABD_EXT.1.1 requirement).

When the system identifies a possible intrusion, it generates an intrusion or preprocessor event. Managed Sensors transmit their events to the FMC, where the administrators can view the aggregated data and gain a greater understanding of the attacks against their network assets. In an inline deployment, managed Sensors can also drop or replace packets that are known to be harmful. The packet decoder, the preprocessors, and the intrusion rules engine can all cause the TOE to generate an event. The TOE can also be configured to use intrusion rules to detect various attacks such as Teardrop, Bonk, Ping of Death, etc. The administrators can use pre-defined anomaly-based rules or can create custom rules to detect these and many other attacks.

Section 6.1 (IPS_ABD_EXT.1) in the ST states that the administrator can configure the Sensor in either a passive or inline deployment. In a passive (promiscuous) IPS deployment, the Sensor monitors traffic flowing across a network using a switch SPAN or mirror port. The SPAN or mirror port allows for traffic to be copied from other ports on the switch. This provides the system visibility within the network without being in the flow of network traffic. When configured in a passive deployment, the system cannot take certain actions such as blocking or shaping traffic. The administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as passive interfaces and deploy the intrusion policy to that interface via security zone (i.e., the interface is added to the zone). In an inline IPS deployment, the administrator configures the Sensor transparently on a network segment by binding two ports together. The administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as inline interfaces then assign a pair of inline interfaces to an inline set. The intrusion policy is then deployed to that inline set via security zone. The management interface (typically eth0) is separate from the other data monitoring interfaces (used as passive or inline) on the Sensor. It is used to set up and register the Sensor to the FMC.

Section 7.1 in the ST indicates that the TOE supports alert rules, pass rules and drop rules. The drop rule is available when operating in “inline mode”.

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions to manually create baselines or anomaly-based rules according to the selections made in IPS_ABD_EXT.1.1. Note that dynamic 'profiling' of a network to establish a baseline is outside the scope of the PP-Module.

The evaluator shall verify that the operational guidance provides instructions to associate reactions specified in IPS_ABD_EXT.1.3 with baselines or anomaly-based rules.

The evaluator shall verify that the operational guidance provides instructions to associate the different policies with distinct network interfaces.

Section “Rate-Based Attack Prevention” in the Supplement contains a hyperlink to documentation for Snort 3. This contains instructions for configuring the Rate Filter value, which controls the ‘frequency’ of packets as described in the TSS.

Section “Intrusion Rule Actions” provides instructions for enabling or disabling a rule within an individual intrusion policy as well as specifying the action the system takes if monitored conditions trigger the rule. The rule’s state can be set to Alert (which only generates events) or Block (which blocks the packet and generates an event).

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement describes how policies are associated with distinct network interfaces and provides instructions for configuring these associations. One FTD can have multiple interfaces, where two interfaces are configured as an inline pair, and a third interface is configured as passive. Traffic policies are defined in terms of network “zones”, which in turn are associated with FTD interfaces. So, a policy may be defined to allow traffic from “zone0” to “zone1”, though those zones may be mapped to the “outside” and “inside” interfaces on one FTD and also be mapped to “int1” and “int2” interfaces of another FTD which enforces the same policy. No FTD interface will forward traffic until policies have been configured an applied to that interface.

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

Test 1: The evaluator shall use the instructions in the operational guidance to configure baselines or anomaly-based rules for each attributes specified in IPS_ABD_EXT.1.1. The evaluator shall send traffic that does not match the baseline or matches the anomaly-based rule and verify the TOE applies the configured reaction. This shall be performed for each attribute in IPS_ABD_EXT.1.1.

Test 2: The evaluator shall repeat the test above to ensure that baselines or anomaly- based rules can be defined for each distinct network interface type supported by the TOE.

Test 1: The preprocessor detection rules for anomaly detected in headers and protocols can be seen where noted throughout IPS_SBD_EXT.1.1. For threshold (ie. frequency), the evaluator configured the TOE with an anomaly-based rule (and policy configuration) which utilized a combination of fields that were otherwise tested in IPS_SBD_EXT.1 (e.g. TCP destination port, IP destination address, and FTP commands). The evaluator then attempted to send traffic where the number of packets exceeds the configured maximum frequency for the anomalous packets and verified that the traffic exceeding the configured anomaly-based rule was detected and the TOE generated an intrusion event and dropped the traffic.

Test 2: The TOE supports only the Ethernet based networking that was tested during IPS_SBD_EXT.1.1-t1.

2.12.2 IP Blocking (IPS10:IPS_IPB_EXT.1)

2.12.2.1 IPS10:IPS_IPB_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.12.2.2 IPS10:IPS_IPB_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 verify how good/bad lists affect the way in which traffic is analyzed with respect to processing packets. The evaluator shall also very that the TSS provides details for the attributes that create a known good list, a known bad list, and their associated rules, including how to define the source or destination IP address (e.g. a single IP address or a range of IP addresses).

If the TSF uses address types other than a single IP or a range of IP addresses (e.g. MAC addresses), the evaluator shall check that the TSS explains what configurations would cause non-IP lists of known-good and known-bad addresses to take precedence over IP-based address lists.

The evaluator shall also verify that the TSS identifies all the roles and level of access for each of those roles that have been specified in the requirement.

Section 6.1 (IPS_IPB_EXT.1 [IPS]) in the ST indicates that prior to having network analysis or intrusion policies applied, traffic is first filtered by Security Intelligence (ie. Block List / Allow List). This filtering can be based on IP address, domain name, or URL.

Security Intelligence lists and feeds are collections of IP addresses, domain names, and URLs that you can use to quickly filter traffic that matches an entry on a list or feed.

• A list is a static collection that can be managed manually.

• A feed is a dynamic collection that updates on an interval.

Security Intelligence lists/feeds are grouped into:

• DNS (Domain names)

• Network (IP addresses)

• URLs

Predefined global Block lists and Allow lists for domains (DNS), IP addresses (Networks), and URLs are available by default and the Administrators can build on the list. Block List/Do Not Block List options are available on IP address, URL, and DNS requests. Using these rules to block or allow an item adds the item to the appropriate default Global list. By default, Access control and DNS policies use these Global lists. These lists can be applied on a per-policy basis.

The IPS_IPB_EXT.1.2 requirement indicates that IPS Administrators, Intrusion Admin and Access Admin can configure the IPS policy elements. Section 6.1 (FMT_SMF.1/IPS) in the ST identifies and describes the roles and level of access for each as follows:

• “IPS Administrator” (or Administrator): Have all privileges and access

• “IPS Analyst” (or Intrusion Admin): Have all access to intrusion policies, IPS policies and network analysis privileges but cannot deploy policies

• Access Admin: Have all access to Access Control policies but cannot deploy policies

• Discovery Admin: Have all access to network discovery, application detection, and correlation features but cannot deploy policies

• Security Analyst: Have all access to security event analysis feature

Component Guidance Assurance Activities: The evaluator shall verify that the administrative guidance provides instructions with how each role specified in the requirement can create, modify and delete the attributes of a known good and known bad lists.

If the TSF uses address types other than a single IP or a range of IP addresses (e.g. MAC addresses), the evaluator shall check that the operational guidance includes instructions for any configurations that would cause non-IP lists of known-good and known-bad addresses to take precedence over IP-based address lists.

Section “Configure Security Intelligence” in the Supplement describes how to configure do-not-block lists and black lists. It instructs the user to login with the Administrator, Intrusion Admin or Access Admin role. It is noted that only the Administrator, Intrusion Admin or Access Admin role can configure these attributes which are part of access control.

Section “User and Role Management” in the FTD Admin Guide describes the roles available to configure for a user account. They are as follows:

• “IPS Administrator” (or Administrator): Have all privileges and access.

• “IPS Analyst” (or Intrusion Admin): Have all access to intrusion policies, IPS policies and network analysis privileges but cannot deploy policies

• Access Admin: Have all access to Access Control policies but cannot deploy policies

• Discovery Admin: Have all access to network discovery, application detection, and correlation features but cannot deploy policies

• Security Analyst: Have all access to security event analysis feature

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

Test 1: The evaluator shall use the instructions in the operational guidance to create a known-bad address list. Using a single IP address, a list of addresses or a range of addresses from that list, the evaluator shall attempt to send traffic through the TOE that would otherwise be allowed by the TOE and observe the TOE automatically drops that traffic.

Test 2: The evaluator shall use the instructions in the operational guidance to create a known-good address list. Using a single IP address, a list of addresses or a range of addresses from that list, the evaluator shall attempt to send traffic that would otherwise be denied by the TOE and observe the TOE automatically allowing traffic.

Test 3: The evaluator shall add conflicting IP addresses to each list and ensure that the TOE handles conflicting traffic in a manner consistent with the precedence in IPS_NTA_EXT.1.1.

Test 1: While configured in in-line mode, the evaluator attempted the following types of traffics with the corresponding results:

Variation
CONTROL: Attempt to send traffic using addresses not in a Known-Good or a Known-Bad list: Pass -- packet allowed as expected.
Attempt to send traffic using a SRC address blocked by a Known-Bad list rule specifying only a single SRC address: Pass -- packet not allowed as expected.
Attempt to send traffic using a DST address blocked by a Known-Bad list rule specifying only a single DST address: Pass -- packet not allowed as expected.
Attempt to send traffic using a DST address blocked by a Known-Bad list rule specifying multiple DST addresses: Pass -- packet not allowed as expected.
Attempt to send traffic using a SRC address blocked by a Known-Bad list rule specifying multiple SRC addresses: Pass -- packet not allowed as expected.

Test 2: Because the TOE defaults to passing packets not otherwise blocked, the only way to “send traffic that would otherwise be denied by the TOE” is to create overlapping Good and Bad list rules. Since that is the focus of test 3 below, no additional tests were performed here. The tests which demonstrate that “Good List” entries are allowed to pass through the TOE are parts 6, 7, 8, 10, 12, 13, 14, 15, 17, and 19 of IPS_IPB_EXT.1-t3.

Test 3: For this product, the policy hierarchy order is not configurable. The following table organizes the results from the individual test variations, which comprise this test case. Details can be found in the log file and packet captures for each variation shown in the following table:

Variation
1. CONTROL: Attempt to send traffic using addresses not in a Known-Good or a Know-Bad list: Pass -- packet allowed as expected.
2. Attempt to send traffic using a SRC address that is in a Known-Good list rule and in a Known-Bad list rule both specifying a single SRC address: Pass -- packet allowed as expected.
3. Attempt to send traffic using a DST address that is in a Known-Good list rule and in a Known-Bad list rule both specifying a single DST address: Pass -- packet allowed as expected.
4. Attempt to send traffic using a SRC address in a Known-Good list rule and in a multiple address Known-Bad list (e.g., list or range): Pass -- packet allowed as expected.
5. Attempt to send traffic using a SRC address contained by a multiple address Known-Bad list (e.g., list or range) that is overlapped by a Known-Good list rule, where the SRC address is not part of the Known-Good list: Pass -- packet not allowed as expected.
6. Attempt to send traffic using a DST address in a Known-Good list rule and in a multiple address Known-Bad list (e.g., list or range): Pass -- packet allowed as expected.
7. Attempt to send traffic using a DST address contained by a multiple address Known-Bad list (e.g., list or range) that is overlapped by a Known-Good list rule, where the DST address is not part of the Known-Good list: Pass -- packet not allowed as expected.
8. Attempt to send traffic using a SRC address in a Known-Good list rule with multiple addresses and a single overlapping Known-Bad list address: Pass -- packet allowed as expected.
9. Attempt to send traffic using a SRC address matching a Known-Bad list rule with a single address overlapped by a Known-Good list rule with multiple addresses: Pass -- packet allowed as expected.
10. Attempt to send traffic using a DST address in a Known-Good list rule with multiple addresses and a single overlapping Known-Bad list address: Pass -- packet allowed as expected.
11. Attempt to send traffic using a DST address matching a Known-Bad list rule with a single address overlapped by a Known-Good list rule with multiple addresses: Pass -- packet allowed as expected.
12. Attempt to send traffic using an SRC address in a Known-Bad list rule, with multiple addresses when the rule is

overlapped by a Known-Good list rule with multiple addresses which does not include the SRC address: Pass -- packet not allowed as expected.
13. Attempt to send traffic using an SRC address in a Known-Bad list rule, with multiple addresses when the rule is overlapped by a Known-Good list rule with multiple addresses which does include the SRC address: Pass -- packet allowed as expected.
14. Attempt to send traffic using an DST address in a Known-Bad list rule, with multiple addresses when the rule is overlapped by a Known-Good list rule with multiple addresses which does not include the DST address: Pass -- packet not allowed as expected.
15. Attempt to send traffic using an DST address in a Known-Bad list rule, with multiple addresses when the rule is overlapped by a Known-Good list rule with multiple addresses which does include the DST address: Pass -- packet allowed as expected.

2.12.3 Network Traffic Analysis (IPS10:IPS_NTA_EXT.1)

2.12.3.1 IPS10:IPS_NTA_EXT.1.1

TSS Assurance Activities: The evaluator shall verify that the TSS explains the TOE's capability of analyzing IP traffic in terms of the TOE's policy hierarchy (precedence). The TSS should identify if the TOE's policy hierarchy order is configurable by the administrator for IPS policy elements (known-good lists, known-bad lists, signaturebased rules, and anomaly-based rules).

Regardless of whether the precedence is configurable, the evaluator shall verify that the TSS describes the default precedence as well as the IP analyzing functions supported by the TOE.

Section FAU_GEN.1/IPS in the ST describes the Access Control Policy which is associated with the intrusion policy where the intrusion, preprocessor, or decoder rule that generates an event is enabled. Access control rules invoke the intrusion policy such that all traffic permitted by the access control policy is then inspected by the designated intrusion policy.

Section 6.1 (IPS_NTA_EXT.1) in the ST indicates that network analysis policies (anomaly-based rules) govern the decoding and preprocessing of traffic after traffic is first filtered by Security Intelligence (ie. Whitelist/Blacklist) and before the traffic is inspected by access control rules and intrusion policies (signature-based rules). A network analysis policy governs packet processing in phases. First the system decodes packets through the first three TCP/IP layers, then continues with normalizing, preprocessing, and detecting protocol anomalies.

• The packet decoder converts packet headers and payloads into a format that can be easily used by the preprocessors and later, intrusion rules. Each layer of the TCP/IP stack is decoded in turn, beginning with the data link layer and continuing through the network and transport layers. The packet decoder also detects various anomalous behaviors in packet headers.

• The inline normalization preprocessor reformats (i.e., normalizes) traffic to minimize the chances of attackers evading detection. It prepares packets for examination by other preprocessors and intrusion rules, and helps ensure that the packets the system processes are the same as the packets received by the hosts on your network.

• Various network and transport layers preprocessors detect attacks that exploit IP fragmentation, perform checksum validation, and perform TCP and UDP session preprocessing.

• Various application-layer protocol decoders normalize specific types of packet data into formats that the intrusion rules engine can analyze. Normalizing application-layer protocol encodings allows the system to effectively apply the same content-related intrusion rules to packets whose data is represented differently, and to obtain meaningful results.

• The Modbus and DNP3 SCADA preprocessors detect traffic anomalies and provide data to intrusion rules. The baselines are provided by the preprocessors and detection of anomalies through rules configured by the administrator.

• Several preprocessors allow administrators to detect specific threats, such as IP/TCP/UDP/ICMP portscans, ICMP/TCP flooding, DoS attacks and other rate-based attacks (“frequency”). The administrator can configure threshold that mimics normal expected frequency and configure the TOE to detect and drop events exceeding the configured thresholds. (The ST notes that the TOE’s definition of the term “threshold” matches the definition of the term “frequency” in the IPS PP module, thus ‘frequency’ rather than ‘threshold’ is selected in the IPS_ABD_EXT.1.1 requirement).

Guidance Assurance Activities: The evaluator shall verify that the guidance describes the default precedence.

If the precedence is configurable, the evaluator shall verify that the guidance explains how to configure the precedence.

Section “Configure Security Intelligence” in the Supplement describes how to configure do-not-block list, block list rules which are applied to traffic before it is inspected by access control rules and intrusion policies. This section states that the policy hierarchy order is not configurable and follows this order: Security Intelligence (whitelist takes precedence over blacklist), anomaly-based rules, then signature-based rules.

Other sections in the Supplement, further describes the default precedence as follows:

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement states that traffic flow policy types include Prefilter, Access Control, and Intrusion policies. Prefilter policies are sub-policies of Access Control policies, and every Access Control policy has an associated Prefilter policy, which is used to define rules for encapsulated traffic.

Section “Configure Access Control Policy” in the Supplement provides instructions for creating an Access Control policy. The Intrusion and Network Analysis policies are associated with the Access Control policy which is then assigned to one or more sensors.

Section “Managing Intrusion Policies” in the Supplement states that Intrusion policies are invoked by your access control policy and are the system’s last line of defense before traffic is allowed to its destination.

Testing Assurance Activities: None Defined

2.12.3.2 IPS10:IPS_NTA_EXT.1.2

TSS Assurance Activities: The evaluator shall verify that the TSS indicates that the following protocols are supported:

- IPv4

- IPv6

- ICMPv4

- ICMPv6

- TCP

- UDP

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

Section 6.1 (IPS_NTA_EXT.1) in the ST describes how the network analysis policy governs packet processing. First the system decodes packets through the first three TCP/IP layers, then continues normalizing, preprocessing and detecting protocol anomalies.

• The packet decoder converts packet headers and payloads into a format that can be easily used by the preprocessors and later, intrusion rules. Each layer of the TCP/IP stack is decoded in turn, beginning with the data link layer and continuing through the network and transport layers. The packet decoder also detects various anomalous behaviors in packet headers.

• The inline normalization preprocessor reformats (i.e., normalizes) traffic to minimize the chances of attackers evading detection. It prepares packets for examination by other preprocessors and intrusion rules, and helps ensure that the packets the system processes are the same as the packets received by the hosts on your network.

• Various network and transport layers preprocessors detect attacks that exploit IP fragmentation, perform checksum validation, and perform TCP and UDP session preprocessing.

• Various application-layer protocol decoders normalize specific types of packet data into formats that the intrusion rules engine can analyze. Normalizing application-layer protocol encodings allows the system to effectively apply the same content-related intrusion rules to packets whose data is represented differently, and to obtain meaningful results. Conformance to protocols has been verified via compliance testing.

• The Modbus and DNP3 SCADA preprocessors detect traffic anomalies and provide data to intrusion rules. The baselines are provided by the preprocessors and detection of anomalies through rules configured by the administrator. The operations associated with the anomaly-based IPS policies are allow the traffic flow for any sensor interface in any mode and allow the traffic flow and block/drop the traffic flow in inline mode. Administrators

• Several preprocessors allow administrators to detect specific threats, such as IP/TCP/UDP/ICMP portscans, ICMP/TCP flooding, DoS attacks and other rate-based attacks (“frequency”). The administrator can configure threshold that mimics normal expected frequency and configure the TOE to detect and drop events exceeding the configured thresholds. (The ST notes that the TOE’s definition of the term “threshold” matches the definition of the term “frequency” in the IPS PP module, thus ‘frequency’ rather than ‘threshold’ is selected in the IPS_ABD_EXT.1.1 requirement).

Section 7.1.1 in the ST indicates that Intrusion rules can specify protocols: ICMPv4, ICMPv6, IPv4, IPv6, TCP and UDP.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.12.3.3 IPS10:IPS_NTA_EXT.1.3

TSS Assurance Activities: The evaluator shall verify that the TSS identifies all interface types capable of being deployed in the modes of promiscuous, and or inline mode as well as the interfaces necessary to facilitate each deployment mode (at a minimum, the interfaces need to support inline mode). The evaluator shall also check that the TSS provides a description for how the management interface is logically distinct from any sensor interfaces.

Section 6.1 (IPS_NTA_EXT.1) states that the administrator can configure the Sensor in either a passive or inline deployment. In a passive (promiscuous) IPS deployment, the Sensor monitors traffic flowing across a network using a switch SPAN or mirror port. The SPAN or mirror port allows for traffic to be copied from other ports on the switch. This provides the system visibility within the network without being in the flow of network traffic. When configured in a passive deployment, the system cannot take certain actions such as blocking or shaping traffic. The administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as passive interfaces and deploy the intrusion policy to that interface via security zone (i.e., the interface is added to the zone). In an inline IPS deployment, the administrator configures the Sensor transparently on a network segment by binding two ports together. The administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as inline interfaces then assign a pair of inline interfaces to an inline set. The intrusion policy is then deployed to that inline set via security zone. The management interface (typically eth0) is separate from the other data monitoring interfaces (used as passive or inline) on the Sensor. It is used to set up and register the Sensor to the FMC.

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions on how to deploy each of the deployment methods outlined in the TSS. The evaluator shall also verify that the operational guidance provides instructions of applying IPS policies to interfaces for each deployment mode. If the management interface is configurable, the evaluator shall verify that the operational guidance explains how to configure the interface as a management interface.

The evaluator shall verify that the operational guidance explains how the TOE sends commands to remote traffic filtering devices if this functionality is supported.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement indicates that an administrator can configure the TOE to be deployed in either Passive or Inline mode. For this requirement, Passive Mode maps to Promiscuous (listen-only) mode and provides insight to the network traffic flow without being able to modify it, while Inline mode maps to inline mode and can drop or shape traffic. Subsections “Passive Deployment” and “Inline Deployment” provide instructions for configuring these different modes. Section “Before Installation” clarifies that the management interface must be separate from the passive and inline interfaces.

Remote traffic filtering is not supported.

Testing Assurance Activities: Testing for this element is performed in conjunction with testing where promiscuous and inline interfaces are tested.

The tests associated with this requirement have been completed in subsequent assurance activities in which promiscuous and inline interfaces are tested (e.g. tests for IPS_SBD_EXT.1) and in the requirement of FTP_ITC.1 (if the ST author selects other interface types) and/or FTP_TRP.1 (for interfaces in management mode) in the base PP.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.12.4 Signature-Based IPS Functionality - per TD0722 (IPS10:IPS_SBD_EXT.1)

2.12.4.1 IPS10:IPS_SBD_EXT.1.1

TSS Assurance Activities: The evaluator shall verify that the TSS describes what is comprised within a signature rule.

The evaluator shall verify that each signature can be associated with a reaction specified in IPS_SBD_EXT.1.5.

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

Section 7.1 in the ST states that an intrusion rule is a set of keywords and arguments that the system uses to detect attempts to exploit vulnerabilities on your network. As the system analyzes network traffic, it compares packets against the conditions specified in each rule. If the packet data matches all the conditions specified in a rule, the rule triggers. If a rule is an alert rule, it generates an intrusion event. If it is a pass rule, it ignores the traffic. For a drop rule in an inline deployment, the system drops the packet and generates an event.

All rules contain two logical sections: the rule header and the rule options. The rule header contains:

• the rule's action or type

• the protocol

• the source and destination IP addresses and netmasks

• direction indicators showing the flow of traffic from source to destination

• the source and destination ports

The rule options section contains:

• event messages

• keywords and their parameters and arguments

• patterns that a packet’s payload must match to trigger the rule

• specifications of which parts of the packet the rules engine should inspect

Section 6.1 (IPS_SBD_EXT.1) in the ST states that an administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as passive interfaces and deploy the intrusion policy to that interface via security zone (i.e., the interface is added to the zone). The administrator can configure one or more physical ports (Gigabit ethernet interfaces) on a managed Sensor as inline interfaces then assign a pair of inline interfaces to an inline set. The intrusion policy is then deployed to that inline set via security zone. The management interface (typically eth0) is separate from the other data monitoring interfaces (used as passive or inline) on the Sensor. It is used to set up and register the Sensor to the FMC.

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions with how to create and/or configure rules using the following protocols and header inspection fields:

- IPv4: version; header length; packet length; ID; IP flags; fragment offset; time to live (TTL); protocol; header checksum; source address; destination address; IP options; and, if selected, type of service (ToS).

- IPv6: Version; payload length; next header; hop limit; source address; destination address; routing header; and, if selected, traffic class and/or flow label.

- ICMP: type; code; header checksum; and, if selected, other header fields (varies based on the ICMP type and code).

- ICMPv6: type; code; and header checksum.

- TCP: Source port; destination port; sequence number; acknowledgement number; offset; reserved; TCP flags; window; checksum; urgent pointer; and TCP options.

- UDP: source port; destination port; length; and UDP checksum.

The evaluator shall verify that the operational guidance provides instructions with how to select and/or configure reactions specified in IPS_SBD_EXT.1.5 in the signature rules.

(TD0722 applied)

The table from section "Intrusion Rule Options and Keywords" in the Supplement specifies all the supported protocols and filtering options available to an administrator. This includes all of the protocols and fields specified in the assurance activity.

Section "Intrusion Rules Editor" in the Supplement provides the instructions for how to select and configure alert rules, pass rules and drop rules.

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

Test 1: The evaluator shall use the instructions in the operational guidance to test that packet header signatures can be created and/or configured with the selected and/or configured reactions specified in IPS_SBD_EXT.1.5 for each of the attributes listed below. Each attribute shall be individually assigned to its own unique signature:

- IPv4: Version; Header Length; Packet Length; ID; IP Flags; Fragment Offset; Time to Live (TTL); Protocol; Header Checksum; Source Address; Destination Address; IP Options; and, if selected, type of service (ToS).

- IPv6: Version; payload length; next header; hop limit; source address; destination address; routing header; and, if selected, traffic class and/or flow label.

- ICMP: Type; Code; Header Checksum; and, if selected, other Header fields (varies based on the ICMP type and code).

- ICMPv6: Type; Code; and Header Checksum.

- TCP: Source port; destination port; sequence number; acknowledgement number; offset; reserved; TCP flags; window; checksum; urgent pointer; and TCP options.

- UDP: Source port; destination port; length; and UDP checksum.

The evaluator shall generate traffic to trigger a signature and shall then use a packet sniffer to capture traffic that ensures the reactions of each rule are performed as expected.

Test 2: The evaluator shall repeat the test above to ensure that signature-based IPS policies can be defined for each distinct network interface type capable of applying signatures as supported by the TOE.

(TD0722 applied)

Test 1: The evaluator created new local rules and identified existing rules that inspected each field in the protocols as listed in the test above. The evaluator then set up a packet capture to capture traffic from each rule and verified that the appropriate signatures are present and that the reactions of each rule are performed as expected.

Test 2: The TOE supports only the Ethernet based networking that was tested during IPS_SBD_EXT.1.1-t1.

2.12.4.2 IPS10:IPS_SBD_EXT.1.2

TSS Assurance Activities: The evaluator shall verify that the TSS describes what is comprised within a string-based detection signature.

The evaluator shall verify that each packet payload string-based detection signature can be associated with a reaction specified in IPS_SBD_EXT.1.5.

Section 6.1 (IPS_SBD_EXT.1) in the ST states that the operations associated with the anomaly-based IPS policies are allow the traffic flow for any sensor interface in any mode and allow the traffic flow and block/drop the traffic flow in inline mode. Administrators can define strings to match URLs/URIs, and web page content for pattern-matching.

Section 7.1.2 in the ST describes the rule options that can be used to match information in a packet for a given rule (where every rule has an action of pass, drop or alert). One option, the content keyword, specifies a data pattern inside a packet. The pattern may be presented in the form of an ASCII string or as binary data in the form of hexadecimal characters.

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions with how to configure rules using the packet payload string-based detection fields defined in IPS_SBD_EXT.1.2.

The evaluator shall verify that the operational guidance provides instructions with how to configure reactions specified in IPS_SBD_EXT.1.5 for each string-based detection signature.

The evaluator shall verify that the operational guidance provides instructions with how rules are associated with distinct network interfaces that are capable of being associated with signatures.

Section “Intrusion Rules Editor” in the Supplement outlines the ability for the administrator to specify patterns that a payload within a packet must match to trigger the rule. The rule can then be configured to drop and log, allow, or log the packet.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement provides instructions for configuring FTD interfaces in either a passive or inline IPS deployment. No FTD interface will forward traffic until policies have been configured and applied to that interface.

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

Test 1: The evaluator shall use the instructions in the operational guidance to test that packet payload string-based detection rules can be assigned to the reactions specified in IPS_SBD_EXT.1.5 using the attributes specified in IPS_SBD_EXT.1.2. However it is not required (nor is it feasible) to test all possible strings of protocol data, the evaluator shall ensure that a selection of strings in the requirement is selected to be tested. At a minimum at least one string using each of the following attributes from IPS_SBD_EXT.1.2 should be tested for each protocol. The evaluator shall generate packets that match the string in the rule and observe the corresponding reaction is as configured.

- Test at least one string of characters for ICMPv4 data: beyond the first 4 bytes of the ICMP header.

- Test at least one string of characters for ICMPv6 data: beyond the first 4 bytes of the ICMP header.

- TCP data (characters beyond the 20 byte TCP header):

i) Test at least one FTP (file transfer) command: help, noop, stat, syst, user, abort, acct, allo, appe, cdup, cwd, dele, list, mkd, mode, nlst, pass, pasv, port, pass, quit, rein, rest, retr, rmd, rnfr, rnto, site, smnt, stor, stou, stru, and type.

ii) HTTP (web) commands and content:

(1) Test both GET and POST commands

(2) Test at least one administrator-defined strings to match URLs/URIs, and web page content.

iii) Test at least one SMTP (email) state: start state, SMTP commands state, mail header state, mail body state, abort state.

iv) Test at least one string in any additional attribute type defined within the 'other types of TCP payload inspection' assignment, if any other types are specified.

- Test at least one string of UDP data: characters beyond the first 8 bytes of the UDP header;

- Test at least one string for each additional attribute type defined in the 'other types of packet payload inspection' assignment, if any other types are specified.

Test 2: The evaluator shall repeat Test 1 above to ensure that signature-based IPS policies can be defined for each distinct network interface type capable of applying signatures as supported by the TOE.

Test 1: The evaluator created new local rules and identified existing rules that inspected each string pattern as listed in the test above. The TOE was able to match the intrusion policy with each of those fields in each packet and act according to the policy. The evaluator repeated the tests with the TOE configured in promiscuous mode. The evaluator observed that all attacks generated IPS intrusion events regardless of inline or Promiscuous. The evaluator also observed that when configured for inline mode of operation, the TOE blocked packets.

Note that Passive mode as described in the AGD is used for Promiscuous mode in this case.

Test 2: The evaluator repeated the UDP string matching test sending 40 packets instead of just one. The TOE detected and blocked all 40 packets.

2.12.4.3 IPS10:IPS_SBD_EXT.1.3

TSS Assurance Activities: The evaluator shall verify that the TSS describes how the attacks defined in IPS_SBD_EXT.1.3 are processed by the TOE and what reaction is triggered when these attacks are identified.

Section 6.1 (IPS_SBD_EXT.1) in the ST explains that the TOE can be configured to use intrusion rules to detect various attacks such as Teardrop, Bonk, Ping of Death, etc. The administrators can use pre-defined rules or create custom rules to detect these attacks and many more. The 'action' for a given rule defines the reaction when a rule detecting these attacks is triggered.

Section 7.1.1 in the ST describes the 'action (alert)' option which generates an intrusion event when triggered and operations (allow, block/drop) associated with policies.

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions with configuring rules to identify the attacks defined in IPS_SBD_EXT.1.3 as well as the reactions to these attacks as specified in IPS_SBD_EXT.1.5.

Section “Specific Attacks” in the Supplement contains a table showing pre-configured rules that can be used to identify and prevent all of the attacks defined in IPS_SBD_EXT.1.3.

Testing Assurance Activities: The evaluator shall create and/or configure rules for each attack signature in IPS_SBD_EXT.1.3. For each attack, the TOE should apply its corresponding signature and enable it to each distinct network interface type capable of applying the signatures. The evaluator shall use packet captures to ensure that the attack traffic is detected by the TOE and a reaction specified in IPS_SBD_EXT.1.5 is triggered and stops the attack. Each attack should be performed one after another so as to ensure that its corresponding signature successfully identified and appropriately reacted to a particular attack.

Test 1: The evaluator created rules for each type of attack and verified via packet capture and syslogs that each attack was detected and dropped or logged depending on which mode the TOE was configured in. The evaluator observed that all attack traffic generated IPS intrusion events regardless of Inline or Promiscuous modes.

2.12.4.4 IPS10:IPS_SBD_EXT.1.4

TSS Assurance Activities: The evaluator shall verify that the TSS describes how the attacks defined in IPS_SBD_EXT.1.4 are processed by the TOE and what reaction is triggered when these attacks are identified.

Section 6.1 (IPS_SBD_EXT.1) in the ST indicates that during network analysis, the TOE can detect specific threats, such as IP/TCP/UDP/ICMP portscans, ICMP/TCP flooding, DoS attacks and other rate-based attacks (“frequency”). The attacks specified here include all of those attacks identified by IPS_SBD_EXT.1.4.

When the system identifies a possible intrusion, it generates an intrusion or preprocessor event (sometimes collectively called intrusion events). Managed Sensors transmit their events to the Firepower Management Center (FMC), where the administrators can view the aggregated data and gain a greater understanding of the attacks against their network assets. In an inline deployment, managed Sensors can also drop or replace packets that are known to be harmful.

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions with configuring rules to identify the attacks defined in IPS_SBD_EXT.1.4 as well as the reactions to these attacks as specified in IPS_SBD_EXT.1.5.

Section “Portscan Detection” in the Supplement specifically outlines IP, TCP, UDP, and ICMP portscan detection methods and provides instructions for the administrator to configure these methods.

Section “Rate-Based Attack Preventions” in the Supplement outlines the risk of a DoS attack and provides instructions for the administrator to configure a rule to limit excess activity by users. The administrator then has the option to configure how such detected events are handled (log, drop and log, allow). Step 7 in this section specifically provides an option to help prevent the flooding of a host by preventing incomplete connections.

Section “Specific Attacks” also mentions one specific type of DoS that a rule already exists for that the administrator can utilize.

Testing Assurance Activities: The evaluator shall configure individual signatures for each attack in IPS_SBD_EXT.1.4. For each attack, the TOE should apply its corresponding signature and enable it to each distinct network interface type capable of applying signatures. The evaluator shall use packet captures to ensure that the attack traffic is detected by the TOE and a reaction specified in IPS_SBD_EXT.1.5 is triggered and stops the attack. Each attack should be performed one after another so as to ensure that its corresponding signature successfully identified and appropriately reacted to a particular attack.

Test 1: The evaluator configured the ‘signature’ for the attack traffic identified in IPS_SBD_EXT.1.4, then followed the abstract procedures when the TOE was configured as an inline mode device. The evaluator repeated the tests with the TOE configured as a Promiscuous device. The evaluator observed that all attack traffic generated IPS intrusion events regardless of Inline or Promiscuous mode. The evaluator also observed that when configured for inline mode of operation, the TOE blocked packets.

2.12.4.5 IPS10:IPS_SBD_EXT.1.5

TSS Assurance Activities: None Defined

Guidance Assurance Activities: The guidance EAs for this element are performed in conjunction with IPS_SBD_EXT.1.1, IPS_SBD_EXT.1.3, and IPS_SBD_EXT.1.4.

The guidance EAs for this element are performed in conjunction with IPS_SBD_EXT.1.1, IPS_SBD_EXT.1.3, and IPS_SBD_EXT.1.4.

Testing Assurance Activities: The test EAs for this element are performed in conjunction with those for IPS_SBD_EXT.1.1, IPS_SBD_EXT.1.2, IPS_SBD_EXT.1.3, and IPS_SBD_EXT.1.4.

The test EAs for this element are performed in conjunction with those for IPS_SBD_EXT.1.1, IPS_SBD_EXT.1.2, IPS_SBD_EXT.1.3, and IPS_SBD_EXT.1.4.

2.12.4.6 IPS10:IPS_SBD_EXT.1.6

TSS Assurance Activities: None Defined

Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides configuration instructions, if needed, to detect payload across multiple packets

Section “Intrusion Rules Editor” in the Supplement outlines the ability for the administrator to specify patterns that a payload within a packet must match to trigger the rule. The rule can then be configured to drop and log, allow, or log the packet.

Section “Passive vs Inline Mode and Default Traffic Flows” in the Supplement provides instructions for configuring FTD interfaces in either a passive or inline IPS deployment. No FTD interface will forward traffic until policies have been configured and applied to that interface.

Section “TCP Stream Assembly” in the Supplement provides a reference, including hyperlink, to an online guide which describes how the stream preprocessor collects and reassembles all the packets that are part of a TCP session’s communication stream. This allows the rules engine to inspect the stream as a single, reassembled entity rather than inspecting only the individual packets that are part of a given stream. Stream reassembly allows the rules engine to identify stream-based attacks, which it may not detect when inspecting individual packets.

Guidance is provided for how an administrator can specify which communication streams the rules engine reassembles based on network needs.

Testing Assurance Activities: The evaluator shall repeat one of the tests in IPS_SBD_EXT.1.2 Test 1 but generate multiple non-fragmented packets that contain the string in the rule defined. The evaluator shall verify that the malicious traffic is still detected when split across multiple non-fragmented packets.

The evaluator repeated the UDP string matching test sending 40 packets instead of just one. The TOE detected and blocked all 40 packets.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

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 Development (ADV)

3.1.1 Basic Functional Specification (ADV_FSP.1)

Assurance Activities: The EAs for this assurance component focus on understanding the interfaces (e.g., application programing 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). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. 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.

The set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

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 uses 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 an interface that is explicitly 'mapped' to invoke the desired functionality. For example, generating a random bit string, destroying a cryptographic key that is no longer needed, or the TSF failing to a secure state, are capabilities that may be specified in SFRs, but are not invoked by an interface.

However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

For this PP, 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.

3.2 Guidance documents (AGD)

3.2.1 Operational User Guidance (AGD_OPE.1)

Assurance Activities: The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps (per TD0536):

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.

In addition, the evaluator performs the EAs specified below.

The evaluator shall ensure the Operational guidance documentation is distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that 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 engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

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 engine associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic engines was not evaluated nor tested during the CC evaluation of the TOE.

b) The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

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

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

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

The **FTD Admin Guide** and **FXOS Admin Guide** provides instructions for configuring CC and FIPS mode for each TOE component such that only the cryptographic algorithms and parameters used for the evaluated configuration are available and that it is clear that no other cryptographic engines have been evaluated or tested. There are warnings and notes throughout both of these Guides and the **Supplement** guide regarding use of functions that are and are not allowed in the evaluated configuration. There are also specific settings identified that must be enabled or disabled in order to remain CC compliant. The process for updating the TOE is described above in NDcPP22e:FPT_TUD_EXT.1.

3.2.2 Preparative Procedures (AGD_PRE.1)

Assurance Activities: As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

It is noted that specific requirements for Preparative Procedures are defined in [SD] for distributed TOEs as part of the Evaluation Activities for FCO_CPC_EXT.1 and FTP_TRP.1(2)/Join.

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 administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that 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 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 to manage the security of the TSF as a product and as a component of the larger operational environment.

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

The preparative procedures must

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

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

The evaluation team had the following documents to use when configuring the TOE:

The evaluation team had the documents identified in Section 1.2 of this AAR to use when configuring the TOE.

In some instances, the document referenced general Cisco manuals which the evaluation could find on the Cisco web site. The completeness of the documentation is addressed by their use in the Assurance Activities carried out in the evaluation.

3.3 Life-cycle support (ALC)

3.3.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activities: This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a 'hard label' (e.g., stamped into the metal, paper label) or a 'soft label' (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

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.

The evaluator verified that the ST, TOE and Guidance 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.

3.3.2 TOE CM Coverage (ALC_CMS.1)

Assurance Activities: Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

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

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

3.4 Tests (ATE)

3.4.1 Independent Testing - Conformance (ATE_IND.1)

Assurance Activities: Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes 'evaluated configuration' instructions). The focus of the testing is to confirm that the requirements specified in Section 5.1.7 are being met. The Evaluation Activities in [SD] identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

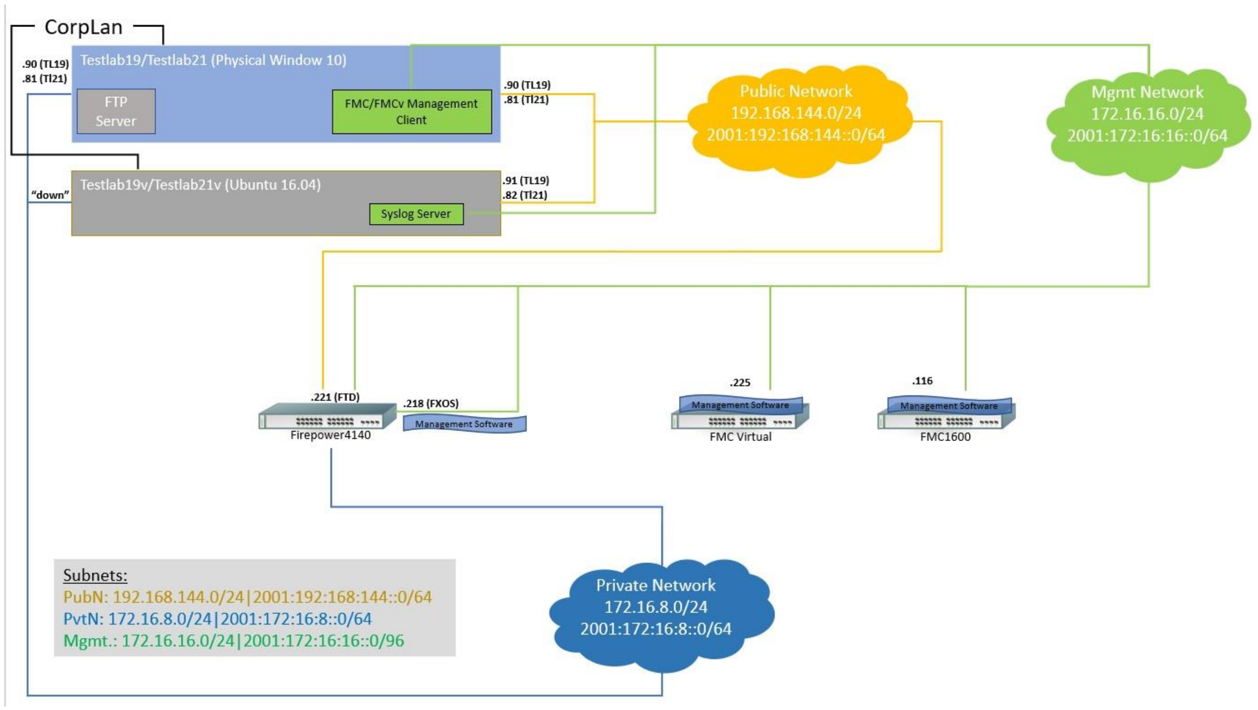
The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

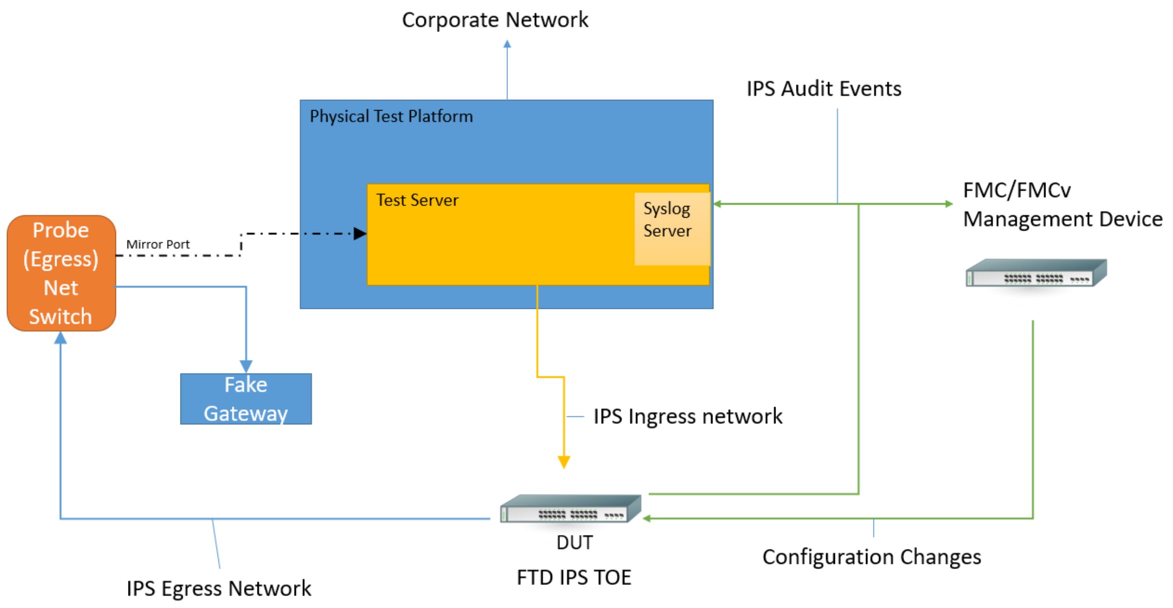
The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

The evaluator should 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 section 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.





TOE Platforms:

- Cisco FMC1600 running FMC v7.4
- Cisco FMCv running on ESXi 7.0 with FMC v7.4
- Cisco Firepower 4140 running Firepower Threat Defense v7.4 (FP4140FTD)

Supporting Products:

- Cisco Secure Client - AnyConnect (Android), version 5.0
- Samsung Galaxy S21 Ultra 5G (for use with AnyConnect Android)

Supporting Software:

- Windows 10.0
- Wireshark version 2.6.6
- Windows SSH Client – Putty version 0.68 (used to connect to device console and SSH)

The Gossamer Test servers with an Ubuntu environment acted as platforms to initiate testing. The test servers also acted as a syslog server.

- Openssh-client version 7.2p2
- Big Packet Putty version 6.2
- Nmap version 7.01
- Tcpdump version 4.9.3
- Libpcap version 1.7.4
- Stunnel version 5.30
- Openssl version 1.0.2g
- Strongswan version 5.2.5
- Rsyslog version 8.16.0
- Nping version 0.7.01
- hping3 version 3.0.0-alpha-2
- TestSSL.sh version 2.9dev
- Python version 2.7.12 & 3.5.2
    - Scapy Version 2.5.0

3.5 Vulnerability assessment (AVA)

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

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, 6.3.3]

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 3.5.1.2 and 3.6.1.2.

The evaluator formulates hypotheses in accordance with process defined in Appendix A. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.

Section “Vulnerability Mitigation” in the **Supplement** provides steps to follow to ensure that the TOE is operating with all potential vulnerabilities mitigated. 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 evaluation team performed a public search for vulnerabilities in order to ensure there are no publicly known and exploitable vulnerabilities in the TOE from the following sources:

- 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),
- cve.org CVE Database (https://www.cve.org/),
- Tenable Network Security (http://nessus.org/plugins/index.php?view=search),
- Offensive Security Exploit Database (https://www.exploit-db.com/)

On 02/07/2025 with the following search terms:

- "ftd 7.4"
- "FXOS 2.14"
- "Firepower threat defense"
- "Firepower management center"
- "cisco ssl fom",
- "Virtual fmc fom"
- "Cisco Security Crypto"
- "Openssh"
- "CiscoSSH"
- "CiscoSSL"
- "Snort"
- "ESXi 7.0"
- "Intel Xeon Bronze"
- "Intel Xeon Silver"
- "Intel Xeon Gold"
- "Intel Xeon Platinum"
- "Intel Xeon D"
- "AMD EPYC"
- "Firepower 4100"

For each resulting vulnerability matching the search terms, the evaluator examined the vulnerability and determined that the vulnerability was either not applicable to the TOE, or otherwise not exploitable.

3.5.2 ADDITIONAL FLAW HYPOTHESES (AVA_VLA.1)

The TOE utilizes TLS ciphersuites that contain the RSA algorithm in the key exchange portion of the TLS handshake. The evaluator used testssl.sh (https://testssl.sh/bleichenbacher/), which is a third-party developed script that tests TLS servers for the ROBOT vulnerability. The evaluator concluded that none of the TOE's TLS servers that use RSA key exchanges are vulnerable to the ROBOT attack.