FortiGate/FortiOS 6.4

Assurance Activity Report

Version 1.1

March 2023

Document prepared by

Lightship Security

www.lightshipsec.com

Document History

Version	Date	Author	Reviewer	Description
0.1	30-Nov-21	K Steiner		Initial Draft
0.2	12 Dec 21	T. Marconnet		PP-Modules added
0.3	11 Jan 22	T. Marconnet		AGD eval
0.4	26 Apr 22	T. Marconnet		Various fixes
0.5	30 Jan 23	O. Oztekin		AGD updates MOD_IPS v1.0 PP-Module removed
0.6	13 Feb 23	K. Yoshino	G. McLearn	Test updates
1.0	15 Feb 23	K. Yoshino		Addressed review comments Updated for check-out
1.1	6 Mar 23	K. Yoshino	C. Cantlon	Updated to address ECR comments

Table of Contents

1 Introduction

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

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	NIAP Common Criteria Evaluation and Validation Scheme
Evaluation Facility	Lightship Security
Developer/Sponsor	Fortinet, Inc.
TOE	FortiGate/FortiOS 6.4 Version 6.4 (FIPS-CC-64-6)
Security Target	FortiGate/FortiOS 6.4 Security Target, v1.1
Protection Profile	PP-Configuration for Network Devices, Stateful Traffic Filter Firewalls, and Virtual Private Network (VPN) Gateways, v1.1 i) collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020 (NDcPP) ii) PP-Module for Stateful Traffic Filter Firewalls, v1.4e (MOD_CPP_FW) iii) PP-Module for Virtual Private Network (VPN) Gateways, v1.1 (MOD_VPNGW)

1.2 Evaluation Methods

2 The evaluation was performed using the methods, tools and standards identified in Table 2.

Table 2: Evaluation Methods

Evaluation Criteria	CC v3.1R5
Evaluation Methodology	CEM v3.1R5
Supporting Documents	• Evaluation Activities for Network Device cPP, December-2019, Version 2.2 (NDcPP-SD) • Evaluation Activities for Stateful Traffic Filter Firewalls PP-Module, June 2020, v1.4 + Errata 20200625 • Supporting Document for PP-Module for Virtual Private Network (VPN) Gateways Version 1.1, 18 June 2020
Interpretations	See Table 3

Table 3: NIAP Technical Decisions

TD #	Name	Applicable PP/Module	Rationale if N/A
TD0527	Updates to Certificate Revocation Testing (FIA_X509_EXT.1)	CPP_ND_V2.2E	
TD0528	NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	CPP_ND_V2.2E	N/A. The TOE does not claim NTP
TD0536	NIT Technical Decision for Update Verification Inconsistency	CPP_ND_V2.2E	
TD0537	NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	CPP_ND_V2.2E	
TD0538	NIT Technical Decision for Outdated link to allowed-with list	CPP_ND_V2.2E	
TD0545	NIT Technical Decision for Conflicting FW rules cannot be configured (extension of Rfl#201837)	MOD_CPP_FW_V1.4E	
TD0546	NIT Technical Decision for DTLS - clarification of Application Note 63	CPP_ND_V2.2E	
TD0547	NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	CPP_ND_V2.2E	
TD0549	Consistency of Security Problem Definition update for MOD_VPNGW_v1.0 and MOD_VPNGW_v1.1	MOD_VPNGW_V1.1	
TD0551	NIT Technical Decision for Incomplete Mappings of OEs in FW Module v1.4+Errata	MOD_CPP_FW_V1.4E	
TD0555	NIT Technical Decision for RFC Reference incorrect in TLSS Test	CPP_ND_V2.2E	
TD0556	NIT Technical Decision for RFC 5077 question	CPP_ND_V2.2E	
TD0563	NiT Technical Decision for Clarification of audit date information	CPP_ND_V2.2E	
TD0564	NiT Technical Decision for Vulnerability Analysis Search Criteria	CPP_ND_V2.2E	
TD0569	NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7	CPP_ND_V2.2E	N/A. The TOE does not claim FCS_DTLSS_EXT.1.7

TD #	Name	Applicable PP/Module	Rationale if N/A
TD0570	NiT Technical Decision for Clarification about FIA_AFL.1	CPP_ND_V2.2E	
TD0571	NiT Technical Decision for Guidance on how to handle FIA_AFL.1	CPP_ND_V2.2E	
TD0572	NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	CPP_ND_V2.2E	
TD0580	NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	CPP_ND_V2.2E	
TD0581	NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	CPP_ND_V2.2E	
TD0590	Mapping of operational environment objectives	MOD_VPNGW_V1.1	
TD0591	NIT Technical Decision for Virtual TOEs and hypervisors	CPP_ND_V2.2E	
TD0592	NIT Technical Decision for Local Storage of Audit Records	CPP_ND_V2.2E	
TD0597	VPN GW IPv6 Protocol Support	MOD_VPNGW_V1.1	
TD0631	NIT Technical Decision for Clarification of public key authentication for SSH Server	CPP_ND_V2.2E	
TD0632	NIT Technical Decision for Consistency with Time Data for vNDs	CPP_ND_V2.2E	
TD0633	NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	CPP_ND_V2.2E	
TD0634	NIT Technical Decision for Clarification required for testing IPv6	CPP_ND_V2.2E	
TD0635	NIT Technical Decision for TLS Server and Key Agreement Parameters	CPP_ND_V2.2E	
TD0636	NIT Technical Decision for Clarification of Public Key User Authentication for SSH	CPP_ND_V2.2E	N/A. The TOE does not claim FCS_SSHC_EXT.1
TD0638	NIT Technical Decision for Key Pair Generation for Authentication	CPP_ND_V2.2E	N/A. The TOE is not distributed.

TD #	Name	Applicable PP/Module	Rationale if N/A
TD0639	NIT Technical Decision for Clarification for NTP MAC Keys	CPP_ND_V2.2E	
TD0670	NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	CPP_ND_V2.2E	

1.3 Reference Documents

Table 4: List of Reference Documents

Ref	Document
[ST]	FortiGate/FortiOS 6.4 Security Target v1.1, March 2023
[FNLOG]	FortiOS - Log Reference, Version 6.4.9, May 20, 2022, 01-649-619093-20220520
[ADMIN]	FortiOS – Administration Guide, Version 6.4.9, August 22, 2022 01-649-607590-20220822
[CLI]	FortiOS - CLI Reference, Version 6.4.9, April 26, 2022, 01-649-684766-20220426
[SUPP]	FortiOS 6.4 and FortiGate NGFW Appliances FIPS140-2 and NDcPP Common Criteria Technote, March 3, 2023 01-649-0773518-20230303
[CCLOG]	FortiOS 6.4 and FortiGate NGFW Appliances, NDcPP Common Criteria Logging Addendum, February 27, 2023 01-649-887811-20230227
[LoP]	FortiOS - Parallel Path Processing, version 6.4.0, January 25, 2021, 01-640-619132-20210125
[HWA]	FortiOS – Hardware Acceleration Guide, version 6.4.9, January 4, 2023, 01-649-538746-20230104
[ADMIN-VM]	FortiOS - VMware ESXi Administration Guide Version 6.4, October 6, 2021, 01-640-619610-20211006

2 TOE Details

2.1 Overview

1 The TOE is a firewall that includes Virtual Private Network (VPN) and packet filtering capabilities. An industry term for this TOE type is Next-Generation Firewall (NGFW).

2 The TOE provides the following security functions:

a) **Security Audit.** The TOE generates logs for auditable events. These logs can be stored locally in protected storage and/or exported to an external audit server via a secure channel.

b) **Cryptographic Support.** The TOE implements a variety of key generation and cryptographic methods to provide protection of data both in transit and at rest within the TOE. In the evaluated configuration, the TOE is in FIPS mode to support the cryptographic functionality.

c) **Residual Data Protection.** The TOE ensures that data cannot be recovered once deallocated.

d) **Identification and Authentication.** The TOE implements mechanisms to ensure that users are both identified and authenticated before any access to TOE functionality or TSF data is granted.

e) **Security Management.** The TOE provides a suite of management functionality, allowing for full configuration of the TOE by an authorized administrator.

f) **Protection of the TSF.** The TOE implements a number of protection mechanisms (including authentication requirements, self-tests and trusted update) to ensure the protection of the TOE and all TSF data.

g) **TOE Access.** The TOE provides session management functions for local and remote administrative sections.

h) **Trusted Path/Channels.** The TOE provides secure channels between itself and local/remote administrators and other devices to ensure data security during transit.

i) **Stateful Traffic and Packet Filtering.** The TOE allows for the configuration and enforcement of stateful packet filtering/firewall rules on all traffic traversing the TOE.

2.2 TOE Models

The physical boundary of the TOE includes the FortiGate hardware models shown in Table 5 and the virtual appliances and related hardware shown in Table 6. The virtual appliances are evaluated as virtual Network Devices (vND), which is case 1 of Section 1.2 of NDcPP v2.2e.

Table 5: TOE Hardware Models

Model	CPU	Architecture	RAM	Boot	Storage	ASIC	Entropy	CAVP
FG-61E	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-61F	Fortinet SoC4	ARMv8	2 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242

Model	CPU	Architecture	RAM	Boot	Storage	ASIC	Entropy	CAVP
FWF-61E	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FWF-61F	Fortinet SoC4	ARMv8	2 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-81E	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-81E-PoE	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-81F	Fortinet SoC4	ARMv8	4 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-81F-2R	Fortinet SoC4	ARMv8	4 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-81F-2R-3G4G-PoE	Fortinet SoC4	ARMv8	4 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-81F-2R-PoE	Fortinet SoC4	ARMv8	4 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-81F-PoE	Fortinet SoC4	ARMv8	4 GB	8GB	128GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-90E	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-91E	Fortinet SoC3	ARMv7-A	2 GB	8GB	128GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-101E	Fortinet SoC3	ARMv7-A	4 GB	8GB	480GB	CP9Lite	SoC3	A2225 A2269 A2241
FG-101F	Fortinet SoC4	ARMv8	4 GB	8GB	480GB	CP9XLite	SoC4	A2225 A2269 A2242
FG-201E	Intel Celeron G1820	Haswell	4GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240

Model	CPU	Architecture	RAM	Boot	Storage	ASIC	Entropy	CAVP
FG-201F	Intel Xeon D-1627	Hewitt Lake	8GB	30GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-301E	Intel i5-6500	SkyLake	8GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-401E	Intel i5-8500	Coffee Lake	8GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-501E	Intel i7-6700	SkyLake	16GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-601E	Intel i7-8700	Coffee Lake	16 GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-1101E	Intel Xeon E-2186G	Coffee Lake	16 GB	16GB	960GB	CP9	CP9	A2225 A2269 A2240
FG-1801F	Intel Xeon W-3223	Cascade Lake	24GB	30GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-1801F-DC	Intel Xeon W-3223	Cascade Lake	24GB	30GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-2000E	Intel Xeon E5-1660v4	Broadwell	32 GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-2201E	Intel Xeon Gold 6126	SkyLake	24 GB	16GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-2500E	Intel Xeon E5-1650v3	Haswell	32 GB	16GB	480GB	CP9	CP9	A2225 A2269 A2240
FG-2601F	Intel Xeon Gold 6208U	Cascade Lake	48 GB	30 GB	2 TB	CP9	CP9	A2225 A2269 A2240
FG-2601F-DC	Intel Xeon Gold 6208U	Cascade Lake	48 GB	30 GB	2 TB	CP9	CP9	A2225 A2269 A2240
FG-3301E	Intel Xeon Gold 5118	SkyLake	96 GB	16GB	2TB	CP9	CP9	A2225 A2269 A2240

Model	CPU	Architecture	RAM	Boot	Storage	ASIC	Entropy	CAVP
FG-3401E	Intel Xeon Gold 6130	SkyLake	96 GB	16GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-3401E-DC	Intel Xeon Gold 6130	SkyLake	96 GB	16GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-3601E	Intel Xeon Gold 6152	SkyLake	96 GB	16GB	2TB	CP9	CP9	A2225 A2269 A2240
FG-4201F	Intel Xeon Gold 6248	Cascade Lake	384 GB	30 GB	4 TB	CP9	CP9	A2225 A2269 A2240
FG-4201F-DC	Intel Xeon Gold 6248	Cascade Lake	384 GB	30 GB	4 TB	CP9	CP9	A2225 A2269 A2240
FG-4401F	Intel Xeon Gold 6248	Cascade Lake	384 GB	30 GB	4 TB	CP9	CP9	A2225 A2269 A2240
FG-4401F-DC	Intel Xeon Gold 6248	Cascade Lake	384 GB	30 GB	4 TB	CP9	CP9	A2225 A2269 A2240
FG-5001E1	Intel Xeon E5-2690v4	Broadwell	64GB	16GB	480 GB	CP9	CP9	A2225 A2269 A2240
FG-6300F	Intel Xeon D-1567	Broadwell	192GB	16GB	2 TB	CP9	Entropy Token	A2225 A2269 A2240
FG-6301F	Intel Xeon D-1567	Broadwell	192GB	16GB	2 TB	CP9	Entropy Token	A2225 A2269 A2240
FG-6500F	Intel Xeon D-1567	Broadwell	320GB	16GB	2 TB	CP9	Entropy Token	A2225 A2269 A2240
FG-6501F	Intel Xeon D-1567	Broadwell	320GB	16GB	2 TB	CP9	Entropy Token	A2225 A2269 A2240

Table 6: TOE Virtual Appliance and Related Hardware

Model	License	Hypervisor	CPU*	Entropy	CAVP
FortiGate-VM64	VM01 (1x vCPU core and unlimited RAM)	VMware ESXi 6.7	Intel Xeon D-1559	Token via USB	A2291 A2298

Model	License	Hypervisor	CPU*	Entropy	CAVP
	VM02 (2x vCPU cores and unlimited RAM)		(Broadwell) Intel Xeon E3-1515MV5 (Skylake) Intel Xeon E-2276ME (Coffee Lake)	pass-through	
	VM04 (4x vCPU cores and unlimited RAM)				
	VM08 (8x vCPU cores and unlimited RAM)				
	VM16 (16x vCPU cores and unlimited RAM)				
	VM32 (32x vCPU cores and unlimited RAM)				
	VMUL (Unlimited vCPU cores and RAM)				

* Provided with PacStar 451/455

2.2.1 Test Platform Equivalency

3 The team used the [NDcPP] as the basis for the following equivalency rationale:

Factor	Evaluator Guidance	Description
Platform/Hardware Dependencies	If there are no identified platform/hardware dependencies, the evaluator shall consider testing on multiple hardware platforms to be equivalent. If there are specified differences between platforms/hardware, the evaluator must identify if the differences affect the cPP-specified security functionality or if they apply to non-cPP-specified functionality. If functionality specified in the cPP is dependent upon platform/hardware provided services, the product must be tested on each of the different platforms to be considered validated on that particular hardware combination. In these cases, the evaluator has the option of only retesting the functionality dependent upon the platform/hardware provided functionality. If the differences only affect non-cPP-specified functionality, the variations	The TOE is available as a hardware appliance as well as a virtual appliance running on a hypervisor. The TOE hardware appliances differ in regard to form factor, CPU, amount of RAM, amount of bootloader storage, amount of storage, ASIC and Entropy source. The amount of RAM, bootloader storage, and amount of storage are not security relevant and considered equivalent. The CPUs vary in architecture via ARMv8, ARMv7-A, Haswell, Hewitt Lake, Skylake, Coffee Lake, Cascade Lake, and Broadwell. As they relate to the TOE functionality, CPU architectures differ regarding cryptography and cryptographic extensions. This functionality was tested and confirmed for each CPU architecture by obtaining CAVP certificates. The CAVP certificates are included in the

Factor	Evaluator Guidance	Description
	may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP-specified functionality.	Security Target and subject to NIAP review and approval. The hardware TOE models utilize three types of ASIC: CP9, CP9XLite, and CP9Lite. As they relate to the TOE, the ASICs are used to accelerate IPsec. Since these relate to the TOE functionality, FCS_IPSEC_EXT.* requirements have been fully tested for each ASIC. The evaluation lab fully tested the FG-2000E with CP9 ASIC instance of the TOE and performed the FCS_IPSEC_EXT.* testing on the FG-81E with a CP9Lite ASIC and FG-81F with a CP9XLite ASIC instances of the TOE. The hardware instances of the TOE also differ in regards to the Entropy source. Each entropy source was full analysed according to [NDcPP] Annex D. The virtual appliances only differ in regards to CPU. Broadwell, Coffee Lake and Skylake As they relate to the TOE functionality, CPU architectures differ regarding cryptography and cryptographic extensions. This functionality was tested and confirmed for each CPU architecture by obtaining CAVP certificates. The CAVP certificates are included in the Security Target and subject to NIAP review and approval. The virtual appliances do not support ASIC and only claim VMware ESXi 6.7 and the USB Token for Entropy. The evaluation lab fully tested the FortiGate-VM64 virtual appliance with VMware ESXi 6.7 and Intel Xeon D-1559 CPU. The remaining virtual appliances can be considered equivalent.
Differences in TOE Software Binaries	If the model binaries are identical, the model variations shall be considered equivalent. If there are differences between model software binaries, a	All instances of the TOE binaries are built from the same source code. Each TOE binary is built for the specific hardware to accommodate the device drivers for different hardware. Each hardware specific build is a separate branch which does

Factor	Evaluator Guidance	Description
	determination must be made if the differences affect cPP-specified security functionality. If cPP-specified functionality is affected, the models are not considered equivalent and must be tested separately. The evaluator has the option of only retesting the functionality that was affected by the software differences. If the differences only affect non-PP specified functionality, the models may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP specified functionality.	not change the underlying functional binary. Since the functional binary is unchanged with the model variations there are no differences in the TOE model variations as they relate to the claimed security/cPP functionality. Thus, the model variations can be considered equivalent.
Differences in Libraries Used to Provide TOE Functionality	If there are no differences between the libraries used in various TOE models, the model variations shall be considered equivalent. If the separate libraries are used between model variations, a determination of whether the functionality provided by the library affects cPP-specified functionality must be made. If cPP-specified functionality is affected, the models are not considered equivalent and must be tested separately. The evaluator has the option of only retesting the functionality that was affected by the differences in the included libraries. If the different libraries only affect non-PP specified functionality, the models may still be considered equivalent. For each different library, the evaluator must provide an explanation of why the different libraries do or do not affect cPP specified functionality.	All instances of the TOE use the same libraries so the model variations can be considered equivalent.
TOE Management Interface Differences	If there are no differences in the management interfaces between various TOE models, the model variations shall be considered equivalent. If the product provides separate interfaces based on the model variation, a determination must be made of whether cPP-specified functionality can be configured by the different interfaces. If the interface differences affect cPP-	All hardware instances of the TOE use the same management interfaces (local serial connection, SSH CLI, and HTTPS Web UI) so the model variations can be considered equivalent. The virtual appliances of the TOE also use the same SSH CLI and HTTPS Web UI management interfaces however the local interface is accessed via the hypervisor console.

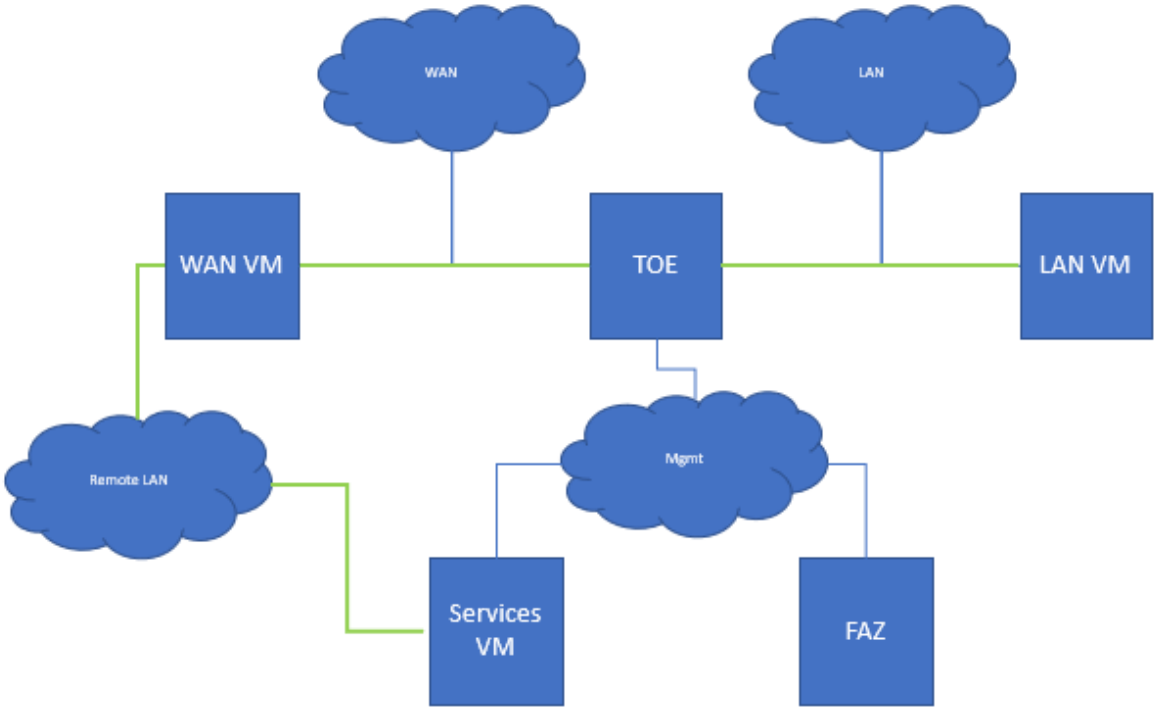
Factor	Evaluator Guidance	Description
	specified functionality, the variations are not considered equivalent and must be separately tested. The evaluator has the option of only retesting the functionality that can be configured by the different interfaces (and the configuration of said functionality). If the different management interfaces only affect non-PP specified functionality, the models may still be considered equivalent. For each management interface difference, the evaluator must provide an explanation of why the different management interfaces do or do not affect cPP specified functionality.	The cPP does not require all management activities to be performed on all management interfaces however each management interface must be exercised throughout testing. The evaluator fully tested a model 2000E hardware appliance and a FortiGate VM-64 virtual appliance, so each management interface is covered.
TOE Functional Differences	If the functionality provided by different TOE model variation is identical, the models variations shall be considered equivalent. If the functionality provided by different TOE model variations differ, a determination must be made if the functional differences affect cPP specified functionality. If cPP-specific functionality differs between models, the models are not considered equivalent and must be tested separately. In these cases, the evaluator has the option of only retesting the functionality that differs model-to-model. If the functional differences only affect non-cPP specified functionality, the model variations may still be considered equivalent. For each difference the evaluator must provide an explanation of why the difference does or does not affect cPP specified functionality.	All instances of the TOE have the same functionality and only differ based on hardware constraints such as speed and the number of ports. These differences cause no changes to functionality or any changes to the security/cPP claims. Thus, the model variations can be considered equivalent.

4 In summary the evaluator performed full end-to-end testing on the FortiGate VM-64 with VMware ESXi 6.7 and Intel Xeon D-1559 CPU and the FortiGate 2000E models. The test sample as stated above was performed on the FortiGate 81E and 81F.

2.2.2 TOE Test Configuration

5 The following diagram provides a high level overview of the test environment.

Figure 1 - Test setup



6 The green line (▬▬) identifies the path traffic may be routed over. All other connections are for the local subnet.

7 For IPsec testing, an IPsec tunnel is established between the WAN VM and the TOE.

8 For a small number of tests (e.g., physical disruption) a dedicated packet capture laptop and traffic mirroring switch were connected directly between the TOE and the rest of the environment.

9

Device / Details	Description / Use
FG-2000E HW: FortiGate FG-2000E SW: 6.4 (FIPS-CC-64-2) SW: 6.4 (FIPS-CC-64-4) SW: 6.4 (FIPS-CC-64-5)	Fully tested hardware TOE.
FG-VM64 HW: TOE Hypervisor SW: 6.4 (FIPS-CC-64-4) SW: 6.4 (FIPS-CC-64-5) SW: 6.4 (FIPS-CC-64-6) License: FortiGate-VM01	Virtual TOE tested against the NDcPP 2.2E requirements.
FG-VM64-3 HW: TOE Hypervisor SW: 6.4 (FIPS-CC-64-5)	Virtual TOE tested against the MOD_CPP_FW_V1.4E requirements.

Device / Details	Description / Use
License: FortiGate-VM08	
81E HW: FortiGate FG-81E SW: 6.4 (FIPS-CC-64-5)	HW TOE for VPN equivalency tests.
81F HW: FortiGate FG-81F SW: 6.4 (FIPS-CC-64-5)	HW TOE for VPN equivalency tests.
FG-VM64-2 HW: TOE Hypervisor SW: SW: 6.4 (FIPS-CC-64-6) License: FortiGate-VM01	VM TOE for MOD_VPNGW testing.
FAZ HW: Infrastructure Hypervisor HW: Infrastructure Hypervisor 2 SW: FortiAnalyzer v6.4.7	Log Server TLS Server Packet Captures Configured to forward logs to the Services VM.[1]
Services VM / Management Workstation HW: Infrastructure Hypervisor HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	DNS Server TLS Server CRL Server TLS/HTTPS Client SSH Client Syslog server Packet Captures
Services-3 VM / Management Workstation HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	TLS/HTTPS Client SSH Client Packet Captures
LAN VM HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	Traffic Generator (FW/VPN) Packet Captures
LAN2 VM HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	Traffic Generator (FW/VPN) Packet Captures
LAN-3 VM HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	Traffic Generator (FW/VPN) Packet Captures
WAN VM	IPsec peer

[1] The Services VM was configured in the Central time zone, so some logs include UTC-5 and UTC-4 local times, but the UTC times are the same.

Device / Details	Description / Use
HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	Traffic Generator (FW/VPN) Packet Captures
WAN-2 VM HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	IPsec peer Packet Captures
WAN-3 VM HW: Infrastructure Hypervisor 2 SW: 5.7.0-kali1-amd64	IPsec peer Packet Captures
Router	Primary Lab Router
TOE Hypervisor HW: PacStar PS451 SW: ESXi 6.7	Hypervisor for the FG-VM64
Infrastructure Hypervisor HW: Dell PowerEdge R540 SW: ESXi, 6.7.0	Hypervisor for the VMs and FAZ.
Infrastructure Hypervisor 2 HW: Dell PowerEdge R440 SW: ESXi, 7.0.3	Hypervisor for the VMs and FAZ. Note: During testing the FAZ and Services VM were migrated from Infrastructure Hypervisor to Infrastructure Hypervisor 2. There was no change to the VMs, their MAC addresses, or IP addresses based on this change.
Console Server	Local Console connection to the HW TOEs
Netgear Switch HW: ProSafe Plus GS105E	Physical disconnect packet captures
Packet Capture Laptop HW: Lenovo ThinkPad E495 SW: Windows 10 Pro	Specific/limited packet capture use cases

2.2.3 Test Tools

10 The following systems with the following tools were used:

2.2.3.1 Services VM

Tool name	Version	Description
Firefox	102.2.0esr	Web browser for accessing the Web UI.
OpenSSH	9.0p1 Debian-1+b1	SSH client for accessing the Remote CLI.
OpenSSL	3.0.5 5 Jul 2022	General purpose crypto tool.
strongSwan	U5.9.6/K5.18.0-kali7-amd64	IPsec peer for NAT testing
dnsmasq	2.86	DNS server

Tool name	Version	Description
vsftpd	3.0.3	FTP server
rsyslogd	8.2208.0 (aka 2022.08)	Syslog Server
Python	3.10.7	HTTP server
tcpdump	4.99.1	Packet capture
Wireshark	3.6.7	Analyzing packet captures
iperf	2.1.2	Bandwidth testing utility
Green Light	3.0.34	Custom Lightship Test Tool that performs protocol manipulation and corruption. Includes: • OpenSSH 8.8p1-Lightship-1.0.1 • OpenSSL 1.0.2g-LS 1 Mar 2016

2.2.3.2 Services-3 VM

Tool name	Version	Description
Firefox	91.6.0esr	Web browser for accessing the Web UI.

2.2.3.3 WAN VM & WAN-2 VM & WAN-3 VM

Tool name	Version	Description
tcpdump	4.99.0	Packet capture
Wireshark	3.6.7	Analyzing packet captures
OpenSSH	OpenSSH_8.4p1 Debian-5	SSH server
strongSwan	U5.7.1-Lightship/K5.7.0-kali1-amd64	IPsec peer
Netcat	1.10-46	TCP/UDP Server/Client
sendip	2.6-1	Arbitrary IPv6 packet creation
ftp	0.17-34.1.1	FTP client
apache2	2.4.46	HTTP server
vsftpd	3.0.3	FTP server
Nmap	7.92	Port/Protocol Scanner
Green Light	3.0.34	Custom Lightship Test Tool that performs protocol manipulation and corruption. Includes: • Scapy 2.4.4 • Python 3.9.2 • OpenSSH 8.8p1-Lightship-1.0.1 • OpenSSL 1.0.2g-LS 1 Mar 2016

2.2.3.4 LAN VM & LAN2 VM & LAN3 VM

Tool name	Version	Description
tcpdump	4.99.0	Packet capture
Wireshark	3.6.7	Analyzing packet captures
Netcat	1.10-46	TCP/UDP Server/Client
iperf	2.1.2	Bandwidth testing utility
ftp	0.17-34.1.1	FTP client
apache2	2.4.46	HTTP server
vsftpd	3.0.3	FTP server
Green Light	3.0.34	Custom Lightship Test Tool that performs protocol manipulation and corruption. Includes: • Scapy 2.4.4 • Python 3.9.2 • OpenSSH 8.8p1-Lightship-1.0.1 • OpenSSL 1.0.2g-LS 1 Mar 2016

2.2.3.5 Packet Capture Laptop

Tool name	Version	Description
Wireshark	3.6.8	Capturing and Analyzing packet

3 Evaluation Activities for SFRs

3.1 Security Audit (FAU)

3.1.1 FAU_GEN.1 Audit data generation

3.1.1.1 TSS

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

Findings:	[ST] Section 6.1 states that the TOE logs the actions (generating, importing or deleting) and key reference for cryptographic keys including CSR.

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

Findings:	The TOE is not a distributed TOE.

3.1.1.2 Guidance Documentation

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

Findings:	[CCLOG] maps specific auditable events to example logs for each applicable auditable event. Additionally, [FNLOG] provides the format and full set of possible log messages that can be generated by the TOE.

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

Findings:	The evaluator performed this activity as part of those AAs associated with ensuring the corresponding guidance documentation satisfied their independent requirements. However, overall, the evaluator considered the administrator guides published by the vendor. The evaluator reviewed the contents of the documentation and looked specifically for functionality related to the scope of the evaluation. Where there was

	missing or incomplete descriptions for the functionality such that the user could not complete the testing AAs, the evaluator requested the vendor to supply augmented guidance information. In the end, the vendor provided a more comprehensive guidance “supplement” document in the form of [SUPP].

3.1.1.3 Tests

15 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.

16 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.

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

High-Level Test Description
Ensure that the TOE displays an audit record for each of the auditable events defined for this requirement.
Findings: PASS – The evaluator performed the testing in conjunction with the testing of the security mechanisms directly. The evaluator confirmed that the TOE correctly generates audit records for the events listed in the table of audit events and administrative actions.

3.1.2 FAU_GEN.2 User identity association

3.1.2.1 TSS & Guidance Documentation

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

3.1.2.2 Tests

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

20 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.

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

3.1.3 FAU_STG_EXT.1 Protected audit event storage

3.1.3.1 TSS

21 The evaluator shall examine the TSS to ensure it describes the means by which the audit data are transferred to the external audit server, and how the trusted channel is provided.

Findings:	[ST] Section 6.1 states that the TOE transmits log data to an external FortiAnalyzer platform via TLS.

22 The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally; what happens when the local audit data store is full; and how these records are protected against unauthorized access.

Findings:	[ST] Section 6.1 states that the TOE writes the logs to hard disk. The TOE deletes the oldest records to make room for new one when the local audit data store is full.

23 The evaluator shall examine the TSS to ensure it describes whether the TOE is a standalone TOE that stores audit data locally or a distributed TOE that stores audit data locally on each TOE component or a distributed TOE that contains TOE components that cannot store audit data locally on themselves but need to transfer audit data to other TOE components that can store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs it contains a list of TOE components that store audit data locally. The evaluator shall examine the TSS to ensure that for distributed TOEs that contain components which do not store audit data locally but transmit their generated audit data to other components it contains a mapping between the transmitting and storing TOE components.

Findings:	[ST] Section 6.1 states that the TOE stores audit data locally on a hard disk.

24 The evaluator shall examine the TSS to ensure that it details the behaviour of the TOE when the storage space for audit data is full. When the option ‘overwrite previous audit record’ is selected this description should include an outline of the rule for overwriting audit data. If ‘other actions’ are chosen such as sending the new audit data to an external IT entity, then the related behaviour of the TOE shall also be detailed in the TSS.

Findings:	[ST] Section 6.1 states that the TOE deletes the oldest audit records when the storage is full.

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

Findings:	[ST] Section 6.1 states that the TOE can transmit audit data to a remote FortiAnalyzer in real-time.

26 For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how audit data transfer to the external audit server is implemented among the different TOE components (e.g. every TOE components does its own transfer or the data is sent to another TOE component for central transfer of all audit events to the external audit server).

Findings:	N/A. The TOE is not a distributed TOE.

27 For distributed TOEs the evaluator shall examine the TSS to ensure it describes which TOE components are storing audit information locally and which components are buffering audit information and forwarding the information to another TOE component for local storage. For every component the TSS shall describe the behaviour when local storage space or buffer space is exhausted.

Findings:	N/A. The TOE is not a distributed TOE.

3.1.3.2 Guidance Documentation

28 The evaluator shall also examine the guidance documentation to ensure it describes how to establish the trusted channel to the audit server, as well as describe any requirements on the audit server (particular audit server protocol, version of the protocol required, etc.), as well as configuration of the TOE needed to communicate with the audit server.

Findings:	The TOE is required to communicate with a FortiAnalyzer logging device. This information is found in the [SUPP] in the subsections under “Log Specific Settings”. The FortiAnalyzer communicates over TLS. The configuration of the logging server communication details is found in the [SUPP] and [ADMIN] guidance documents. The evaluator was able to configure the logging server using the provided guides.

29 The evaluator shall also examine the guidance documentation to determine that it describes the relationship between the local audit data and the audit data that are sent to the audit log server. For example, when an audit event is generated, is it simultaneously sent to the external server and the local store, or is the local store used as a buffer and “cleared” periodically by sending the data to the audit server.

Findings:	The [SUPP] in the subsections under “Log Specific Settings” describes the relationship between local and remote logs. The [SUPP] characterizes the local logs as being “cached” before being transmitted to the remote logging server. In the [ADMIN] section “Log and Report” (starting from page 1959), this relationship is expanded upon when describing the specific configuration items. Realtime transfer is configured using the “upload-option” setting in the CLI as per [CLI] section “config log fortianalyzer setting” (page 494). This is also mentioned under [SUPP] section “FortiAnalyzer configuration”.

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

Findings:	The TOE only claims “overwrite of the oldest audit log” and therefore additional description of this functionality is unnecessary.

3.1.3.3 Tests

31 Testing of the trusted channel mechanism for audit will be performed as specified in the associated assurance activities for the particular trusted channel mechanism. The evaluator shall perform the following additional tests for this requirement:

a) Test 1: The evaluator shall establish a session between the TOE and the audit server according to the configuration guidance provided. The evaluator shall then examine the traffic that passes between the audit server and the TOE during several activities of the evaluator's choice designed to generate audit data to be transferred to the audit server. The evaluator shall observe that these data are not able to be viewed in the clear during this transfer, and that they are successfully received by the audit server. The evaluator shall record the particular software (name, version) used on the audit server during testing. The evaluator shall verify that the TOE is capable of transferring audit data to an external audit server automatically without administrator intervention.

High-Level Test Description
Login to the TOE while capturing traffic sent to the audit server. Verify the successful authentication log is received by the audit server and is not transferred in plaintext. Record the name and version of the audit server.
Findings: PASS – The logging server is a FortiAnalyzer v6.4.7 as described in the Test Setup. The evaluator confirmed the audit log was successfully received by the audit server and was not sent in plaintext.

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

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

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

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

High-Level Test Description
Verify the oldest log file is overwritten when the configured storage space for logs is exhausted.
Findings: PASS – The evaluator confirmed that the TOE overwrites the oldest log file when the configured storage space for audit logs is filled.

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

High-Level Test Description
FAU_STG_EXT.2/LocSpace is not claimed by the TOE.
Findings: N/A

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.

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

3.2 Cryptographic Support (FCS)

3.2.1 FCS_CKM.1 Cryptographic Key Generation

3.2.1.1 TSS

32 The evaluator shall ensure that the TSS identifies the key sizes supported by the TOE. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme.

Findings:	[ST] Table 20 in Section 6.2 identifies all key sizes supported by the TOE.

3.2.1.2 Guidance Documentation

33 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key generation scheme(s) and key size(s) for all cryptographic protocols defined in the Security Target.

Findings:	Key exchange IPSec VPNs are configured as per the VPN configuration items as described in the [ADMIN] document in sections “Phase 1 configuration” and “Phase 2 configuration” starting on pages 1425 and 1441, respectively. Both sections describe the “Diffie-Hellman Group” parameter. As per [SUPP] section “Miscellaneous”, IKE should be configured to use one of DH Groups 14 (2048-bit MODP), 19 (256-bit Random ECP) or 20 (384-bit Random ECP) to match the evaluated configuration. TLS and SSH trusted paths for management and TLS trusted channels to the remote FortiAnalyzer are not modifiable by the user except for the Diffie-Hellman group which should be set to Group 14 (2048-bit modulus) as mentioned in [SUPP] section “Enabling administrative access”.

3.2.1.3 Tests

34 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

35 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*.

36 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

37 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

38 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

39 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)

40 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.

41 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

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

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

43 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.

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

45 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.

46 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

47 for each FFC parameter set and key pair.

NIAP TD0580

FFC Schemes using "safe-prime"

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

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2240, A2241, and A2242 for RSA. The vendor uses the CAVP certificates A2269 and A2298 for ECDSA. Schemes using safe primes are tested in FCS_CKM.2.1. This is described in [ST] Table 24.

3.2.2 FCS_CKM.2 Cryptographic Key Establishment

3.2.2.1 TSS

49 The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme, the evaluator shall examine the TSS to verify that it identifies the usage for each scheme. It is sufficient to provide the scheme, SFR, and service in the TSS.

Findings:	[ST] Table 21 in Section 6.2 identifies all supported key establishment schemes and their usage for each scheme.

NIAP TD0580

50 **Removed:** ~~If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall claim the TOE meets RFC 3526 Section 3.~~

Findings:	This activity was removed by TD0580

51 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
~~Diffie-Hellman (Group 14)~~ **Removed per TD0580**	~~FCS_SSHC_EXT.1~~ **Removed per TD0580**	~~Backup Server~~ **Removed per TD0580**
ECDH	FCS_IPSEC_EXT.1	Authentication Server

52 The information provided in the example above does not necessarily have to be included as a table but can be presented in other ways as long as the necessary data is available.

Findings:	[ST] Table 21 in Section 6.2 identifies the usage for each scheme.

3.2.2.2 Guidance Documentation

53 The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s).

Findings:	The TOE permits the user to configure DH groups for IPSec VPN channels. IPSec VPNs are configured as per the VPN configuration items as described in the [ADMIN] section “Site-to-site VPN” (starting at page 1449). TLS and SSH trusted paths for management and TLS trusted channels to the remote FortiAnalyzer are not modifiable by the user except for the Diffie-Hellman group which should be set to Group 14 (2048-bit modulus) as mentioned in [SUPP] section “Enabling administrative access”.

3.2.2.3 Tests

Key Establishment Schemes

54 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

55 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

56 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.

57 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.

58 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.

59 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.

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

Validity Test

61 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.

62 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).

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

Findings:	The vendor uses the CAVP certificates A2269 and A2298 for ECC Key Establishment. This is described in [ST] Table 24.

RSA-based key establishment schemes

64 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.

High-Level Test Description
The TOE does not claim "RSA-based key establishment."
Findings: N/A

NIAP TD0580 Removed:

~~***Diffie-Hellman Group 14***~~

65 ~~The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 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 Diffie-Hellman group 14.~~

66 ***FFC Schemes using "safe-prime" groups***

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

High-Level Test Description
Verify the TOE can successfully perform key exchanges with a known good FFC scheme using Diffie-Helman group 14.
Findings: PASS – FTP_TRP.1/Admin, FTP_ITC.1, and FTP_ITC.1/VPN claim SSH, TLS, and IPsec. Each protocol uses Diffie-Hellman Group 14. Refer to test cases for FCS_SSHS_EXT.1.7 Test 2, FCS_TLSC_EXT.1.1 Test 1 (any DHE ciphersuite), FCS_TLSS_EXT.1.3 Test 2, and FCS_IPSEC_EXT.1.11. Those test cases use an independent, known-good interoperable cryptographic implementation.

3.2.3 FCS_CKM.4 Cryptographic Key Destruction

3.2.3.1 TSS

68 The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted

for[2]). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator checks that this is consistent with the operation of the TOE.

Findings:	[ST] Section 6.2.2 lists all relevant keys, key destruction situations and the destruction method used in each case. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE.

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

Findings:	[ST] Section 6.2.2 identifies how the keys stored as plaintext in non-volatile memory are destroyed. The description also includes identification and description of the interfaces that the TOE uses to destroy the keys.

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

Findings:	N/A. The ST does not claim 'destruction of reference' for volatile memory or 'invocation of an interface' for non-volatile memory.

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

Findings:	[ST] Table 23 identifies keys that are stored in a non-plaintext form. The TSS identifies the encryption method, the key-encrypting key used and where it's stored.

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

Findings:	[ST] Section 6.2. The TOE does not have any circumstances that may not conform to key destruction requirements.

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

Findings:	[ST] Section 6.2. The selection was not selected in the ST.

[2] Where keys are stored encrypted or wrapped under another key then this may need to be explained in order to allow the evaluator to confirm the consistency of the description of keys with the TOE functions.

3.2.3.2 Guidance Documentation

74 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.

75 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[3] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings:	There are no circumstances where delayed or prevented key destruction can occur. The “Key Zeroization” section in the [SUPP] describes the process for clearing CSPs and other sensitive information from the TOE when required.

3.2.3.3 Tests

76 None

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

3.2.4.1 TSS

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

Findings:	[ST] Table 22 in Section 6.2 identifies the key sizes and modes supported by the TOE for data encryption/decryption.

3.2.4.2 Guidance Documentation

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

Findings:	These parameters are non-configurable and supported by default.

3.2.4.3 Tests

AES-CBC Known Answer Tests

79 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

[3] 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).

evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

80 **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.

81 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.

82 **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.

83 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.

84 **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].

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

86 **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].

87 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

88 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.

89 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

90 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]
```

91 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.

92 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

93 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.

94 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.

95 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.

96 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

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

98 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.

99 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.

100 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.

101 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].

102 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

103 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

104 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]

105 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.

106 There is no need to test the decryption engine.

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2240, A2241, and A2242 for AES. This is described in [ST] Table 24.

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

3.2.5.1 TSS

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

Findings:	[ST] Table 22 in Section 6.2 specifies the cryptographic algorithms and key sizes supported by the TOE for signature services.

3.2.5.2 Guidance Documentation

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

Findings:	Section ‘Certificates’ in [ADMIN], starting from page 1012 describes how to configure the key sizes and key types (RSA and ECDSA) when generating CSRs.

3.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

109 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

110 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

111 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.

112 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

113 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.

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

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2240, A2241, and A2242 for RSA and ECDSA signature generation and verification. This is described in [ST] Table 24.

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

3.2.6.1 TSS

115 The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Findings:	[ST] Section 6.2.1 documents the association of the hash function with other TSF cryptographic functions.

3.2.6.2 Guidance Documentation

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

Findings:	[SUPP] section "Configuration and use of approved cryptographic algorithms" indicates HTTPS/TLS and SSH cryptographic algorithms are not configurable, and use of approved algorithms is enforced by the FIPS-CC mode of operation. The HMAC/Hashes that can be used in IPsec are specified and include the block and output sizes.

3.2.6.3 Tests

117 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.

118 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

119 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

120 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

121 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

122 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

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

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2225, and A2291 for Hashing. This is described in [ST] Table 24.

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

3.2.7.1 TSS

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

Findings:	[ST] Table 22 in Section 6.2 specifies the key length, hash function used, block size and output MAC length used by the HMAC function.

3.2.7.2 Guidance Documentation

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

Findings:	[SUPP] section “Configuration and use of approved cryptographic algorithms” indicates HTTPS/TLS and SSH cryptographic algorithms are not configurable, and use of approved algorithms is enforced by the FIPS-CC mode of operation. The HMAC/Hashes that can be used in IPsec are specified and include the block and output sizes.

3.2.7.3 Tests

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

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2225, and A2291 for HMAC. This is described in [ST] Table 24.

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

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

3.2.8.1 TSS

128 The evaluator shall examine the TSS to determine that it specifies the DRBG type, identifies the entropy source(s) seeding the DRBG, and state the assumed or calculated min-entropy supplied either separately by each source or the min-entropy contained in the combined seed value.

Findings:	[ST] Section 6.2.3 specifies the DRBG type, the entropy source and states the calculated min-entropy contained in the combined seed value.

3.2.8.2 Guidance Documentation

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

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

3.2.8.3 Tests

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

131 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).

132 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.

133 The following paragraphs contain more information on some of the input values to be generated/selected by the evaluator.

Entropy input: the length of the entropy input value must equal the seed length.

Nonce: If a nonce is supported (CTR_DRBG with no Derivation Function does not use a nonce), the nonce bit length is one-half the seed length.

Personalization string: The length of the personalization string must be <= seed length. If the implementation only supports one personalization string length, then the same length can be used for both values. If more than one string length is support, the evaluator shall use personalization strings of two different lengths. If the implementation does not use a personalization string, no value needs to be supplied.

Additional input: the additional input bit lengths have the same defaults and restrictions as the personalization string lengths.

Findings:	The vendor uses the CAVP certificates A2225 and A2291 for DRBG. This is described in [ST] Table 24.

3.3 Identification and Authentication (FIA)

3.3.1 FIA_AFL.1 Authentication Failure Management

3.3.1.1 TSS

134 The evaluator shall examine the TSS to determine that it contains a description, for each supported method for remote administrative actions, of how successive unsuccessful authentication attempts are detected and tracked. The TSS shall also describe the method by which the remote administrator is prevented from successfully logging on to the TOE, and the actions necessary to restore this ability.

Findings:	[ST] Section 6.7 identifies SSH or the web GUI (TLS/HTTPS) as the methods for remote administrative actions. The TSS describes the TOE's behavior in case of unsuccessful authentication attempts. The TSS states that the remote administrator is prevented from successfully logging on when the failed remote authentication attempt limit is met. The account is locked for a configured a time period. And access is restored at the end of the time period.

135 The evaluator shall examine the TSS to confirm that the TOE ensures that authentication failures by remote administrators cannot lead to a situation where no administrator access is available, either permanently or temporarily (e.g. by providing local logon which is not subject to blocking).

Findings:	[ST] Section 6.7 states that the local console does not implement any lockout mechanism to ensure authentication failures by remote administrators cannot lead to a situation where no administrator access is available.

3.3.1.2 Guidance Documentation

136 The evaluator shall examine the guidance documentation to ensure that instructions for configuring the number of successive unsuccessful authentication attempts and time period (if implemented) are provided, and that the process of allowing the remote administrator to once again successfully log on is described for each "action" specified (if that option is chosen). If different actions or mechanisms are implemented depending on the secure protocol employed (e.g., TLS vs. SSH), all must be described.

Findings:	[ADMIN] section "Configuring the maximum log in attempts and lockout period" shows the commands to use to set the maximum attempts allowed and the lockout period. The [CLI] under "config user setting" (page 1182) shows the limits for each option.

137 The evaluator shall examine the guidance documentation to confirm that it describes, and identifies the importance of, any actions that are required in order to ensure that administrator access will always be maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

Findings:	The account lockout does not affect the local console by default, no additional actions are needed to ensure administrator access will always be maintained.

3.3.1.3 Tests

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

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

High-Level Test Description
Configure the remote login failure threshold and duration until unlocking locked accounts.

High-Level Test Description
For the Web UI and remote CLI, attempt to login using an incorrect password until the login failure threshold has been met. Attempt to login using the correct password and verify the login attempt fails.
Findings: PASS – The evaluator confirmed that the TOE blocks login attempts to a user after the configured threshold of invalid login attempts is met at both the Web UI and the remote CLI.

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.

High-Level Test Description
Administrator action selection is not included in the ST. For the Web UI and remote CLI, attempt to login using an incorrect password until the login failure threshold has been met. Shortly before the lockout duration has expired, attempt to login using the correct password and verify the login attempt fails. After the lockout duration has expired, attempt to login using the correct password and verify the login attempt succeeds.
Findings: PASS – The evaluator confirmed that, using correct credentials, access was denied prior to the lockout duration expiring and access was granted after the lockout duration had expired.

3.3.2 FIA_PMG_EXT.1 Password Management

3.3.2.1 TSS

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

Findings:	[ST] Section 6.7 contains the list of all the supported special characters and states that passwords must be 8-64 characters. The evaluator confirmed the list of special characters matches FIA_PMG_EXT.1.1.

3.3.2.2 Guidance Documentation

140 The evaluator shall examine the guidance documentation to determine that it:

a) identifies the characters that may be used in passwords and provides guidance to Security Administrators on the composition of strong passwords, and

b) provides instructions on setting the minimum password length and describes the valid minimum password lengths supported.

Findings:	The default complexity is stated in [SUPP] in section “The FIPS-CC Mode of Operation”. The [ADMIN] document provides guidance under “Password policy” starting page 861 to change the secure options for passwords and CLI commands. The minimum length of 8 characters is enforced by the TOE and described in [SUPP].

3.3.2.3 Tests

141 The evaluator shall perform the following tests.

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

High-Level Test Description
Set the minimum password length to 8 characters. Set the password for an account and verify authentication succeeds using the configured password using passwords that are 8 characters long and passwords that contain all of the claimed characters. Verify the passwords can be successfully set and used to login.
Findings: PASS – The evaluator confirmed that 8 character passwords and passwords consisting of all claimed characters could be successfully set and used to login.

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

High-Level Test Description
Using each management interface, attempt to set a password whose length is one less than the configured password policy. Verify the password change is rejected.
Findings: PASS – The evaluator confirmed that the TOE did not allow the user to set passwords that did not meet the configured minimum length.

3.3.3 FIA_UIA_EXT.1 User Identification and Authentication

3.3.3.1 TSS

142 The evaluator shall examine the TSS to determine that it describes the logon process for each logon method (local, remote (HTTPS, SSH, etc.)) supported for the product. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a “successful logon”.

Findings:	[ST] Section 6.7 describes the logon process for SSH and the web GUI (TLS/HTTPS). The description contains information regarding the password authentication and SSH public-key based authentication. The TSS describes what constitutes a successful logon for each credential type.

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

Findings:	[ST] Section 6.9 states the TOE does not permit any actions and only the warning banner is displayed before user authentication for local and remote administration.

144 For distributed TOEs the evaluator shall examine that the TSS details how Security Administrators are authenticated and identified by all TOE components. If not all TOE components support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the TSS shall describe how the overall TOE functionality is split between TOE components including how it is ensured that no unauthorized access to any TOE component can occur.

Findings:	N/A. The TOE is not a distributed TOE.

145 For distributed TOEs, the evaluator shall examine the TSS to determine that it describes for each TOE component which actions are allowed before user identification and authentication. The description shall cover authentication and identification for local and remote TOE administration. For each TOE component that does not support authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2 the TSS shall describe any unauthenticated services/services that are supported by the component.

Findings:	N/A. The TOE is not a distributed TOE.

3.3.3.2 Guidance Documentation

146 The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps (e.g., establishing credential material such as pre-shared keys, tunnels, certificates, etc.) to logging in are described. For each supported the login method, the evaluator shall ensure the guidance documentation provides clear instructions for successfully logging on. If configuration is necessary to ensure the services provided before login are limited, the evaluator shall determine that the guidance documentation provides sufficient instruction on limiting the allowed services.

Findings:	Under “config system admin” starting page 869 in the [CLI] document, instructions can be found to configure a public key for SSH authentication. Instructions for successfully logging on for CLI or GUI are in the [ADMIN] document under “Getting Started” in “Using the GUI” (page 19) and “Using the CLI” (page 24).

3.3.3.3 Tests

147 The evaluator shall perform the following tests for each method by which administrators access the TOE (local and remote), as well as for each type of credential supported by the login method:

a) Test 1: The evaluator shall use the guidance documentation to configure the appropriate credential supported for the login method. For that credential/login method, the evaluator shall show that providing correct I&A information results in the ability to access the system, while providing incorrect information results in denial of access.

High-Level Test Description
For each management interface and credential type: • Log into using a known-good credential and verify login succeeds.

High-Level Test Description
• Log into using a known-bad credential and verify login fails.
Findings: PASS – The evaluator confirmed that the TOE permits logins when valid credentials are used and denies logins when invalid credentials are used.

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

High-Level Test Description
Examine the login pages to determine if any services are available prior to authentication. Attempt to browse directly to pages/services and verify access is denied. Verify the user is unable to run any commands or services other than the warning banner.
Findings: PASS – The evaluator confirmed that viewing the warning banner is the only service available to remote entities prior to authentication.

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

High-Level Test Description
At the Local Console, enter common shell key combinations and strings to escape and/or run commands. Verify the user is unable to run any commands or services other than the warning banner.
Findings: PASS – The evaluator confirmed that viewing the warning banner is the only service available at the local console prior to authentication.

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.

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

3.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

148 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.

3.3.5 FIA_UAU.7 Protected Authentication Feedback

3.3.5.1 TSS

149 None

3.3.5.2 Guidance Documentation

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

Findings:	[ST] Section 6.7 states, "The TOE provides no feedback while authentication is in progress at the console," so no configuration is necessary in guidance to ensure authentication data is not revealed.

3.3.5.3 Tests

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

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

High-Level Test Description
Authenticate to the TOE at the local console. Verify that at most obscured feedback is provided while entering the authentication information.
Findings: PASS – The evaluator confirmed that no feedback is provided while entering authentication information.

3.4 Security management (FMT)

3.4.1 General requirements for distributed TOEs

3.4.1.1 TSS

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

Findings:	N/A. The TOE is not a distributed TOE.

3.4.1.2 Guidance Documentation

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

Findings:	N/A. The TOE is not a distributed TOE.

3.4.1.3 Tests

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

High-Level Test Description
The TOE is not a distributed TOE.

High-Level Test Description
Findings: N/A

3.4.2 FMT_MOF.1/ManualUpdate

3.4.2.1 TSS

155 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.1. There are no specific requirements for non-distributed TOEs.

Findings:	N/A. The TOE is not a distributed TOE.

3.4.2.2 Guidance Documentation

156 The evaluator shall examine the guidance documentation to determine that any necessary steps to perform manual update are described. The guidance documentation shall also provide warnings regarding functions that may cease to operate during the update (if applicable).

Findings:	The [ADMIN] document provides instructions to perform a manual update in “System > Firmware” starting page 866. The [SUPP] document also provides instructions regarding the CC firmware in “Installing the CC Certified Firmware” starting page 7.

157 For distributed TOEs the guidance documentation shall describe all steps how to update all TOE components. This shall contain description of the order in which components need to be updated if the order is relevant to the update process. The guidance documentation shall also provide warnings regarding functions of TOE components and the overall TOE that may cease to operate during the update (if applicable).

Findings:	N/A. The TOE is not a distributed TOE.

3.4.2.3 Tests

158 The evaluator shall try to perform the update using a legitimate update image without prior authentication as Security Administrator (either by authentication as a user with no administrator privileges or without user authentication at all – depending on the configuration of the TOE). The attempt to update the TOE shall fail.

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

High-Level Test Description
Verify updates cannot be installed without user authentication. Verify the security administrator can install updates.
Findings: PASS – While testing FIA_UIA_EXT.1.1, the evaluator confirmed that no administrator actions were possible prior to authentication. While testing FPT_TUD_EXT.1 Test 1, the evaluator confirmed that Security Administrator is able to install legitimate updates.

3.4.3 FMT_MTD.1/CoreData Management of TSF Data

3.4.3.1 TSS

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

Findings:	[ST] Section 6.9 states there are no administrative functions accessible through an interface prior to login and that the management functions are restricted to the Security Administrator.

161 If TOE supports handling of X.509v3 certificates and implements a trust store, the evaluator shall examine the TSS to determine that it contains sufficient information to describe how the ability to manage the TOE's trust store is restricted.

Findings:	[ST] Section 6.9 indicates management of the trust store is restricted to the Security Administrator. The Security Administrator can manage the TOE's trust store by generating and deleting cryptographic keys associated with CSRs and importing X.509v3 certificates.

3.4.3.2 Guidance Documentation

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

Findings:	The [CLI], [ADMIN] and [SUPP] list all the functions that can be used to manipulate TSF data. For specific references, please refer to the SFR AA of interest. For example, firewall policy TSF data manipulating functions are described in FFW_RUL_EXT.*.

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

Findings:	The [ADMIN] document in "System > Certificates" (starting on page 1012) provides information for the administrator to configure and maintain the trust store in a secure way. Additional information can be found in the [SUPP] in the section "VPN and Certificate Specific Settings" starting page 22.

3.4.3.3 Tests

164 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.

3.4.4 FMT_SMF.1 Specification of Management Functions

165 The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1/ManualUpdate, FMT_MOF.1/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1/Services, and FMT_MOF.1/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

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

166 The evaluator shall examine the TSS, Guidance Documentation and the TOE as observed during all other testing and shall confirm that the management functions specified in FMT_SMF.1 are provided by the TOE. The evaluator shall confirm that the TSS details which security management functions are available through which interface(s) (local administration interface, remote administration interface).

Findings:	[ST] Section 6.9 details which security management functions are available through remote and local administration.

167 The evaluator shall examine the TSS and Guidance Documentation to verify they both describe the local administrative interface. The evaluator shall ensure the Guidance Documentation includes appropriate warnings for the administrator to ensure the interface is local.

Findings:	[ST] Section 6.9 and the Guidance Documentation describe the local interface.

168 For distributed TOEs with the option 'ability to configure the interaction between TOE components' the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

Findings:	N/A. The TOE is not a distributed TOE.

3.4.4.2 Guidance Documentation

169 See [NDcPP-SD] section 2.4.4.1.

3.4.4.3 Tests

170 The evaluator tests management functions as part of testing the SFRs identified in [NDcPP-SD] 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.

3.4.5 FMT_SMR.2 Restrictions on security roles

3.4.5.1 TSS

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

Findings:	[ST] Section 6.9 states that the TOE supports a single role, Security Administrator.

3.4.5.2 Guidance Documentation

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

Findings:	The [ADMIN] document describes the methods to administer the TOE locally (Section "Using the CLI") and remotely (Sections "Using the CLI" and "Using the GUI") in "Getting started" starting page 19. Those sections contain instructions for configuring different options for each method.

3.4.5.3 Tests

173 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.

High-Level Test Description
Verify that all supported administrative interfaces are exercised during the evaluation.
Findings: PASS – All interfaces are tested in the course of performing other tests. For example, FMT_MOF.1/ManualUpdate tests the Remote CLI and Web UI and FIA_AFL.1 tests the Local Console.

3.5 Protection of the TSF (FPT)

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

3.5.1.1 TSS

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

Findings:	[ST] Section 6.10 describes the keys storage and how they are unable to be viewed through an interface.

3.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

3.5.2.1 TSS

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

Findings:	[ST] Section 6.10 indicates passwords are stored encrypted using AES-128. It is not possible to view the plaintext passwords, because only the encrypted passwords can be viewed.

3.5.3 FPT_TST_EXT.1 TSF testing

3.5.3.1 TSS

176 The evaluator shall examine the TSS to ensure that it details the self-tests that are run by the TSF; this description should include an outline of what the tests are actually doing (e.g., rather than saying "memory is tested", a description similar to "memory is tested by writing a value to each memory location and reading it back to ensure it is identical to what was written" shall be used). The evaluator shall ensure that the TSS makes an argument that the tests are sufficient to demonstrate that the TSF is operating correctly.

Findings:	[ST] Section 6.10 describes each self-test run by the TSF and what they are doing. The TSS also provides an argument that the tests are sufficient to demonstrate that the TSF is operating at its intended level of capability.

177 For distributed TOEs the evaluator shall examine the TSS to ensure that it details which TOE component performs which self-tests and when these self-tests are run.

Findings:	N/A. The TOE is not a distributed TOE.

3.5.3.2 Guidance Documentation

178 The evaluator shall also ensure that the guidance documentation describes the possible errors that may result from such tests, and actions the administrator should take in response; these possible errors shall correspond to those described in the TSS.

Findings:	The [SUPP] document describes the FIPS Error Mode (page 18) which can occur and the actions the administrator should take in response. This is consistent with the TSS.

179 For distributed TOEs the evaluator shall ensure that the guidance documentation describes how to determine from an error message returned which TOE component has failed the self-test.

Findings:	The TOE is not a distributed TOE.

3.5.3.3 Tests

180 It is expected that at least the following tests are performed:

a) Verification of the integrity of the firmware and executable software of the TOE

b) Verification of the correct operation of the cryptographic functions necessary to fulfil any of the SFRs.

181 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.

182 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.

High-Level Test Description
Restart the TOE and verify that the startup includes an indicator that self-tests were executed and passed permitting the device to operate.
Findings: PASS – The evaluator confirmed that the startup output indicates the claimed self-tests were performed.

183 For distributed TOEs the evaluator shall perform testing of self-tests on all TOE components according to the description in the TSS about which self-test are performed by which component.

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

3.5.4 FPT_TUD_EXT.1 Trusted Update

3.5.4.1 TSS

184 The evaluator shall verify that the TSS describe how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the TSS needs to describe how and when the inactive version becomes active. The evaluator shall verify this description.

Findings:	[ST] Section 6.10 describes how to query the currently active version. The TOE does not support delayed activation.

185 The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system firmware and software (for simplicity the term 'software' will be used in the following although the requirements apply to firmware and software). The evaluator shall verify that the description includes a digital signature verification of the software before installation and that installation fails if the verification fails. Alternatively an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

Findings:	[ST] Section 6.10 describes in all TSF software update mechanisms for updating the system software. The description includes a digital signature verification of the uploaded software before installation. The description also states that if the verification is successful, the upgrade proceeds and if the verification fails, the upgrade will fail and an audit record will be generated.

186 If the options ‘support automatic checking for updates’ or ‘support automatic updates’ are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that

the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

Findings:	N/A. The ST does not include any of those selections.

187 For distributed TOEs, the evaluator shall examine the TSS to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component. Alternatively, this description can be provided in the guidance documentation. In that case the evaluator should examine the guidance documentation instead.

Findings:	N/A. The TOE is not a distributed TOE.

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

Findings:	N/A. The TOE does not use a published hash.

3.5.4.2 Guidance Documentation

189 The evaluator shall verify that the guidance documentation describes how to query the currently active version. If a trusted update can be installed on the TOE with a delayed activation, the guidance documentation needs to describe how to query the loaded but inactive version.

Findings:	[SUPP] under “Installing the CC Certified Firmware, the section “Installing the FIPS-CC firmware build” describes the recommended way of determining the current active version of the firmware. Delayed activation is not supported by the TOE.

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

Findings:	The [SUPP] document describes the process for validating the firmware’s integrity in section “Verifying the integrity of the firmware build”. The [ADMIN] states in section “System > Firmware > Testing a firmware version” (page 869) that “Firmware images are signed, and the signature is attached to the code as it is built and verification is done during the upgrade. The description is consistent with the TSS.

191 If a published hash is used to protect the trusted update mechanism, the evaluator shall verify that the guidance documentation describes how the Security Administrator can obtain authentic published hash values for the updates.

Findings:	Published hashes are not supported.

192 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 user; it does not need to give information about the internal communication that takes place when applying updates.

Findings:	The TOE is not a distributed TOE

193 If this was information was not provided in the TSS: For distributed TOEs, the evaluator shall examine the Guidance Documentation to ensure that it describes how all TOE components are updated, that it describes all mechanisms that support continuous proper functioning of the TOE during update (when applying updates separately to individual TOE components) and how verification of the signature or checksum is performed for each TOE component.

Findings:	The TOE is not a distributed TOE

194 If this was information was not provided in the TSS: If the ST author indicates that a certificate-based mechanism is used for software update digital signature verification, the evaluator shall verify that the Guidance Documentation contains a description of how the certificates are contained on the device. The evaluator also ensures that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

Findings:	Certificate-based update authentication is not supported

3.5.4.3 Tests

195 The evaluator shall perform the following tests:

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

High-Level Test Description
Get the current version of the TOE. Attempt to install a valid update using the CLI and Web UI. Verify the installation succeeds and the version is updated.
Findings: PASS – The evaluator confirmed the TOE displayed its current version, successfully installed a valid update, and displayed the updated version.

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed

(otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

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

2) An image that has not been signed

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

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

High-Level Test Description
Get the current version of the TOE. Attempt to install updates with a bad signature (modified update), without a signature, with an untrusted signature (untrusted key) using the CLI and Web UI. Verify the installation fails and the version has not changed.
Findings: PASS – The evaluator confirmed that the TOE did not install illegitimate updates and the version did not change.

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 user to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

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

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

196 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.

High-Level Test Description
The TOE does not use a published hash to verify updates.
Findings: N/A

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

Note	The TOE only supports manual updates. Manual updates are tested above.

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

Test Not Applicable	The TOE is not a distributed TOE.

3.5.5 FPT_STM_EXT.1 Reliable Time Stamps

NIAP TD0632

3.5.5.1 TSS

199 The evaluator shall examine the TSS to ensure that it lists each security function that makes use of time, and that it provides a description of how the time is maintained and considered reliable in the context of each of the time related functions.

Findings:	[ST] Section 6.10 lists each security function that makes used of time and states the time is set by the security administrator. The TSS considers the time source reliable, because: **-** For physical models: “The TOE maintains its own time source, which is free from outside interference. The physical form factors have an internal battery-backed hardware clock for reliability.” **-** For the virtual model: “The virtual form factors rely on an internal hardware clock on the virtualization host system.” A.VS_CORRECT_CONFIGURATION indicates the VS is assumed to support ND functionality.

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

Findings:	N/A. The TOE does not obtain time from the underlying virtualization system.

3.5.5.2 Guidance Documentation

201 The evaluator examines the guidance documentation to ensure it instructs the administrator how to set the time. If the TOE supports the use of an NTP server, the guidance documentation instructs how a communication path is established between the TOE and the NTP server, and any configuration of the NTP client on the TOE to support this communication.

Findings:	Instructions to set the time are found in the [ADMIN] document in the section “Firmware > Settings > Setting the system time” page 876. NTP is not claimed and must be disabled to be compliant with the evaluated configuration as stated in the [SUPP] document in the section “Disable NTP” page 18.

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

Findings:	N/A. The TOE does not obtain time from the underlying virtualization system.

3.5.5.3 Tests

203 The evaluator shall perform the following tests:

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

High-Level Test Description
Change the time to times in the past and future using the CLI and Web UI.
Findings: PASS – The evaluator confirmed the Security Administrator was able to change the time using the CLI and Web UI.

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

High-Level Test Description
The TOE does not support NTP.
Findings: N/A

NIAP TD0632

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.

High-Level Test Description
The TOE does not obtain time from the underlying VS.
Findings: N/A

204 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.

High-Level Test Description
The TOE does not support independent time information.
Findings: N/A

3.6 TOE Access (FTA)

3.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.6.1.1 TSS

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

Findings:	[ST] Section 6.11 states the TOE supports local session termination based on a time period set by the security administrator.

3.6.1.2 Guidance Documentation

206 The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Findings:	The instructions to set the idle timeout for session termination are described in [ADMIN] in “System > Settings > Setting the idle timeout time” (page 880) and [CLI] in “CLI configuration commands > system > config system global" (page 766). The “admintimeout” parameter (applies to all administrative interfaces). [CLI] page767 explains the "admin-console-timeout” parameter which is a local console specific setting that overrides the global “admintimeout” parameter.

3.6.1.3 Tests

207 The evaluator shall perform the following test:

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

High-Level Test Description
Configure the global and local console idle timeout settings to several different values while establishing Local Console sessions. Verify the Local Console session is terminated when the threshold is reached.
Findings: PASS – The evaluator confirmed the TOE terminates local console sessions when the inactivity timeout period is reached.

3.6.2 FTA_SSL.3 TSF-initiated Termination

3.6.2.1 TSS

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

Findings:	[ST] Section 6.11 states the TOE supports remote session termination based on a time period set by the security administrator.

3.6.2.2 Guidance Documentation

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

Findings:	The instructions to set the idle timeout for session termination is described in [ADMIN] in “System > Settings > Setting the idle timeout time” (page 880) and [CLI] in “CLI configuration commands > system > config system global" (page 766) “admintimeout” parameter (applies to all administrative interfaces).

3.6.2.3 Tests

210 For each method of remote administration, the evaluator shall perform the following test:

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

High-Level Test Description
Configure the inactivity timeout to several values. Login to each remote interface and verify the TOE terminates the session when the inactivity timer has expired.
Findings: PASS – The evaluator confirmed that the TOE terminates remote sessions (both CLI and Web UI) when the inactivity timeout period is reached.

3.6.3 FTA_SSL.4 User-initiated Termination

3.6.3.1 TSS

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

Findings:	[ST] Section 6.11 states that remote and local administrators can manually terminate their sessions. Web UI sessions are terminated using “Logout” and CLI sessions are terminated using the “exit” command.

3.6.3.2 Guidance Documentation

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

Findings:	The instructions to terminate a local or remote interactive session are found in the [SUPP] in section “Administration > Logging out from the GUI and CLI”.

3.6.3.3 Tests

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

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

High-Level Test Description
Log into the Local Console Log out of the Local Console session.
Findings: PASS – The evaluated confirmed that the local console session is terminated when the administrator logs out.

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

High-Level Test Description
Log into the SSH and Web UI interfaces. Log out of each session.
Findings: PASS – The evaluated confirmed that the remote administrative sessions at the Web UI and remote CLI are terminated when the administrator logs out.

3.6.4 FTA_TAB.1 Default TOE Access Banners

3.6.4.1 TSS

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

Findings:	[ST] Section 6.11 identifies the methods of administration as the local console, web GUI, and SSH. The TSS states that a warning and consent banner is presented on all methods of access prior to authentication.

3.6.4.2 Guidance Documentation

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

Findings:	The instructions to enable/disable the banner message are in the [SUPP] document in the section “Administration > Admin access disclaimer”. [CLI] section “CLI configuration commands > system > config system replacemsg admin” describes how to use the “buffer” parameter to configure the banner message.

3.6.4.3 Tests

216 The evaluator shall also perform the following test:

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

High-Level Test Description
Configure a notice and consent warning message. Verify the notice and consent warning message is displayed prior to establishing an administrative session at the Web UI, Remote CLI, and Local Console.
Findings: PASS – The evaluator confirmed that the administrator is able to configure the warning message and that the warning message is displayed prior to authentication at each administrative interface.

3.7 Trusted path/channels (FTP)

3.7.1 FTP_ITC.1 Inter-TSF trusted channel

3.7.1.1 TSS

217 The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

Findings:	[ST] Section 6.12 describes all communications with authorized IT entities in the requirement and the protocols used. All secure communications mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

3.7.1.2 Guidance Documentation

218 The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

Findings:	For the logging server, the [SUPP] document describes how to set up the TOE to communicate with the FortiAnalyzer (FAZ) in the “FortiAnalyzer configuration” section. If the connection to the FAZ is unintentionally broken, it can be rescued by following the instructions given in the section “Reconnecting to FortiAnalyzer”. IPSec VPN connections can be configured as per the [ADMIN] document in the section “VPN” > “IPSec VPNs” (starting on page 1424). The troubleshooting instructions are provided under [ADMIN] section “VPN IPsec troubleshooting” (starting from 1703). The [SUPP] document in section “VPN and Certificate Specific Settings > Phase 1/Phase2 encryption strength” also provides instructions to ensure that IPSec Phase 2 encryption strength should not exceed the IKE Phase 1 encryption strength.

3.7.1.3 Tests

219 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.

220 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.

High-Level Test Description
Ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.
Findings: PASS – The TOE maintains a trusted channel to the remote audit server, which is set up as per the evaluated configuration. It is constantly tested throughout the evaluation. The trusted channel is specifically tested as part of FCS_TLSC_EXT.1. The TOE supports an IPsec trussed channel with "authorized IT entities supporting VPN communications. The trusted channel is tested as part of FCS_IPSEC_EXT.1.

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

High-Level Test Description
Ensure the trusted channel can be initiate form the TOE.
Findings: PASS – FCS_TLSC_EXT.1 testing shows the TOE can initiate the trusted channel to the remote audit server. FCS_IPSEC_EXT.1.3 Test 1 Step 1 shows the TOE can initiate the trusted channel.

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

High-Level Test Description
Ensure the trusted channel data is not sent in plaintext.
Findings: PASS – FCS_TLSC_EXT.1 testing shows the TOE successfully establishing a trusted channel with the remote audit server. The remote audit server is a known good TLS server implementation, so the successful transfer of Application Data shows the channel data is not sent in plaintext (i.e., the server would terminate the connection due to decryption and/or integrity errors if the data was sent in plaintext). FCS_IPSEC_EXT.1.5 Test 2 Step 2 shows that trusted 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.

High-Level Test Description
Physically disrupt the connection with the remote IT entity. Verify the communications are not sent in plaintext while the connection is disrupted or when it is restored.
Findings: PASS – The evaluator confirmed that the TOE did not send trusted channel data (TLS or IPsec) in plaintext when the channel was disrupted for the network layer or application layer timeout durations.

221 Further assurance activities are associated with the specific protocols.

222 For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of external secure channels to TOE components in the Security Target.

Findings:	This is not a distributed TOE.

223 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.

Note	The developer provided sufficient information regarding application layer timeout settings for the evaluator to perform FTP_ITC.1 Test 4.

3.7.2 FTP_TRP.1/Admin Trusted Path

3.7.2.1 TSS

224 The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Findings:	[ST] Section 6.12 states that the TOE provides remote administration methods, and those communications are protected. The TSS lists HTTPS for the web GUI and SSH for remote CLI. The evaluator confirmed these claims are consistent with the selections in FTP_TRP.1.1/Admin.

3.7.2.2 Guidance Documentation

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

Findings:	Instructions for establishing the remote administrative sessions for each supported method can be found in the [ADMIN] guide under “Getting Started” in sections “Using the GUI” (page 19) and “Using the CLI” (page 24) respectively. The [SUPP] also describes the cryptographic parameters of the web GUI TLS channel in section “Web browser requirements” starting on page 17.

3.7.2.3 Tests

226 The evaluator shall perform the following tests:

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

High-Level Test Description
Ensure that communications using each remote administration method is tested during the course of the evaluation.
Findings: PASS – The trusted paths are the TLS/HTTPS Web UI and SSH Remote CLI, which both are set up as per the evaluated configuration. They are constantly tested throughout the evaluation. TLS is tested in FCS_TLSS_EXT.1, and SSH is tested in FCS_SSHS_EXT.1.

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

High-Level Test Description
Ensure that the trusted channel data is not sent in plaintext.
Findings: PASS – FCS_TLSS_EXT.1 and FCS_SSHS_EXT.1 testing shows the TOE successfully establishing trusted paths. The remote trusted path client is a known good TLS or SSH client implementation, so the successful transfer of channel data shows the channel data is not sent in plaintext (i.e., the client would terminate the connection due to decryption and/or integrity errors if the data was sent in plaintext).

227 Further assurance activities are associated with the specific protocols.

228 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.

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

4 Evaluation Activities for Optional Requirements

4.1 Cryptographic Support (FCS)

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

4.1.1.1 TSS

FCS_TLSC_EXT.2.1

229 The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Findings:	[ST] Section 6.3.3 describes the use of a client side certificate for TLS mutual authentication.

4.1.1.2 Guidance Documentation

FCS_TLSC_EXT.2.1

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

Findings:	The client certificate is set using the “certificate” option in the “config log fortianalyzer setting” configuration tree. The process for generating or loading this certificate can be found in the [ADMIN] section “System > Certificates” (starting on page 1012).

4.1.1.3 Tests

NIAP TD0670

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

FCS_TLSC_EXT.2.1

232 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.

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

High-Level Test Description
Have the TOE connected to a TLS server that requests mutual authentication. Verify the TOE sends Certificate and Certificate Verify messages.
Findings: PASS – The evaluator confirmed the TOE sends Certificate and Certificate verify messages when it connects to a server that sends a Certificate Request message.

5 Evaluation Activities for Selection-Based Requirements

5.1 Cryptographic Support (FCS)

5.1.1 FCS_HTTPS_EXT.1 HTTPS Protocol

5.1.1.1 TSS

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

Findings:	[ST] Section 6.3.1 provides enough details to explain the implementation of HTTPS complies with RFC 2818.

5.1.1.2 Guidance Documentation

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

Findings:	Use of HTTP/TLS for the remote web GUI is described in the [ADMIN] document section “Getting started” page 19. The [SUPP] also describes the cryptographic parameters of the web GUI TLS channel in section “Web browser requirements”.

5.1.1.3 Tests

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

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

238 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.

5.1.2 FCS_IPSEC_EXT.1 IPsec Protocol

5.1.2.1 TSS

FCS_IPSEC_EXT.1.1

239 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.

240 As noted in section 4.4.1 of RFC 4301, the processing of entries in the SPD is non-trivial and the evaluator shall determine that the description in the TSS is sufficient to determine which rules will be applied given the rule structure implemented by the TOE. For example, if the TOE allows specification of ranges, conditional rules, etc., the evaluator shall determine that the description of rule processing (for both inbound and outbound packets) is sufficient to determine the action that will be applied,

especially in the case where two different rules may apply. This description shall cover both the initial packets (that is, no SA is established on the interface or for that particular packet) as well as packets that are part of an established SA.

Findings:	[ST] Section 6.5 describes how the TOE processes a packet, how the SPD and the rules for processing inbound and outbound packets are implemented. The TSS description of rule processing for inbound and outbound traffic covers both the initial packets and packets that are part of an established SA. The TSS identifies that BYPASS, DISCARD, and PROTECT actions can be assigned to the rules. The TSS states that the rules are processed in the order defined by the security administrator.

FCS_IPSEC_EXT.1.3

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

Findings:	[ST] Section 6.5 states that the VPN can be established to operate in transport mode or tunnel mode.

FCS_IPSEC_EXT.1.4

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

Findings:	[ST] Section 6.5 indicates the TOE implements AES-CBC-128, AES-CBC-256, AES-GCM-128 and AES-GCM-256 in conjunction with a HMAC-SHA-256 to provide encryption services for ESP. The evaluator verified these algorithms and key sizes are claimed in FCS_COP.1/DataEncryption and FCS_COP.1/KeyedHash.

FCS_IPSEC_EXT.1.5

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

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

Findings:	[ST] Section 6.5 states the TOE implements both IKEv1 and IKEv2. The TSS states, “The TOE does not use aggressive mode for IKEv1 Phase 1 exchanges and only main mode is permitted in the evaluated configuration.”

FCS_IPSEC_EXT.1.6

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

Findings:	[ST] Section 6.5 identifies AES-CBC-128 and AES-CBC-256 as the encryption algorithms used for IKEv1 and IKEv2. The TSS also states, “IKE Peer-to-peer authentication uses HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512.”

FCS_IPSEC_EXT.1.7

246 The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 1 SA lifetime and/or the IKEv2 SA lifetime. The evaluator

shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Findings:	[ST] Section 6.5 states the TOE allows for the IKEv1 Phase 1 SA lifetime and IKEv2 SA lifetime to be configured between 120 and 172800 seconds. The evaluator confirmed that lifetime type and ranges are consistent with FCS_IPSEC_EXT.1.7. The evaluator confirmed the IKE version is consistent with FCS_IPSEC_EXT.1.5.

FCS_IPSEC_EXT.1.8

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

Findings:	[ST] Section 6.5 states the TOE allows for the IKEv1 Phase 2 SA lifetime and the IKEv2 Child SA lifetime to be configured between 5KB and 4GB or 120 and 172800 seconds. The evaluator confirmed that lifetime type and ranges are consistent with FCS_IPSEC_EXT.1.8. The evaluator confirmed the IKE version is consistent with FCS_IPSEC_EXT.1.5.

FCS_IPSEC_EXT.1.9

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

Findings:	[ST] Section 6.5 states the TOE utilises CTR-DRBG with AES (as specified in FCS_RBG_EXT.1) to generate the exponents used in IKE key exchanges, having the possible lengths of 224, 256 or 384 bits, corresponding to each of the supported DH groups.

FCS_IPSEC_EXT.1.10

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

Findings:	[ST] Section 6.5 states the TOE generates nonces used in IKE using CTR-DRBG with AES for negotiated PRF hashes.

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

Findings:	[ST] Section 6.5 states, “The TOE utilizes CTR-DRBG with AES (as specified in FCS_RBG_EXT.1) to generate the exponents used in IKE key exchanges.” The TSS indicates the nonces sizes are:128 bits for SHA-1 and SHA-256; and 256 bits for SHA-384 and SHA-512. 128-bits is greater than or equal to half the output size of SHA-1 and SHA-256. 256-bits is greater than or equal to half the output size of SHA-384 and SHA-512.

FCS_IPSEC_EXT.1.11

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

Findings:	[ST] Section 6.5 states, "The TOE supports Diffie-Hellman groups 14, 19 and 20. The specific group to be used for any given IPsec connection is specified in the IPsec policy configuration."

FCS_IPSEC_EXT.1.12

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

Findings:	[ST] Section 6.5 states the strength of the algorithms allowed for the IKE and ESP exchanges is between 128 and 256 bits. The TSS also describes that the TOE checks the strength of the algorithm used for Phase 2 (IKEv1) or CHILD_SA (IKEv2) to be lesser or equal than the algorithm strength used for the IKE SA.

FCS_IPSEC_EXT.1.13

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

Findings:	[ST] Section 6.5 states the TOE permits RSA and ECDSA public keys to perform peer authentication. This is consistent with FCS_COP.1/SigGen which selects RSA and ECDSA.

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

Findings:	[ST] Section 6.7 states the TOE accepts text-based pre-shared keys that are between 6 and 128 characters in length and composed of any combination of upper and lower case letters, numbers, and special characters (as specified in FIA_PSK_EXT.1.2). The TOE also accepts bit-based pre-shared keys. The TSS also states that the TOE converts text-based pre-shared keys into an authentication value using SHA-1 or the PRF that is configured as the hash algorithm for the IKE exchange.

FCS_IPSEC_EXT.1.14

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

type is compared to the peer's presented certificate, including what field(s) are compared and which fields take precedence in the comparison.

Findings:	[ST] Section 6.5 states the TOE compares the reference identifier of the peer against the reference identifier stored in the associated certificate. If the two values are not a match, the TOE will not establish the connection. The TOE supports DN reference identifiers.

5.1.2.2 Guidance Documentation

FCS_IPSEC_EXT.1.1

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

Findings:	The instructions for the Administrator to construct entries into the SPD that specify a rule for processing a packet can be found in the [ADMIN] document in the section "VPN > IPsec VPNs > VPN security policies" starting on page 1445. [CLI] section "config firewall policy" also provides information to construct entries (starting from page 309). The instructions are sufficient to allow an administrator to set up the SPD in an unambiguous fashion, including how ordering of rules impacts the processing of an IP packet.

FCS_IPSEC_EXT.1.3

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

Findings:	[CLI] describes under "config vpn ipsec phase2-interface" (page 1330) how to configure tunnel mode and transport mode using the "encapsulation" parameter and indicates tunnel-mode is the default.

FCS_IPSEC_EXT.1.4

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

Findings:	The instructions on how to configure the TOE to use the algorithms selected are found in the [ADMIN] document in the section "VPN > IPsec VPNs > General IPsec VPN configuration > Phase 2 configuration > Encryption & Authentication" (pages 1441 & 1442) and in the [CLI] section "config vpn ipsec phase2-interface" (page 1324) using the "proposal" parameter. [SUPP] section "Configuration and use of approved cryptographic algorithms" describes the algorithms allowed for CC. The evaluator confirmed the list of algorithms in [SUPP] matches the selected algorithms in FCS_IPSEC_EXT.1.4 in the [ST].

FCS_IPSEC_EXT.1.5

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

Findings:	The instructions to configure the IKE version can be found in the [ADMIN] document in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 1 configuration > IKE Version” (1428). The [CLI] document also provides the commands to configure the IKE version in the “config vpn ipsec phase1-interface” section using the “ike-version” parameter (pages 1303-1304). The [CLI] document describes the configuration of NAT Traversal in the in the “config vpn ipsec phase1-interface” section using the “nattraversal” parameter (page 1320).

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

Findings:	[CLI] “config vpn ipsec phase1-interface” section for the “mode” parameter (page 1304) indicates main mode is the default.

FCS_IPSEC_EXT.1.6

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

Findings:	The [ADMIN] document describes the configuration of all selected algorithms in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 1 configuration > Phase 1 Proposal” (page 1429). The [CLI] document also provides commands to configure the selected algorithms in the “config vpn ipsec phase1-interface” section using the “proposal” parameter (page 1310-1312). [SUPP] section “Configuration and use of approved cryptographic algorithms” describes the algorithms allowed for CC. The evaluator confirmed the list of algorithms in [SUPP] matches the selected algorithms in FCS_IPSEC_EXT.1.6 in the [ST].

FCS_IPSEC_EXT.1.7

NIAP TD0633

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

Findings:	The TOE only claims time-based limits for IKEv1 phase 1 and IKEv2 SA. The [ADMIN] document describes the configuration for the SA lifetimes in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 1 configuration > Key Lifetime” (page 1430). The guidance states that lifetime range is between 120 and 172800 seconds (48 hours). The [CLI] document also provides commands to configure the SA lifetimes in the “config vpn ipsec phase1-interface” section using the “keylife” parameter (page 1304). The guidance does not include instructions about configuring a time value below the desired threshold.

FCS_IPSEC_EXT.1.8

NIAP TD0633

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

Findings:	The TOE claims both volume-based and time-based limits for IKEv1 phase 2 and IKEv2 Child SA. The [ADMIN] document provides instructions to change those limits in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 2 configuration > Key Lifetime” (page 1442). The [CLI] document describes the command to use to make those changes in the “config vpn ipsec phase2-interface” using parameters “keylife-type,” keylifeseconds,” and “keylifekbs” (page 1330). The evaluator confirmed the specified values are consistent with FCS_IPSEC_EXT.1.8 in the [ST]. The guidance does not include instructions about configuring a time value below the desired threshold.

FCS_IPSEC_EXT.1.11

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

Findings:	The [ADMIN] document describes how to configure all key agreement algorithms in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 1 configuration” section (page 1429) and the [CLI] section “config vpn ipsec phase1-interface” using the “dhgrp” parameter (page 1315). [SUPP] section “VPN specific certificate settings > Miscellaneous” describes the algorithms allowed for CC. The evaluator confirmed the list of algorithms in [SUPP] matches the selected algorithms in FCS_IPSEC_EXT.1.11 in the [ST].

FCS_IPSEC_EXT.1.13

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

Findings:	The [ADMIN] document provides instructions on how to set up the TOE to use certificates in the “VPN > IPsec VPNs > General IPsec VPN configuration > Pre-shared key vs digital certificates” (page 1436). The [CLI] also provides the command to use to select the certificate to use in the “config vpn ipsec phase1-interface” section using the “certificate” and “peer” parameters (pages 1304 & 1305).

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

Findings:	The [ADMIN] document provides instructions on how to configure pre-shared keys in the “VPN > IPsec VPNs > General IPsec VPN configuration > Pre-shared key vs digital certificates” (pages 1435 & 1436). The [CLI] document also provides the command to configure pre-shared keys in the “config vpn ipsec phase1-interface” section using the “psksecret” parameter (page 1313). The TOE only uses pre-generated pre-shared keys.

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

Findings:	The TOE does not connect to an external CA for any PKI operations except for automatically refreshing CRLs. CAs are configured manually by the Administrator by following instructions found in the [ADMIN] document in the “System > Certificates” section (page 1012) or [CLI] using the “config vpn certificate crl” command (page 1220).

FCS_IPSEC_EXT.1.14

268 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not, and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE does not guarantee unique identifiers, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings:	The instructions to configure the peer’s reference identifier are found in the [ADMIN] document in the “VPN > IPsec VPNs > General IPsec VPN configuration > Phase 1 configuration” section using the Peer Options parameter (page 1428). [ADMIN] “VPN > IPsec VPNs > Site-to-site VPN > Site-to-site VPN with digital certificate” section (pages 1455-1461) and [CLI] “config user peer” section (page 1183-1184) describe how to configure the peer subject (DN). The [SUPP] document section “VPN specific certificate settings > Miscellaneous” states that SANs are not supported in IPsec VPN peer authentication.

5.1.2.3 Tests

FCS_IPSEC_EXT.1.1

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

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

High-Level Test Description
Create three rules will bypass the VPN, encrypt packets in the VPN, or be dropped. Send packets matching each rule and verify the appropriate action is taken.

High-Level Test Description
Findings: PASS – The evaluator confirmed that TOE correctly forwards packets unencrypted, tunnels packets through the VPN, or drops packets based on the configured rules.

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.

High-Level Test Description
Create rules with overlapping ranges and conflicting entries. Verify the rules are applied in the order they are configured.
Findings: PASS – This test is conducted as part of MOD_cPP_FW FFW_RUL_EXT.1.8 Tests 1 and 2. Since the VPN SPD is implemented as firewall policy rules with the VPN specified as the source or destination interface, the behavior seen in MOD_cPP_FW FFW_RUL_EXT.1.8 Tests 1 and 2.

FCS_IPSEC_EXT.1.2

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

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

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

High-Level Test Description
Verify that packets matching a bypass rule are forwarded without modification or encryption and that packets not matching any configured rules are dropped by a default deny rule.
Findings: PASS – This test is performed as part of FCS_IPSEC_EXT.1.1 Test 1 Step 2 in which a plaintext packet was successfully transmitted through the TOE. This test is conducted as part of MOD_cPP_FW FFW_RUL_EXT.1.9 which shows the TOE implementing a default drop rule for packets not matching a configured rule.

FCS_IPSEC_EXT.1.3

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

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

High-Level Test Description
Verify the TOE can initiate and successfully establish an IPsec connection in tunnel mode.
Findings: PASS – The evaluator confirmed the TOE successfully established an IPsec connection in tunnel mode.

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

High-Level Test Description
Verify the TOE can initiate and successfully establish an IPsec connection in transport mode.
Findings: PASS – The evaluator confirmed the TOE successfully established an IPsec connection in transport mode.

FCS_IPSEC_EXT.1.4

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

High-Level Test Description
Attempt to establish an ESP connection using each claimed algorithm. Verify all connection attempts succeed.
Findings: PASS – The evaluator confirmed that the TOE can use each claimed algorithm to protect ESP communications.

FCS_IPSEC_EXT.1.5

275 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.

High-Level Test Description
Show that the TOE will not permit establishing the VPN IKEv1 in aggressive mode.
Findings: PASS – The evaluator confirmed the TOE will not negotiate IKEv1 aggressive mode.

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

High-Level Test Description
Attempt to establish a transport mode IPsec connection that traverses a NAT router. Verify the connection succeeds.
Findings: PASS – The evaluator confirmed the TOE is able to establish an IPsec connection that traverses a NAT router.

FCS_IPSEC_EXT.1.6

276 The evaluator shall configure the TOE to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

High-Level Test Description
Attempt to negotiate each claimed encryption algorithm with IKEv1 and IKEv2. Verify the connection succeeds with each IKE version and each encryption algorithm.
Findings: PASS – The evaluator confirmed the TOE is able to uses each claimed algorithm to negotiate IKEv1 and IKEv2 connections.

FCS_IPSEC_EXT.1.7

277 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."

278 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.

High-Level Test Description
The TOE does not select 'number of bytes.'
Findings: N/A

NIAP TD0633

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

High-Level Test Description
For IKEv1 and IKEv2, establish Phase 1 SAs. Verify the TOE initiates a rekey of the Phase 1 SAs before 1 hour has elapsed.
Findings: PASS – The evaluator confirmed the TOE rekeyed the Phase 1 SAs before 1 hour had elapsed.

FCS_IPSEC_EXT.1.8

279 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."

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

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

High-Level Test Description
For IKEv1 and IKEv2, establish Phase 2 SAs. Verify the TOE initiates a rekey of the Phase 2 SAs before the configured number of types has been reached.
Findings: PASS – The evaluator confirmed the TOE rekeyed the Phase 2 SAs before 5MB had been sent.

NIAP TD0633

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

High-Level Test Description
For IKEv1 and IKEv2, establish Phase 2 SAs. Verify the TOE initiates a rekey of the Phase 2 SAs before 0.75 hours have elapsed.
Findings: PASS – The evaluator confirmed the TOE rekeyed the Phase 2 SAs before 0.75 hours had elapsed.

FCS_IPSEC_EXT.1.10

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

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

Findings:	The [ST] does not claim the first selection (security strength of the DH group).

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

Findings:	The [ST] claims the second selection (size of the PRF). Section 6.5 of the [ST] indicates nonces are the following lengths: 128 bits for SHA-1 and SHA-256. 256 bits for SHA-384 and SHA-512 Both nonce sizes are at least 128-bits and half the size of the associated hashes.

FCS_IPSEC_EXT.1.11

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

High-Level Test Description
Attempt to negotiate each claimed DH group with IKEv1 and IKEv2. Verify the connection succeeds with each IKE version and each DH group.
Findings: PASS – The evaluator confirmed the TOE supports each claimed DH group with each claimed IKE version.

FCS_IPSEC_EXT.1.12

283 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.

High-Level Test Description
For IKEv1 and IKEv2, attempt to establish an IKE connection using each supported algorithm. Verify the IKE connection succeeds.
Findings: PASS – The evaluator confirmed the TOE established an IKE connection using each claimed IKE algorithm.

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.

High-Level Test Description
For IKEv1 and IKEv2, attempt to establish an ESP connection using algorithms that are not supported. Verify the ESP connection fails.
Findings: PASS – The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to use unsupported algorithms.

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

High-Level Test Description
For IKEv1 and IKEv2, attempt to establish an IKE connection using algorithms that are not supported. Verify the IKE connection fails.
Findings: PASS – The evaluator confirmed the TOE will not establish an IKE connection when the peer attempts to use unsupported algorithms.

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.

High-Level Test Description
For IKEv1 and IKEv2, attempt to establish an ESP connection using algorithms that are not supported. Verify the IKE connection succeeds but the ESP connection fails.
Findings: PASS – The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to use unsupported algorithms.

FCS_IPSEC_EXT.1.13

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

FCS_IPSEC_EXT.1.14

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

286 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.

High-Level Test Description
The TOE does not select CN identifiers.
Findings: N/A

- 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.

High-Level Test Description
The TOE does not select SAN identifiers.
Findings: N/A

- 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 '\0'. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the evaluator shall configure the SAN so it matches the reference identifier.

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

High-Level Test Description
The TOE does not select CN identifies.
Findings: N/A

- 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.

High-Level Test Description
The TOE does not select SAN identifiers.
Findings: N/A

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

High-Level Test Description
Configure the TOE to used X.509 authentication based on the DN. Verify the connection succeeds when the peer presents a valid certificate with a matching DN.
Findings: PASS – The evaluator confirmed the TOE successfully authenticates the peer when the peer uses a certificate with a matching DN.

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

High-Level Test Description
Configure the TOE to used X.509 authentication based on the DN. Verify the connection fails when the peer presents an otherwise valid certificate with the CN duplicated (i.e., present twice) or with a null character inserted in the OU field.
Findings: PASS – The evaluator confirmed the TOE rejects connections from the peer when the peer uses a certificate with a different DN (CN presented twice, null character in the OU filed).

5.1.3 FCS_SSHS_EXT.1 SSH Server

5.1.3.1 TSS

FCS_SSHS_EXT.1.2

NIAP TD0631

287 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).

Findings:	[ST] Section 6.4 states the TOE supports SSH-RSA as acceptable public key algorithm used for client authentication. The evaluator confirmed this is consistent with FCS_COP.1/SigGen which selects RSA.

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

Findings:	[ST] Section 6.4 states, "The TOE establishes a user identity by either verifying that the SSH client's present public key matches the one that is stored within the SSH server's authorized keys file…"

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

Findings:	[ST] Section 6.7 describes how password-based authentication is used in SSH connections.

FCS_SSHS_EXT.1.3

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

Findings:	[ST] Section 6.4 states the TOE examines the size of each received SSH packet. If the packet is greater than 256KB, it is automatically dropped.

FCS_SSHS_EXT.1.4

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

Findings:	[ST] Section 6.4 does not identify any optional SSH characteristics supported by the TOE. This section indicates, "The TOE utilizes AES-CBC-128 and AES-CBC-256 for SSH encryption." The evaluator confirmed these algorithms are consistent with the selections in FCS_SSHS_EXT.1.4.

FCS_SSHS_EXT.1.5

NIAP TD0631

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

Findings:	[ST] Section 6.4 identifies SSH_RSA as the only hostkey algorithm supported by the TOE. The evaluator confirmed this is consistent with the selection in FCS_SSHS_EXT.1.5.

FCS_SSHS_EXT.1.6

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

Findings:	[ST] Section 6.4 lists HMAC-SHA1, HMAC-SHA2-256 and HMAC-SHA2-512 as the supported data integrity algorithms. The evaluator confirmed this list matches the selections in FCS_SSHS_EXT.1.6.

FCS_SSHS_EXT.1.7

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

Findings:	[ST] Section 6.4 list Diffie-Hellman Group 14 SHA-1 as the supported key exchange algorithm. The evaluator confirmed this algorithm claim is consistent with FCS_SSHS_EXT.1.7.

FCS_SSHS_EXT.1.8

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

a) Both thresholds are checked by the TOE.

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

Findings:	[ST] Section 6.4 states that the TOE will initiate a rekey an SSH connection after reaching either 1 hour or 1 gig of data, whichever occurs first.

5.1.3.2 Guidance Documentation

FCS_SSHS_EXT.1.4

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

Findings:	[SUPP] “Configuration and use of approved cryptographic algorithms” section indicates that no further configuration is needed to ensure the SSH server conforms with the description in the TSS after the FIPS-CC mode of operation is configured.

FCS_SSHS_EXT.1.5

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

Findings:	[SUPP] “Configuration and use of approved cryptographic algorithms” section indicates that no further configuration is needed to ensure the SSH server conforms with the description in the TSS after the FIPS-CC mode of operation is configured.

FCS_SSHS_EXT.1.6

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

Findings:	[SUPP] “Configuration and use of approved cryptographic algorithms” section indicates that no further configuration is needed to ensure the SSH server conforms with the description in the TSS after the FIPS-CC mode of operation is configured.

FCS_SSHS_EXT.1.7

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

Findings:	The Diffie-Hellman group should be set to Group 14 (2048-bit modulus) as mentioned in [SUPP] section “Enabling administrative access” as per the evaluated configuration. No additional configuration is needed to ensure SSH conforms to the description in the TSS.

FCS_SSHS_EXT.1.8

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

Findings:	The thresholds are not configurable.

5.1.3.3 Tests

FCS_SSHS_EXT.1.2

NIAP TD0631

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

NIAP TD0631

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

High-Level Test Description
Verify the TOE allows users to authenticate using ssh-rsa.
Findings: PASS – The evaluator confirmed ssh-rsa could be used to authenticate to the TOE while performing FIA_UIA_EXT.1 Test 1.

NIAP TD0631

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

High-Level Test Description
Verify that using an SSH key pair that has not been configured on the TOE results in an authentication failure.
Findings: PASS – The evaluator confirmed that attempting to authenticate to the TOE using an SSH key ssh-rsa that was not configured as trusted resulted in an authentication failure while performing FIA_UIA_EXT.1 Test 1.

NIAP TD0631

304 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.

High-Level Test Description
Verify the TOE allows users to authenticate using a password.
Findings: PASS – The evaluator confirmed a password could be used to authenticate to the TOE while performing FIA_UIA_EXT.1 Test 1.

NIAP TD0631

305 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.

High-Level Test Description
Verify using an incorrect password results in an authentication failure.
Findings: PASS – The evaluator confirmed that using an incorrect password resulted in an authentication failure while performing FIA_UIA_EXT.1 Test 1.

FCS_SSHS_EXT.1.3

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

High-Level Test Description
Transmit a packet larger than allowed by the TOE SSH implementation and verify that the TOE rejects the packet.
Findings: PASS – The evaluator confirmed the TOE rejects SSH packets larger than 256KB.

FCS_SSHS_EXT.1.4

307 The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish a SSH connection. To verify this, the evaluator shall start session establishment for a SSH connection from a remote client (referred to as 'remote endpoint' below). The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the captured traffic that the TOE offers all the ciphers defined in the TSS for the TOE for SSH sessions, but no additional ones compared to the definition in the TSS. The evaluator shall perform one successful negotiation of an SSH session to verify that the TOE behaves as expected. It is sufficient to observe the successful negotiation

of the session to satisfy the intent of the test. If the evaluator detects that not all ciphers defined in the TSS for SSH are supported by the TOE and/or the TOE supports one or more additional ciphers not defined in the TSS for SSH, the test shall be regarded as failed.

High-Level Test Description
Connect to the TOE using each claimed SSH cipher. Verify the TOE only proposes the claimed ciphers.
Findings: PASS – The evaluator confirmed that the TOE successfully negotiates each claimed encryption algorithms and only proposes the claimed encryption algorithms.

FCS_SSHS_EXT.1.5

NIAP TD0631

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

NIAP TD0631

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

High-Level Test Description
Using an SSH client, connect to the TOE server using each hostkey algorithm and verify each connection succeeds.
Findings: PASS – The evaluator confirmed the TOE successfully identifies itself with ssh-rsa.

NIAP TD0631

310 Has effectively been moved to FCS_SSHS_EXT.1.2.

NIAP TD0631

311 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.

NIAP TD0631

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

High-Level Test Description
Using an SSH client, connect to the TOE server using the specified public key algorithms in turn. This requires the TOE to be loaded with a public key corresponding to the key pair.
Findings: PASS – The evaluator confirmed that the SSH connection was rejected when the client proposed a hostkey algorithm not claimed by the TOE

FCS_SSHS_EXT.1.6

313 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.

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

High-Level Test Description
Attempt to negotiate each claimed integrity algorithm and show that each algorithm is used in a successful connection.
Findings: PASS – The evaluator confirmed the TOE successfully establishes an SSH connection with each claimed integrity algorithm.

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

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

High-Level Test Description
Attempt to negotiate an SSH connection with hmac-md5 (i.e., an algorithm not included in the ST) and verify the connection fails.
Findings: PASS – The evaluator confirmed an SSH connection with the TOE fails when hmac-md5 is the only integrity algorithm proposed by the client.

FCS_SSHS_EXT.1.7

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

High-Level Test Description
Attempt to negotiate an SSH connection with diffie-hellman-group1-sha1 and verify the connection fails.
Findings: PASS – The evaluator confirmed an SSH connection with the TOE fails when diffie-hellman-group1-sha1 is the only key exchange algorithm proposed by the client.

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

High-Level Test Description
Attempt to negotiate each claimed key exchange method and show that each method is used in a successful connection.

High-Level Test Description
Findings: PASS – The evaluator confirmed the TOE successfully establishes a connection using diffie-hellman-gorup14-sha1.

FCS_SSHS_EXT.1.8

319 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.

320 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).

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

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

322 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).

323 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).

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

High-Level Test Description
Using a custom SSH client, connect to the TOE and send slightly more than 1GB of data. Verify the TOE initiates a rekey before 1GB of data has been encrypted or decrypted using a key.
Findings: PASS – The evaluator confirmed the TOE initiates a rekey before 1 GB of data has been encrypted or decrypted using a key.

325 If one or more thresholds that are checked by the TOE to fulfil the SFR are configurable, the evaluator needs to verify that the threshold(s) can be configured as described in the guidance documentation and the evaluator needs to test that

modification of the thresholds is restricted to Security Administrators (as required by FMT_MOF.1/Functions).

Findings:	These thresholds are not configurable.

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

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

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

Findings:	The TOE does not have hardware limitations.

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

5.1.4.1 TSS

FCS_TLSC_EXT.1.1

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

Findings:	[ST] Section 6.3.3 contains the description of the TLS implementation and lists the twelve ciphersuites supported by the TOE. The evaluator confirmed the list in the TSS matches the list in FCS_TLSC_EXT.1.1.

FCS_TLSC_EXT.1.2

328 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.

Findings:	[ST] Section 6.3.3 states the administrator can configure the reference identifier as an IP address or DNS name. The TSS also describes how the TOE compares the configured identifier to the SAN and/or CN. The TOE supports wildcards for DNS names in the CN and SAN.

329 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.

Findings:	N/A. The TOE is not a distributed TOE.

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

Findings:	N/A. IP addresses are not supported in CN.

FCS_TLSC_EXT.1.4

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

Findings:	[ST] Section 6.3.3 states the TOE sends the Supported Elliptic Curves extension with NIST curves P-256, P-384, and P-521. This is default behavior.

5.1.4.2 Guidance Documentation

FCS_TLSC_EXT.1.1

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

Findings:	The [SUPP], section "Enabling FIPS-CC mode" provides instructions to enable the FIPS-CC mode which makes the TLS client conforms with the description in the TSS. [SUPP] section "Remote access requirements" also states that the "FIPS-CC mode of operation restricts the cipher suites used by HTTPS and SSH to a subset of the NDcPP compliant suites". No further action needed.

FCS_TLSC_EXT.1.2

333 The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

Findings:	The [SUPP] document provides instructions to configure the reference identifier in the "FortiAnalyzer configuration" section. As per section "Miscellaneous", SANs are supported in the certificates when acting as a TLS client. The [CLI] also provides detailed commands in the "log > config log fortianalyzer setting" section starting on page 492. The reference identifier can be IP addresses or DNS.

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

Findings:	The TOE is not a distributed TOE.

FCS_TLSC_EXT.1.4

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

Findings:	[SUPP] "Configuration and use of approved cryptographic algorithms" section indicates that no further configuration is needed to ensure the TLS client conforms with the description in the TSS after the FIPS-CC mode of operation is configured.

5.1.4.3 Tests

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

FCS_TLSC_EXT.1.1

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to negotiate all specifically claimed ciphersuites. Verify each connection succeeds.
Findings: PASS – The evaluator confirmed the TOE successfully negotiated a TLS connection using each ciphersuite.

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

High-Level Test Description
Construct two X.509 certificates: one with an extendedKeyUsage with 'serverAuth' and another without. Verify the TOE successfully connects to a server using an X.509 certificate with the 'serverAuth' purpose and fails to connect when the server presents a certificate without the 'serverAuth' purpose.
Findings: PASS – The evaluator confirmed the TOE successfully establishes a TLS connection when the server certificate contains the 'serverAuth' purpose and does not establish a connection when the TLS server certificate does not contain the 'serverAuth' purpose.

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

High-Level Test Description
User a Lightship developed TLS server to present a certificate that does not match the server selected ciphersuite. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE does not establish a TLS connection if the server presents a certificate whose algorithm does not match the server selected ciphersuite.

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

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

High-Level Test Description
Using a Lightship developed TLS server, have the server select the TLS_NULL_WITH_NULL_NULL (cipher ID 0x0000) ciphersuite and verify the TOE rejects the connection.
Findings: PASS – The evaluator confined the TOE denies connections to a TLS server that attempts to negotiate the TLS_NULL_WITH_NULL_NULL ciphersuite.

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that selects a ciphersuite that was not proposed by the client. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the TOE rejects a TLS connection if the server selects a ciphersuite that was not proposed by the TOE.

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that selects an unsupported elliptic curve. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects a TLS connection if the server selects a key exchange curve that is not supported by the TOE.

341 Test 5: The evaluator performs the following modifications to the traffic:

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that selects an unsupported TLS version. Verify the connection fails.

High-Level Test Description
Findings: PASS – The evaluator confirmed the TOE rejects TLS connections when the server selects a non-supported TLS version.

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that corrupts the signature block in the Server Key Exchange message. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects TLS connections when the signature block in the Server Key Exchange message is corrupted/invalid.

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

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that modifies the Server Finished message. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects TLS handshakes when the Server Finished message is corrupted/invalid.

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that sends a garbled message after the Change Cipher Spec message. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects TLS handshakes when the server sends a garbled message after the Change Cipher Spec message.

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

High-Level Test Description
Have the TOE connect to a Lightship TLS test server that modifies the nonce sent to the client. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects the TLS handshake when the server nonce is modified.

FCS_TLSC_EXT.1.2

343 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.

344 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.

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

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

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

High-Level Test Description
Have the TOE connect to an OpenSSL TLS test server that presents a certificate with an invalid identifier in the CN. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE does not establish a connection when a server presents a certificate with an invalid identifier in the CN.

b) Test 2 [conditional]: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN that matches the reference identifier. The

evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, URI). When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the SAN.

High-Level Test Description
For each identifier type, have the TOE connect to an OpenSSL TLS test server that presents a certificate with a valid identifier in the CN but an invalid identifier in the SAN. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE does not establish a connection when a server presents a certificate with a valid identifier in the CN but an invalid identifier 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.

High-Level Test Description
Have the TOE connect to a Lightship developed TLS test server that presents a certificate with a valid identifier in the CN and no SAN extension. Verify the connection succeeds.
Findings: PASS – The evaluator confirmed the TOE successfully establishes a connection when the server presents a certificate with a valid identifier in the CN and no SAN extension.

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

High-Level Test Description
For each identifier type, have the TOE connect to an OpenSSL TLS test server that presents a certificate with an invalid identifier in the CN but a valid identifier in the SAN. Verify the connection succeeds.
Findings: PASS – The evaluator confirmed the TOE successfully establishes a connection when the server presents a certificate with a valid identifier in the SAN and an invalid identifier in the CN.

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

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

High-Level Test Description
Have the TOE connect to an OpenSSL TLS test server that presents certificates with CN and SAN identifiers where the wildcard is not in the left-most position. Verify the connection fails.

High-Level Test Description
Findings: PASS – The evaluator confirmed the TOE does not establish a connection when the server presents a certificate where the wildcard is not in the left-most position, whether the identifier is in the CN or the SAN.

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

High-Level Test Description
Have the TOE connect to an OpenSSL TLS test server that presents certificates with CN and SAN identifiers with a single wildcard in the left-most position. Configure the TOE with a reference identifier with a single left-most label in addition to the fixed identifier. Verify the connections succeed. Configure the TOE with reference identifiers that only match the static identifier and contain two left-most labels in addition to the static identifier Verify the connections fail.
Findings: PASS – The evaluator confirmed the TOE correctly processed a wildcard in the left-most label, accepting it when a single left-most label was expected and rejecting it when no left-most or two left-most labels were expected.

NIAP TD0634

f) Test 6 [conditional]: If IP address identifiers 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 192.168.0.1, 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.

High-Level Test Description
Have the TOE connect to an OpenSSL TLS test server that presents a certificate with an IP address containing a wildcard in the CN. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects certificates with IP address identifiers containing a wildcard in the CN.

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

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

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

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

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

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

347 **FCS_TLSC_EXT.1.3**

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

349 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.

High-Level Test Description
This test case is performed as part of FIA_X509_EXT.1.1 Test 1.
Findings: PASS – The evaluator confirmed a connection using a valid certificate chain succeeds in conjunction with FIA_X509_EXT.1.1/Rev Test 1.

350 Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the

revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

High-Level Test Description
This test case is performed as part of FIA_X509_EXT.1.1 Test 1.
Findings: PASS – The evaluator confirmed the TOE does not establish a connection when certificate validation fails.

351 Test 3 [conditional]: The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

High-Level Test Description
The TOE does not claim any override mechanisms.
Findings: N/A

FCS_TLSC_EXT.1.4

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

High-Level Test Description
Using a Lightship developed TLS server, force the TOE client to negotiate each claimed curve. Verify each connection succeeds.
Findings: PASS – The evaluator confirmed the TOE establishes a connection using each supported curve.

5.1.5 FCS_TLSS_EXT.1 Extended: TLS Server Protocol

5.1.5.1 TSS

FCS_TLSS_EXT.1.1

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

Findings:	[ST] Section 6.3.2 describes the TLS implementation and lists the twelve supported ciphersuites. The evaluator confirmed the ciphersuites are identical to those listed in FCS_TLSS_EXT.1.1.

FCS_TLSS_EXT.1.2

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

Findings:	[ST] Section 6.3.2 describes how the TOE rejects any TLS protocol version that is not 1.2 or 1.1 (implicitly rejecting old SSL and TLS versions).

FCS_TLSS_EXT.1.3

NIAP TD0635

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

Findings:	[ST] Section 6.3.2 identifies secp256r1, secp384r1, secp521r1, and 2048 bits (which corresponds to Group 14) as the key agreement parameters for ECDHE and DHE

FCS_TLSS_EXT.1.4

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

Findings:	N/A: The TOE does not perform session resumption based on Session IDs.

357 If session tickets are supported, the evaluator shall verify that the TSS describes that the session tickets are encrypted using symmetric algorithms consistent with FCS_COP.1/DataEncryption. The evaluator shall verify that the TSS identifies the key lengths and algorithms used to protect session tickets.

Findings:	[ST] Section 6.3.2 describes that the TLS server supports session tickets. Session tickets adhere to the structural format provided in section 4 of RFC 5077. Session tickets are encrypted according to the TLS negotiated symmetric encryption algorithm.

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

Findings:	[ST] Section 6.3.2 states that Session Tickets adhere to the structural format provided in section 4 of RFC 5077.

NIAP TD0569

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

Findings:	The TOE performs session resumption using session tickets in a single context. The session resumption flow is defined in RFC 5077.

5.1.5.2 Guidance Documentation

FCS_TLSS_EXT.1.1

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

Findings:	No further configuration is needed to ensure the TLS server conforms with the description in the TSS after FIPS-CC mode is enabled as per [SUPP] section "Configuration and use of approved cryptographic algorithms".

FCS_TLSS_EXT.1.2

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

Findings:	No further configuration is needed to ensure the TLS server conforms with the description in the TSS after FIPS-CC mode is enabled as per [SUPP] section "Configuration and use of approved cryptographic algorithms".

FCS_TLSS_EXT.1.3

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

Findings:	The [SUPP] section "Enabling administrative access" and [CLI] section "system > config system global" provide the command to be used to configure the Diffie-Hellman parameter size.

FCS_TLSS_EXT.1.4

NIAP TD0569

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

Findings:	The TOE supports session resumption based on session tickets according to RFC 5077 by default. No configuration is needed.

5.1.5.3 Tests

FCS_TLSS_EXT.1.1

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

High-Level Test Description
Connect to the TOE using each claimed ciphersuite and verify each connection succeeds.
Findings: PASS – The evaluator confirmed the TOE allows TLS connections with each claimed ciphersuite.

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

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using an unsupported ciphersuites and verify the TOE rejects the connection. Then connect to the TOE using TLS_NULL_WITH_NULL_NULL and verify the TOE rejects the connection.
Findings: PASS – The evaluator confirmed the TOE rejects connection using the TLS_NULL_WITH_NULL_NULL ciphersuite.

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

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

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE and modify the first payload byte in the Client Finished message. Verify the connection is rejected.
Findings: PASS – The evaluator confirmed the TOE rejects a connection when the client sends a modified/corrupted Client Finished message.

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

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

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

High-Level Test Description
Perform a successful handshake and verify that the Server Finished message is encrypted.
Findings: PASS – The evaluator confirmed the TOE encrypts the Sever Finished message.

FCS_TLSS_EXT.1.2

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

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE and attempt to negotiate SSL 2.0, SSL 3.0, and TLS 1.0. Verify each connection is rejected.
Findings: PASS – The evaluator confirmed the TOE does not negotiate unsupported versions of TLS/SSL.

FCS_TLSS_EXT.1.3

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

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

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using a valid ECDHE ciphersuite and curve combination and verify that the public key size that comes back in the Server Key Exchange message matches the expected bit size for the chosen curve.
Findings: PASS – The evaluator confirmed the TOE supports each claimed elliptic curve.

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

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

368 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).

High-Level Test Description
Using a Lightship developed TLS client, connect to the TOE using a valid DHE ciphersuite and verify that the public key size that comes back in the Server Key Exchange message matches the expected bit size for the chosen DH parameter.
Findings: PASS – The evaluator confirmed the TOE used the claimed DH parameter size.

369 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.

High-Level Test Description
RSA key establishment ciphersuites are not supported by the TOE.
Findings: N/A

FCS_TLSS_EXT.1.4

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).

370 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.

NIAP TD0569

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

acceptable for a control channel to establish and application channel to resume the session.

High-Level Test Description
The TOE supports session resumption based session tickets.
Findings: N/A

371 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).

High-Level Test Description
The TOE does not support session resumption based on session IDs.
Findings: N/A

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.

NIAP TD0569

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

High-Level Test Description
The TOE does not support session resumption based on session IDs.
Findings: N/A

372 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):

NIAP TD0556

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 an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

High-Level Test Description
Verify the TOE can successfully perform session resumption using Session Tickets.
Findings: PASS – The evaluator confirmed the TOE resumed a session when provided with a valid Session Ticket.

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.

NIAP TD0569

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

High-Level Test Description
Verify that the TOE will not resume a session when an altered/invalid session ticket is presented.
Findings: PASS – The evaluator confirmed the TOE did not resume a session when an altered/invalid session ticket was presented.

5.2 Identification and Authentication (FIA)

5.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

5.2.1.1 TSS

373 The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify

the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

Findings:	[ST] Section 6.8 states the TOE checks for certificate validity during TLS client connections, during IPsec peer connections, and when certificates are loaded into the TOE. Rules for extendedKeyUsage are supported by the TOE. The TOE supports CRL for certificate revocation used for authentication. The TSS states, “If, during the entire trust chain verification activity, any certificate under review fails a verification check, then the certificate is deemed untrusted and the connection is rejected.”

374 The TSS shall describe when revocation checking is performed and on what certificates. If the revocation checking during authentication is handled differently depending on whether a full certificate chain or only a leaf certificate is being presented, any differences must be summarized in the TSS section and explained in the Guidance.

Findings:	[ST] Section 6.8 states revocation checking is performed on leaf and intermediate CA certificates during authentication steps.

5.2.1.2 Guidance Documentation

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

Findings:	The [CLI] section “config vpn certificate setting” in page 1223 describes the commands to check the validity of the certificates imported. The commands include options to check the validity of the certificates imported (CA, chain). The [SUPP] states all CRLs should be imported to the Fortigate unit in the “VPN and Certificate Specific Settings > CAs and CRLs” section. [SUPP] section “Miscellaneous” describes the rules on how the FortiGate validates extendedKeyUsage field. Certificate revocation is described in section “Certificates > Uploading a certificate using a GUI > CRL” in [ADMIN].

5.2.1.3 Tests

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

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

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

High-Level Test Description
Have the TOE connect to server/peer that sends the intermediate CA certificate necessary to validate the chain in one connection and omits the intermediate CA certificate in a second connection. Verify the connection succeeds when the TOE can validation the full chain of certificates and fails when the chain is incomplete.
Findings: PASS – The evaluator confirmed the TOE successfully validates X.509 certificates when a full chain is provided and rejects certificates when the chain is incomplete.

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

High-Level Test Description
Verify the TOE rejects connections when an expired certificate is presented.
Findings: PASS – The evaluator confirmed the TOE rejects expired certificates.

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates-–conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

High-Level Test Description
Configure the TOE to use CRLs. Verify the TOE does not establish a connection when the Intermediate or leaf certificates are revoked.
Findings: PASS – The evaluator confirmed the TOE does not establish a connection when a certificate is revoked.

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

High-Level Test Description
Present a CRL that is signed by a CA certificate without the CRL signing bit set. Verify the TOE does not consider the CRL valid.
Findings: PASS – The evaluator confirmed the TOE does not consider a CRL valid if the CA cert used to sign the CRL does not have the CRL signing bit set.

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

High-Level Test Description
Have the TOE to connect to a server/peer which will send back a certificate whose 5th byte has been modified. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the connection fails when the TOE receives a certificate with a modified 5th byte.

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

High-Level Test Description
Have the TOE to connect to a server/peer which will send back a certificate whose first signature byte has been modified. Verify the connection fails.
Findings: PASS – The evaluator confirmed the connection fails when the TOE receives a certificate with a modified signature.

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

High-Level Test Description
Have the TOE to connect to a server/peer which will send back a certificate whose public key has been modified. Verify the connection fails
Findings: PASS – The evaluator confirmed the connection fails when the TOE receives a certificate with a modified public key.

NIAP TD0527 (REVISED 1 December 2020)

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.

High-Level Test Description
Have the TOE connect to a server/peer that presents a valid chain of ECDSA certificates. Verify the connection succeeds.

High-Level Test Description
Findings: PASS – Th evaluator confirmed the TOE can establish a connection when a chain of ECDSA certificates is used.

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.

High-Level Test Description
Have the TOE connect to a TLS server that presents an intermediate ECDSA certificate that specifies its curve using explicit parameters. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE rejects an intermediate ECDSA certificate that specify its curve using explicit parameters.

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

High-Level Test Description
Attempt to load an intermediate ECDSA CA certificate with a named curve onto the TOE. Verify the load succeeds. Attempt to load an intermediate ECDSA CA certificate with an explicitly defined curve onto the TOE. Verify the load fails.
Findings: PASS – The evaluator confirmed the TOE allows an intermediate CA certificate using a named curve to be loaded and rejects an intermediate CA certificate using an explicitly specified curve.

377 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.

378 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).

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

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

High-Level Test Description
Have the TOE connect to a server that presents an intermediate CA certificate without the Basic Constraints extension. Verify the connection fails. Attempt to load an intermediate CA certificate without the Basic Constraints extension. Verify the operation fails.
Findings: PASS – The evaluator confirmed the TOE will not trust an intermediate CA certificate without the Basic Constraints extension.

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

High-Level Test Description
Clone the known good CA certificate and set the basicConstraints extension to have the CA flag set to FALSE. Replace the existing known-good CA with the cloned CA. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE will not trust an intermediate CA certificate without the Basic Constraints extension.

380 The evaluator shall repeat these tests for each distinct use of certificates. Thus, for example, use of certificates for TLS connection is distinct from use of certificates for trusted updates so both of these uses would be tested. But there is no need to repeat the tests for each separate TLS channel in FTP_ITC.1 and FTP_TRP.1/Admin (unless the channels use separate implementations of TLS).

Findings:	These tests were performed for the TLS Client and IPsec trusted channels.

5.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication

5.2.2.1 TSS

381 The evaluator shall check the TSS to ensure that it describes how the TOE chooses which certificates to use, and any necessary instructions in the administrative guidance for configuring the operating environment so that the TOE can use the certificates.

Findings:	[ST] Section 6.8 states the TOE uses the leaf certificate provided by the IT entity. The TOE uses the intermediate certificates loaded in its trust store and provided by the IT entity to perform FIA_X509_EXT.1/Rev certificate path validation. Trust anchor certificates have to be configured according to the guidance document.

382 The evaluator shall examine the TSS to confirm that it describes the behaviour of the TOE when a connection cannot be established during the validity check of a certificate used in establishing a trusted channel. The evaluator shall verify that any distinctions between trusted channels are described. If the requirement that the administrator is able to specify the default action, then the evaluator shall ensure that the guidance documentation contains instructions on how this configuration action is performed.

Findings:	[ST] Section 6.8 states the TOE will use the last known information about the certificate to determine either to accept or reject it when the connection cannot be established to the CRL. If there is no last known information, the TOE will accept the certificate. The administrator is not able to select the default action.

5.2.2.2 Guidance Documentation

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

Findings:	The [ADMIN] document describes how to configure the TOE to use certificates in the "System > Certificates > Uploading a certificate using the GUI" section starting on page 1012. The instructions to use an already imported certificate are located in [ADMIN] section "Configure your FortiGate to use the signed certificate" on page 1023. In addition, the [ADMIN] document in section "VPN > IPSec VPN > General IPsec VPN configuration > Pre-shared key vs digital certificates" starting page 1435 describes the requirements for peer certificates to be validated.

5.2.2.3 Tests

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

385 The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate, and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the guidance documentation to determine that all supported administrator-configurable options behave in their documented manner.

High-Level Test Description
Have the TOE connect to a server/peer while the TOE is unable to contact the revocation server. Verify the TOE accepts certificates when it does not have a cached revocation list and uses the last cached revocation list when available.

High-Level Test Description
Findings: PASS – The evaluator confirmed when the TOE is unable to fetch revocation status, it accepts connections when the revocation status is unknown, but uses the last cached revocation status if available.

5.2.3 FIA_X509_EXT.3 Extended: X509 Certificate Requests

5.2.3.1 TSS

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

Findings:	FIA_X509_EXT.3.1 does not select “device-specific information,” so this Evaluation Activity is not applicable.

5.2.3.2 Guidance Documentation

387 The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate Request. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the Certification Request.

Findings:	[ADMIN] Section “System > Certificates > Procure and import a signed SSL certificate” starting on page 1020. The section describes how to generate a CSR and import a signed certificate.

5.2.3.3 Tests

388 The evaluator shall perform the following tests:

a) Test 1: The evaluator shall use the guidance documentation to cause the TOE to generate a Certification Request. The evaluator shall capture the generated message and ensure that it conforms to the format specified. The evaluator shall confirm that the Certification Request provides the public key and other required information, including any necessary user-input information.

High-Level Test Description
Using the TOE CSR generator, create a new CSR and download it to an external CA entity for signing. Using OpenSSL, verify that the information in the CSR is as expected.
Findings: PASS – The evaluator confirm the TOE is capable of generating CSRs that include all of the claimed 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 certificate response message, and demonstrate that the function succeeds.

High-Level Test Description
Attempt to import a signed certificate when there is not a valid path to verify trust. Verify the import fails.

High-Level Test Description
Attempt to import a signed certificate when there is a valid path to verify trust. Verify the import succeeds.
Findings: PASS – The evaluator confirmed the TOE fails to load a signed CSR (certificate) when it cannot complete the trust chain and successfully loads a signed CSR when it is able to complete the trust chain.

5.3 Security management (FMT)

5.3.1 FMT_MOF.1/Functions Management of security functions behaviour

5.3.1.1 TSS

389 For distributed TOEs see [NDcPP-SD] chapter 3.4.1.1.

Findings:	N/A, the TOE is not a distributed TOE.

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

Findings:	[ST] Section 6.9 states how the administrator determines or modifies the behavior of the transmission of audit data to an external IT entity.

5.3.1.2 Guidance Documentation

391 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

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

Findings:	In the [CLI], section “config log fortianalyzer setting” starting on page 492 shows the commands to modify the behaviour of transmitting audit data to a Fortianalyzer. Audit functionality when Local Audit Storage Space is full is described in [SUPP] section “Local Logging”.

5.3.1.3 Tests

393 Test 1 (if ‘transmission of audit data to external IT entity’ is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity without prior authentication as Security Administrator (by authentication as a user with no administrator privileges or without user authentication at all). Attempts to modify parameters without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user

authentication the user might not be able to get to the point where the attempt to modify the security related parameters can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
This test case is covered in FIA_UIA_EXT.1.1 Test 2 and Test 3 which show an unauthenticated user cannot perform actions prior to authentication. [ST] Section 6.9 says, “The TOE defines a single role, which is that of the Security Administrator.”
Findings: PASS – As part of FIA_UIA_EXT.1.1 Test 2 and Test 3, the evaluator confirmed unauthenticated users are not able to perform actions prior to authentication.

394 Test 2 (if ‘transmission of audit data to external IT entity’ is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity with prior authentication as Security Administrator. The effects of the modifications should be confirmed.

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

High-Level Test Description
FCS_TLSC_EXT.1.2 Test 6 shows the admin is able to modify the transmission of audit data to an external IT entity.
Findings: PASS – As part of FCS_TLSC_EXT.1.2 Test 6, the evaluator confirmed the administrator is able to specify the external IT entity the TOE transmits audit data to.

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

High-Level Test Description
The TOE does not claim this functionality.
Findings: N/A

397 Test 2 (if 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the handling of audit data with prior authentication as Security Administrator. The effects of the modifications should be confirmed. The term ‘handling of audit data’ refers to the different options for selection and assignments in SFRs FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2/LocSpace.

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

High-Level Test Description
The TOE does not claim this functionality.
Findings: N/A

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

High-Level Test Description
The TOE does not claim this functionality.
Findings: N/A

400 Test 2 (if 'audit functionality when Local Audit Storage Space is full' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify the behaviour when Local Audit Storage Space is full with prior authentication as Security Administrator. This attempt should be successful. The effect of the change shall be verified.

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

High-Level Test Description
The TOE does not claim this functionality.
Findings: N/A

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

High-Level Test Description
The TOE does not claim this functionality.

High-Level Test Description
Findings: N/A

403 Test 4 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen from the second selection with prior authentication as Security Administrator. This can be done in one test or in separate tests. The attempt(s) to determine the behaviour of the selected functions with administrator authentication shall be successful.

High-Level Test Description
The TOE does not claim this functionality.
Findings: N/A

5.3.2 FMT_MOF.1/Services Management

5.3.2.1 TSS

404 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.1.

Findings:	N/A, the TOE is not a distributed TOE.

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

Findings:	[ST] Section 6.9 states all the services the Security Administrator is able to start and stop and how the operation is performed.

5.3.2.2 Guidance Documentation

406 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

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

Findings:	The [ADMIN] document defines the Security Administrator role in the “System > Administrators > Administrator Profiles” section on page 856. The [ADMIN] throughout lists the functions that the Security Administrator is able to perform under various sections. The information aligns with the functions listed in the TSS.

5.3.2.3 Tests

408 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.

High-Level Test Description
This test case is covered in FIA_UIA_EXT.1.1 Test 2 and Test 3 which show an unauthenticated user cannot perform actions prior to authentication. [ST] Section 6.9 says, “The TOE defines a single role, which is that of the Security Administrator.”
Findings: PASS – As part of FIA_UIA_EXT.1.1 Test 2 and Test 3, the evaluator confirmed unauthenticated users are not able to perform actions prior to authentication.

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

High-Level Test Description
As the privileged user, attempt to stop one of the predefined services. The attempt will be successful. Privileged user service actions are performed as part of other tests.
Findings: PASS – The evaluator confirmed the Security Administrator is able to stop services.

5.3.3 FMT_MTD.1/CryptoKeys Management of TSF Data

5.3.3.1 TSS

410 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.1.

Findings:	N/A, the TOE is not a distributed TOE.

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

Findings:	[ST] Section 6.9 identifies the keys the security administrator is able to manage. These keys are described in detail in Table 23 in Section 6.2.2 of the [ST].

5.3.3.2 Guidance Documentation

412 For distributed TOEs see [NDcPP-SD] chapter 2.4.1.2.

Findings:	The TOE is not a distributed TOE.

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

Findings:	The [ADMIN] document defines the Security Administrator role in the “System > Administrators > Administrator Profiles” section on page 856. Section “System > Certificates” starting page 1012 describes how to import/export certificates and generate certificates. Deletion of keys are also mentioned in [SUPP] under sections “Disabling FIPS-CC mode” and “Key Zeroization”.

5.3.3.3 Tests

414 The evaluator shall try to perform at least one of the related actions (modify, delete, generate/import) without prior authentication as Security Administrator (either by authentication as a non-administrative user, if supported, or without authentication at all). Attempts to perform related actions without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to manage cryptographic keys can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
This test case is covered in FIA_UIA_EXT.1.1 Test 2 and Test 3 which show an unauthenticated user cannot perform actions prior to authentication. [ST] Section 6.9 says, “The TOE defines a single role, which is that of the Security Administrator.”
Findings: PASS – As part of FIA_UIA_EXT.1.1 Test 2 and Test 3, the evaluator confirmed unauthenticated users are not able to perform actions prior to authentication.

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

High-Level Test Description
Attempt to generate SSH and certificate cryptographic keys as the security administrator. Verify the generation succeeds.
Findings: PASS – The evaluator confirmed the Security Administrator is able to generate SSH and certificate cryptographic keys.

6 Evaluation Activities for Security Assurance Requirements

6.1 ASE: Security Target

6.1.1 General ASE

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

Findings:	See above sections.

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

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

Findings:	N/A, the TOE is not a distributed TOE.

6.2 ADV: Development

6.2.1 Basic Functional Specification (ADV_FSP.1)

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

419 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.

420 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.

421 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.

6.2.1.1 Evaluation Activity

422 *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.*

423 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.

424 The set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

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

6.2.1.2 Evaluation Activity

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

Findings:	See comments in the previous work unit.

6.2.1.3 Evaluation Activity

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

427 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.

428 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.

429 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'.

Findings:	See comments in the previous work unit.

6.3 AGD: Guidance Documents

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

431 Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in section A.9.1.1. (in the NDcPP-SD)

6.3.1 Operational User Guidance (AGD_OPE.1)

432 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.

433 In addition, the evaluator performs the EAs specified below.

6.3.1.1 Evaluation Activity

434 *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.*

Findings:	The documentation is available for public download from Fortinet's documentation web site (https://docs.fortinet.com).

6.3.1.2 Evaluation Activity

435 *The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.*

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

6.3.1.3 Evaluation Activity

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

Findings:	The [SUPP] provides wording indicating that the Network Processing Unit (NPU) is not FIPS-validated and it must be turned off in section “Disabling NPU support”.

6.3.1.4 Evaluation Activity

437 *The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.*

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

6.3.1.5 Evaluation Activity

438 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.

NIAP TD0536

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

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

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

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

Findings:	The TOE claims digital signatures. The process for obtaining the update and verifying downloaded file is not corrupted is described in [SUPP]. Additional information regarding the use of claimed digital signatures is provided in the Section Firmware of the [ADMIN] guide. The process for manually upgrading the TOE is provided in [SUPP] and [ADMIN]. [SUPP] makes it clear to an administrator which security functionality is covered and in scope.

6.3.2 Preparative Procedures (AGD_PRE.1)

439 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.

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

441 In addition, the evaluator performs the EAs specified below.

6.3.2.1 Evaluation Activity

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

443 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).

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

6.3.2.2 Evaluation Activity

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

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

6.3.2.3 Evaluation Activity

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

Findings:	See previous work unit.

6.3.2.4 Evaluation Activity

446 *The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.*

Findings:	The guidance documentation provides extensive information on managing the security of the TOE as an individual product. Additional best practice guidance provided within those documents help install a culture of secure manageability within a larger operational environment.

6.3.2.5 Evaluation Activity

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

448 The preparative procedures must:

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

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

Findings:	The entire [SUPP] document is designed to ensure the administrator is aware of how to configure the TOE to provide a protected administrative capability. The TOE has default TOE passwords. However, when placing the device into FIPS-CC mode, the administrator is required to change the password to meet the minimum password requirements as stated in the [SUPP]. These complexity requirements are enforced by the TOE rather than by policy.

6.4 ALC: Life-cycle Support

6.4.1 Labelling of the TOE (ALC_CMC.1)

449 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.

Findings:	The evaluator verified that the ST, TOE and Guidance are all labelled 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

6.4.2 TOE CM coverage (ALC_CMS.1)

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

Findings:	The evaluator verified that the ST, TOE and Guidance are all labelled 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

6.5 ATE: Tests

6.5.1 Independent Testing – Conformance (ATE_IND.1)

451 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.

452 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.

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

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

Findings:	A high level overview of the independent testing document is provided throughout the AAR. The full details of the Independent Testing effort are documented in the non-public Detailed Test Report. The TOE is not a distributed TOE.

6.6 Vulnerability Assessment

6.6.1 Vulnerability Survey (AVA_VAN.1)

455 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

456 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.

457 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.

6.6.1.1 Evaluation Activity (Documentation):

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

459 *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.*

NIAP TD0547

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

and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

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

461 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 3.4.1.2 and 3.5.1.2.

6.6.1.2 Evaluation Activity

462 The evaluator formulates hypotheses in accordance with process defined in Appendix A. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.

Findings: The following sources of public vulnerabilities were considered in formulating the specific list of flaws to be investigated by the evaluators, as well as to reference in directing the evaluators to perform key-word searches during the evaluation of the TOE. Hypothesis sources for public vulnerabilities were:

- NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search
- US-CERT: http://www.kb.cert.org/vuls/html/search
- Tenable Network Security: https://www.tenable.com/cve
- Tipping Point Zero Day Initiative: https://www.zerodayinitiative.com/advisories
- Offensive Security Exploit Database: https://www.exploit-db.com/
- Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities
- Fortinet FortiGuard Services: https://www.fortiguard.com/psirt

Type 1 Hypothesis searches were conducted on January 31, 2023 and included the following search terms:

- Each FortiGate hardware and virtual model.
- FortiOS 6.4.9
- Each Processor and Crypto Accelerator used by the TOE.
- OpenSSL 1.1.1q
- OpenSSH 7.1

- TLS
- IPSec
- Fortinet Entropy Token
- Araneus USB TRNG hardware token
- Araneus Alea
- Apache 2.4.41
- Firewall
- TCP, UDP, IPv4, IPv6

The evaluation team identified two applicable vulnerabilities CVE-2022-42472 and CVE-2022-39948. Neither vulnerability affects evaluated functionality, and the vendor plans to patch them by May 15, 2023. The evaluation team determined that no other residual vulnerabilities exist based on these searches that are exploitable by attackers with Basic Attack Potential.

RSA key transport attacks are the only type-2 hypotheses identified for the NDcPP. The TOE does not support RSA key transport.

The evaluation team developed Type 3 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

The evaluation team developed Type 4 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

7 Evaluation Activities for Stateful Traffic Filter Firewalls PP-Module

7.1 Security Audit (FAU)

7.1.1 FAU_GEN.1 Audit data generation (MOD CPP FW)

7.1.1.1 TSS

463 No additional Evaluation Activities are specified.

7.1.1.2 Guidance Documentation

464 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.

Findings:	[FNLOG] All events and their format are in the Log Reference document. Samples are provided of the expected audit messages.

7.1.1.3 Tests

465 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.

High-Level Test Description
Ensure that the TOE displays an audit record for each Firewall Auditable Event.
Findings: PASS – The evaluator performed the testing in conjunction with the testing of the security mechanisms directly. The evaluator confirmed that the TOE correctly generates audit records for the firewall auditable events listed in the table of audit events and administrative actions.

7.2 User Data Protection (FDP)

7.2.1 FDP_RIP.2 Full Residual Information Protection (MOD CPP FW)

7.2.1.1 TSS

466 “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.

Findings:	[ST] Section 6.6 states that no information from previously processed information flows is transferred to subsequent information flows. The TSS describes how the removal of residual information is done through the zeroization of data when the memory structure is initially created and strict bounds checking on the data prior to it being assigned in memory.

7.3 Firewall (FFW)

7.3.1 FFW_RUL_EXT.1 Stateful Traffic Filtering (MOD CPP FW)

467 The following table provides an overview about execution of test cases regarding IPv4 and IPv6.

SFR Element/Test Case	Test execution
FFW_RUL_EXT.1, Tests 1-2	Both, IPv4 and IPv6.
FFW_RUL_EXT.1.2/1.3/1.4, Tests 1-2	As defined in the test description.
FFW_RUL_EXT.1.5, Tests 1-8	Both, IPv4 and IPv6.
FFW_RUL_EXT.1.6, Tests 1-2	Both IPv4 and IPv6 shall be tested for items a), b), c), d), and e) of the SFR element FFW_RUL_EXT.1.6. 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.
FFW_RUL_EXT.1.7, Tests 1-2	Both, IPv4 and IPv6.
FFW_RUL_EXT.1.8, Tests 1-2	Both, IPv4 and IPv6.
FFW_RUL_EXT.1.9, Test 1	As defined in the test description.
FFW_RUL_EXT.1.10, Tests 1	Both, IPv4 and IPv6.

7.3.1.1 TSS

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

Findings:	[ST] Section 6.13 states that the TOE provides stateful packet filtering policies and network packet processing is done on each packet that arrives on an interface. The TSS provides a detailed description on how packets cannot flow during this process.

469 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.

Findings:	[ST] Section 6.13 states all received network packets are processed by the TOE policy engine. The policy engine does stateful filtering of the received network packets according to the configured firewall policies. The TOE kernel monitors the state of any running processes, including the policy engine, VPN processes and IPS processes.

	The network interfaces of the TOE remain down until the self-tests have passed and all processes are up and running. The failure of any of the self-tests during operation results in the network interfaces being downed and all traffic blocked. During operation, if any of the processes fail or terminate unexpectedly, the kernel will block traffic - i.e. the TOE fails closed. The TOE also implements a conserve mode as a self-protection measure if a memory shortage occurs. Conserve mode activates protection measures in order to recover memory space such as throttling traffic. In extreme cases conserve mode will cause any new connection requests to be dropped. When sufficient memory is recovered to resume normal operation, the TOE exits conserve mode state and releases the protection measures.

7.3.1.2 Guidance Documentation

470 The guidance documentation associated with this requirement is assessed in the subsequent test assurance activities.

7.3.1.3 Tests

471 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.

High-Level Test Description
Create a rule to deny ICMP traffic passing through the TOE. Initiate continuous ICMP Pings while rebooting the TOE. Verify no ICMP pings are forward through the TOE.
Findings: PASS – The evaluator confirmed the TOE does not permit network traffic while the TOE is being initialized.

472 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.

High-Level Test Description
Create a rule to allow ICMP traffic to pass through the TOE. Initiate continuous ICMP Pings while rebooting the TOE. Verify no ICMP pings are forward through the TOE while the TOE is being initialized.
Findings: PASS – The evaluator confirmed the TOE does not permit network traffic while the TOE is being initialized.

473 Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test assurance activities.

7.3.2 FFW_RUL_EXT.1.2/FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4 (MOD CPP FW)

7.3.2.1 TSS

474 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
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source address
  - Destination Address
  - Transport Layer Protocol
- IPv6
  - Source address
  - Destination Address
  - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
  - Source Port
  - Destination Port
- UDP
  - Source Port
  - Destination Port

475 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.

Findings:	[ST] Section 6.13 states the TOE permits the configuration of stateful packet filtering for all associated protocols and allow rules to be configured to permit or drop traffic. The TSS states “each rule can be tied to a specific interface,” “each packet that arrives on an interface is subject to the enforcement of stateful traffic filtering,” and “all received network packets are processed by the TOE policy engine.” Interface types does not affect the TOE’s processing of packet filtering rules.

7.3.2.2 Guidance Documentation

476 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
  - Type
  - Code
- ICMPv6
  - Type
  - Code
- IPv4
  - Source address
  - Destination Address

    - Transport Layer Protocol
- IPv6
    - Source address
    - Destination Address
    - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
    - Source Port
    - Destination Port
- UDP
    - Source Port
    - Destination Port

477 The evaluator shall verify that the guidance documentation indicates that each rule can identify the following actions: permit, drop, and log.

Findings:	The [CLI] document in the “config firewall service custom” starting on page 150 describes the process by which each of the protocol properties can be configured for use in the firewall policy table. Once the object is configured, specifying the action is described under “Policy and Objects > Policies” in the [ADMIN] document starting on page 1050 and in [CLI] section “config firewall policy” starting on page 309. Policies can be set to “ACCEPT” or “DENY”. Independently, policies can be set to log the traffic and optionally capture specific packets associated with the rule.

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

Findings:	The [ADMIN] section “Policy and Objects > Policies” starting on page 1050, firewall rules are associated with specific incoming and outcoming network interfaces.

7.3.2.3 Tests

479 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
    - Type
    - Code
- ICMPv6
    - Type
    - Code
- IPv4
    - Source address
    - Destination Address
    - Transport Layer Protocol
- IPv6
    - Source address
    - Destination Address
    - Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields
- TCP
    - Source Port
    - Destination Port
- UDP
    - Source Port
    - Destination Port

High-Level Test Description

Configure firewall rules that filter based on the following criteria:
• ICMPv4
  ○ type
  ○ code
• ICMPv6
  ○ type
  ○ code
• IPv4
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
• IPv6
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
  ○ Extensions
• TCP
  ○ Source port
  ○ Destination port
• UDP
  ○ Source port
  ○ Destination port
• Interface

Findings: PASS – This test is performed as part of FFW_RUL_EXT.1.9 where the firewall rules are first configured (satisfying this test), then verified to perform the specified action(s).

480 Test 2: Repeat the test assurance activity above to ensure that stateful traffic filtering rules can be defined for each distinct network interface type supported by the TOE.

High-Level Test Description
Section 6.13 of the [ST] indicates rules apply to and are assigned to specific interfaces, so interface type does affect how stateful traffic filtering operates. There are no additional interface types to test.
Findings: N/A

481 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.

7.3.3 FFW_RUL_EXT.1.5 (MOD CPP FW)

7.3.3.1 TSS

482 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.

Findings:	[ST] Section 6.13 identifies all the protocols that support stateful session handling.

483 The evaluator shall verify that the TSS describes how stateful sessions are established (including handshake processing) and maintained.

Findings:	[ST] Section 6.13 describes how stateful sessions are established and maintained.

484 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.

Findings:	[ST] Section 6.13 identifies and describes the attributes in session determination for TCP.

485 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.

Findings:	[ST] Section 6.13 identifies and describes the attributes in session determination for UDP.

486 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.

Findings:	[ST] Section 6.13 identifies and describes the attributes in session determination for ICMP.

487 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).

Findings:	[ST] Section 6.13 states that connections are removed based on timeout conditions.

7.3.3.2 Guidance Documentation

488 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.

Findings:	The [LoP] document describes the flow of packet data as it enters a physical interface on the TOE. The description throughout provides an extensive view of how the stateful nature of established sessions is handled. It includes a discussion of hardware accelerated capabilities vs. non-accelerated behaviours. Diagrams and flow charts are provided to give the reader an understanding of the process. Stateful sessions are described at a high-level in section “Packet flow ingress and egress: FortiGates without network processor offloading” in [LoP]. Specific functionality included in the stateful inspection are described starting in section “Kernel” in [LoP]. A summary and review of stateful inspection components are included in section “Comparison of inspection types” in [LoP].

7.3.3.3 Tests

489 The following tests shall be run using IPv4 and IPv6.

490 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.

High-Level Test Description
Begin a TCP 3-way handshake and set packets with flags that are not consistent with the 3-way handshake (i.e., PSH and URG) before the handshake has completed. Verify the PSH and URG packets are not accepted as part of the TCP session. Complete the TCP 3-way handshake and send packets that do not match the session based on differing source address, differing destination address, differing source port, differing destination port, a sequence number outside the window, and flags not consistent with the session (i.e., SYN). Verify these packets are not accepted as part of the session.
Findings: PASS – The evaluator confirmed the TOE does not allow TCP packets that are in an invalid TCP state or that are similar to but do not match an established TCP session.

491 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.

High-Level Test Description
Perform a valid TCP 3-way handshake to establish a session and then terminate the session. Send a packet matching the former session. Verify the packet matching the former session is not forwarded.
Findings: PASS – The evaluator confirmed the TOE does not permit/forward packets that match a terminated session.

492 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.

High-Level Test Description
Establish a TCP connection through the TOE. Wait for the TOE to expire the TCP session due to inactivity (i.e., no packets for 1 hour). Send packets that match the session and verify the TOE does not accept them as part of the previous session.
Findings: PASS – The evaluator confirmed the TOE does not permit/forward packets that match a TCP session that has timed out.

493 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.

High-Level Test Description
Establish a UDP session through the TOE. Send packets that do not match the session based on differing source address, differing destination address, differing source port, and differing destination port. Verify the packets are not accepted as part of the original session.
Findings: PASS – The evaluator confirmed the TOE does not permit UDP packets as part of an established session when the UDP packets are similar but different than the established session.

494 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.

High-Level Test Description
Establish a UDP session through the TOE. Wait for the TOE to expire the UDP session due to inactivity (i.e., no packets for 180 seconds). Send packets that match the session and verify the TOE does not accept them as part of the previous session.
Findings: PASS – The evaluator confirmed the TOE does not permit UDP packets as part of a previous session after the original session has timed out.

495 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.

High-Level Test Description
Establish an ICMP session through the TOE. Send packets that do not match the session based on differing source address, differing destination address, differing type, and differing code. Verify the packets are not accepted as part of the original session.
Findings: PASS – The evaluator confirmed the TOE does not permit ICMP packets as part of an established session when the packets are similar but different than the established session.

496 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.

High-Level Test Description
The only termination of ICMP sessions described in the TSS is through timeout. Timeout of ICMP session is tested in Test 8.
Findings: N/A

497 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.

High-Level Test Description
Establish an ICMP session through the TOE. Wait for the ICMP session to timeout. Send a packet that matches the former session. Verify the TOE does not accept the packet part of the original session (e.g., it logs the packet as a new ICMP session).
Findings: PASS – The evaluator confirmed the TOE does not permit ICMP packets as part of a previous session after the original session has timed out.

7.3.4 FFW_RUL_EXT.1.6 (MOD CPP FW)

7.3.4.1 TSS

498 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)

Findings: [ST] Section 6.12 states the TOE will automatically drop the following packets and an audit log generated for each event:

a) Packets which are invalid fragments (see below);

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) 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) 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) Packets where the source address is equal to the address of the network interface where the network packet was received;

j) Packets where the source or destination address of the network packet is a linklocal address; and

k) Packets where the source address does not belong to the networks associated with the network interface where the network packet was received - the TOE implements Reverse Path Forwarding (RPF), also called Anti Spoofing. This prevents an IP packet from being forwarded if its source IP address either does not belong to a locally attached subnet (local interface), or be a hop on the routing between the TOE and another source (static route, RIP, OSPF, BGP).

7.3.4.2 Guidance Documentation

499 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.

Findings: The [SUPP] document under "Miscellaneous Logging" describes the types of events and packets for which logging is enabled by default without configuration. Specifically, this is "dropped ICMP packets, dropped invalid IP packets". The [SUPP] under "Miscellaneous administration related changes" specifies configuring the default drop rules. Additional logging is configurable as described in the [CLI] under "log > config log setting" section starting on page 487.

7.3.4.3 Tests

500 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.

501 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.

High-Level Test Description

Send packets from the WAN to the LAN with the following characteristics:

a) IP fragments that are not valid
b) IP fragmented packets which cannot be re-assembled completely
c) IP packets where the source address is a broadcast address (xxx.xxx.xxx.255)
d) IP packets where the source address is a multicast address (224.0.0.0/24 or f ff08::/8)
e) IP packets where the source address is a loopback address (127.0.0.0/8 or ::1/128)
f) IP packets where the source or destination address is unspecified (0.0.0.0) and "reserved for future use" (240.0.0.0/4)
g) IP packets where the source or destination address is unspecified (::) and "reserved for future definition and use" (i.e. unicast addresses not in this address range: 2000::/3)

High-Level Test Description
h) IP packets with the Loose Source Routing, Strict Source Routing, and Record Route options Verify these packets are dropped and logged.
Findings: PASS – The evaluator confirmed that firewall rules can be configured based on each characteristic and the TOE drops packets matching each characteristic.

502 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).

Note	The logging and review of logs are done in the previous test case.

7.3.5 FFW_RUL_EXT.1.7 (MOD CPP FW)

7.3.5.1 TSS

503 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.

Findings:	See FFW_RUL_EXT.1.6

7.3.5.2 Guidance Documentation

504 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.

Findings:	The “strict-src-check” is enabled as part of the evaluated configuration to enable strict source verification. Further information is provided in [ADMIN] section “Reverse path look-up” (page 532). [SUPP] section “Additional settings” provides the settings that are required to maintain CC compliance. The [CLI] document describes the process by which each of the protocol properties can be configured for use in the firewall policy table in section “config firewall service custom” starting on page 151. The [SUPP] document under “Miscellaneous Logging” describes the types of events and packets for which logging is enabled by default without configuration. Specifically, this is “dropped ICMP packets, dropped invalid IP packets”. Additional logging is configurable as described in the [CLI] under “log > config log setting” section starting on page 487.

505 The following tests shall be run using IPv4 and IPv6.

506 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.

High-Level Test Description
Send packets from the WAN to the LAN where the source address belongs to the TOE's WAN interface. Verify the packets are dropped and logged.
Findings: PASS – The evaluator confirmed the TOE dropped packets when the source address was the same as the TOE interface that received the packets.

507 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.

High-Level Test Description
Send packets from the WAN to the LAN where the source address belongs to LAN subnet. Verify the packets are dropped and logged.
Findings: PASS – The evaluator confirmed the TOE dropped packets when the source address was not an address the TOE believes is routable from the interface that received the packet.

7.3.6 FFW_RUL_EXT.1.8 (MOD CPP FW)

7.3.6.1 TSS

NIAP TD0545

508 If the TOE implements a mechanism that ensures that no conflicting rules can be configured, the TSS shall describe the underlying mechanism.

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

Findings:	[ST] Section 6.13 states 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

7.3.6.2 Guidance Documentation

510 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.

Findings:	The [ADMIN] document in the "Firewall policy parameters" section starting on page 1051) describes the order of policies. The [ADMIN] describes the commands

	necessary to adjust the precedence with the move command in the “Getting started > Using the CLI > Subcommands” starting on page 36.

7.3.6.3 Tests

NIAP TD0545

511 Test 1: 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.

High-Level Test Description
Configure two rules, one that allows traffic to pass from the WAN to the LAN, and a second that denies traffic from the WAN to the LAN. With the allow rule ordered before the deny rule, attempt a connection from the WAN to the LAN. Verify the connection succeeds. With the deny rule ordered before the allow rule, attempt a connection from the WAN to the LAN. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE processes firewall rules in the administrator configured order by observing that traffic was allowed or denied based on the order of the rules.

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

High-Level Test Description
Configure two rules, one that allows traffic to pass from the WAN to the LAN, and a second (more specific rule) that denies traffic from the WAN VM to the LAN VM. With the allow rule ordered before the deny rule, attempt a connection from the WAN to the LAN. Verify the connection succeeds. With the deny rule ordered before the allow rule, attempt a connection from the WAN to the LAN. Verify the connection fails.
Findings: PASS – The evaluator confirmed the TOE processes firewall rules in the administrator configured order by observing that traffic was allowed or denied based on the order of the rules.

7.3.7 FFW_RUL_EXT.1.9 (MOD CPP FW)

7.3.7.1 TSS

513 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).

Findings:	[ST] Section 6.13 states if no matching rule is found, the TOE will automatically deny the packets and generate a log entry accordingly.

7.3.7.2 Guidance Documentation

514 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.

Findings:	The [ADMIN] document in section “Firewall policy parameters” starting on page 1051 describes that packets are denied by default. This behaviour is not configurable.

7.3.7.3 Tests

515 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. 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.

High-Level Test Description

Configure firewall rules that filter based on the following criteria:
• ICMPv4
  ○ type
  ○ code
• ICMPv6
  ○ type
  ○ code
• IPv4
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
• IPv6
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
  ○ Extensions
• TCP
  ○ Source port
  ○ Destination port
• UDP
  ○ Source port
  ○ Destination port
• Interface
Send traffic matching the criteria and verify the configured action (allow or deny) is performed.

Findings: PASS – The evaluator confirmed firewall rules for each criteria could be configured on the TOE and the TOE filtered traffic based on the configured criteria.

7.3.8 FFW_RUL_EXT.1.10 (MOD CPP FW)

7.3.8.1 TSS

516 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).

Findings:	[ST] Section 6.3 states the TOE maintains half-open TCP sessions in the same manner as full TCP sessions. Once the administrator-defined limit for total sessions is met, sessions (both valid and half-open) are automatically closed based on their timeout value.

7.3.8.2 Guidance Documentation

517 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.

Findings:	As per the [ST], the TOE does not differentiate (out of the box) between maximum half-open TCP sessions and maximum total TCP sessions. The [ADMIN] document in section “DoS Protection” starting on page 1118 and [CLI] in section “config firewall DoS-policy{6}" starting on page 360 both describe how a denial of service (DoS) policy can be established to limit the number of concurrent open TCP sessions by limiting the “tcp_dst_session” parameter in a DoS policy to the appropriate amount.

7.3.8.3 Tests

518 The following tests shall be run using IPv4 and IPv6.

519 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.

High-Level Test Description
Send TCP SYN packets from random IPv4 and IPv6 addresses through the TOE. Verify the TOE drops and logs SYNs when the TCP connection limit is exceeded (the TOE treats half-open and fully open connections the same).
Findings: PASS – The evaluator confirmed the TOE limits the number of half-open TCP connections.

7.4 Security management (FMT)

7.4.1 FMT_SMF.1/FFW Specification of Management Functions (MOD CPP FW)

520 The evaluation activities specified for FMT_SMF.1 in the Supporting Document for the Base-PP shall be applied in the same way to the newly added management functions defined in FMT_SMF.1/FFW in the FW Module.

8 Evaluation Activities for SARs defined in the Stateful Traffic Filter Firewalls PP-Module

521 No additional Evaluation Activities for SARs (over and above those in [SDND]) are defined here. The evaluator shall perform the SAR Evaluation Activities defined in the NDcPP Supporting Document against the entire TOE (i.e. both the network device portion and the stateful firewall portion).

522 The evaluator shall also supplement the AVA_VAN.1 Evaluation Activities with the materials provided in Appendix A of the current document.

8.1.1 Vulnerability Survey (AVA_VAN.1)

Note	[CPP_FW_MODv1.4e-SD] does not define any specific Evaluation Activities, rather it provides some guidance specific search terms that must be included while performing the [NDcPP-SD] AVA_VAN.1 Evaluation Activities. The results are captured in section 6.6.1.2.

9 Evaluation Activities for NDcPP modified by VPN Gateway PP-Module

9.1 Security Audit (FAU)

9.1.1 FAU_GEN.1 Audit data generation (MOD VPNGW)

9.1.1.1 TSS

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

Findings:	Addressed by TSS assurance activities for FPF_RUL_EXT.1.

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

Findings:	[ST] Section 6.1 states the TOE drops packets and attempt

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

Findings:	[ST] Section 5.3.1 lists all auditable events and the TSS in section 6.1 refers to that section for the list of auditable events.

9.1.1.2 Operational Guidance

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

Findings:	[CLI] section “config firewall policy” starting from page 309 shows the configuration to log traffic for a specific policy. Option to record logs when a session start is also available for configuration. Additional logging is configurable as described in the [CLI] under “log > config log setting” section starting on page 487.

9.1.1.3 Test

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

528 Test 1: The evaluator shall attempt to flood the TOE with network packets such that the TOE will be unable to process all the packets. This may require the evaluator to

configure the TOE to limit the bandwidth the TOE is capable to handling (e.g., use of a 10 MB interface). The evaluator shall then review the audit logs to verify that the TOE correctly records that it is unable to process all of the received packets and verify that the TOE logging behavior is consistent with the TSS.

High-Level Test Description
Configure the TOE to limit packet throughput to 10Mbps. Send data through the TOE at greater than 10Mbps and verify the TOE logs that packets were dropped.
Findings: PASS – The evaluator confirmed the TOE logs when it is unable to process all of the packets it received.

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

High-Level Test Description
This test is conducted as a part of FCS_IPSEC_EXT.1.6.
Findings: PASS – The evaluator confirmed the TOE logs the establishment of an IPsec session.

9.2 Cryptographic Support (FCS)

9.2.1 FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption) (MOD VPNGW)

530 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to require the ST author to make certain selections, but these selections are all part of the original definition of the SFR so no new behavior is defined by the PP-Module.

9.2.2 FCS_IPSEC_EXT.1 IPsec Protocol (MOD VPNGW)

531 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to require the ST author to make certain selections, but these selections are all part of the original definition of the SFR so no new behavior is defined by the PP-Module.

9.3 Identification and Authentication (FIA)

9.3.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation (MOD VPNGW)

532 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.

9.3.2 FIA_X509_EXT.2 X.509 Certificate Authentication (MOD VPNGW)

533 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.

9.3.3 FIA_X509_EXT.3 X.509 Certificate Requests (MOD VPNGW)

534 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.

9.4 Security management (FMT)

9.4.1 FMT_MTD.1/CryptoKeys Management of TSF Data (MOD VPNGW)

535 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 and to state that it applies specifically to the keys and certificates used for VPN operation. The evaluator shall perform the Evaluation Activities as written for this SFR as applicable to the VPN cryptographic data.

9.4.2 FMT_SMF.1 Specification of Management Functions (MOD VPNGW)

536 There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to require the ST author to make certain selections, but these selections are all part of the original definition of the SFR so no new behavior is defined by the PP-Module.

9.5 Protection of the TSF (FPT)

9.5.1 FPT_TST_EXT.1 TSF Testing (MOD VPNGW)

537 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.

9.5.2 FPT_TUD_EXT.1 Trusted Update (MOD VPNGW)

538 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.

10 Evaluation Activities for VPN Gateway PP-Module

10.1 Cryptographic Support (FCS)

10.1.1 FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication) (MOD VPNGW)

10.1.1.1 TSS

539 The evaluator shall check to ensure that the TSS describes how the key-pairs are generated.

Findings:	[ST] Section 6.2.2 states IKE RSA and IKE ECDSA keys are generated via CSR or directly imported.

540 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;

Findings:	[ST] Section 6.2 lists the information in table 20. The TOE complies with all “shall” and “should” and does not implement “shall not” or “should not” statements. Table 20 includes a description of the “should” statements.

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

Findings:	No such specific extensions have been claimed; no alternative implementations are claimed.

10.1.1.2 Operational Guidance

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

Findings:	The [ADMIN] document in the “System > Certificates” starting on page 1012 provides information on how to generate a CSR necessary to authenticate to the VPN peer. Same section also describes the key type, key size and curve name (if key type is elliptic curve) selections. Further, section “Site-to-site VPN with digital certificate” of the [ADMIN] document (page 1455) describes the process for configuring the IKEv1 and IKEv2 to use the TOE’s certificate.

	In addition to on-board CSR generation, the TOE is capable of importing certificate pairs from the environment. The process is described in the “System > Certificates” section of the [ADMIN] document starting at page 1012 for both web-based GUI and CLI. As per [SUPP] section “Miscellaneous”, RSA and ECDSA keys that are generated during a Certificate Signing Request operation are stored on the boot device of the FortiGate.

10.1.1.3 Test

For FFC Schemes using “safe-prime” groups:

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

For all other selections:

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

Findings:	The vendor uses the CAVP certificates A2269, A2298, A2240, A2241, and A2242 for RSA. The vendor uses the CAVP certificates A2269 and A2298 for ECDSA. This is described in [ST] Table 24.

10.2 Security management (FMT)

10.2.1 FMT_SMF.1/VPN Specification of Management Functions (VPN) (MOD VPNGW)

10.2.1.1 TSS

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

Findings:	[ST] Section 6.9 lists all management functions provided by the TOE. The TSS identifies the logical interfaces used to perform those functions.

10.2.1.2 Operational Guidance

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

Findings:	Section “Policy and Objects > Policies > Firewall policy parameters” of the [ADMIN] document starting on page 1051 describes stateful firewalls in general and how the TOE implements the required functionality. The section describes the TOE’s firewall policies, the applicable configurable rule attributes, actions, how to enable logging, how to assign policies to interfaces and how to ensure they are ordered correctly.

10.2.1.3 Test

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

Note:	The management functions in FMT_SMF.1.1/VPN are exercised by other SFRs. The creation of rules and enforcing ordering is as part of MOD_cPP_FW FFW_RUL_EXT.1.8 Tests 1 and 2. The assignment of rules to interfaces is tested as part of MOD_cPP_FW FFW_RUL_EXT.1.9.

10.3 Packet Filtering (FPF)

10.3.1 FPF_RUL_EXT.1 Rules for Packet Filtering (MOD VPNGW)

10.3.1.1 FPF_RUL_EXT.1.1

10.3.1.1.1 TSS

548 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.

Findings:	[ST] Section 6.10.1 provides a description of the TOE's initialization/startup process. The TSS describes when the firewall rules are being loaded and when the TOE allows traffic to flow through its interfaces.

549 The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Findings:	[ST] Section 6.13 provides an overview of the processing flow and how abnormal circumstances result in the TOE fails closed to a secure state.

10.3.1.1.2 Operational Guidance

550 The operational guidance associated with this requirement is assessed in the subsequent test assurance activities.

10.3.1.1.3 Test

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

High-Level Test Description
Create a rule to deny ICMP traffic passing through the TOE. Initiate continuous ICMP Pings while rebooting the TOE. Verify no ICMP pings are forward through the TOE.

High-Level Test Description
Findings: PASS – This test is satisfied by FFW_RUL_EXT.1.1 Test 1.

552 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.

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

High-Level Test Description
Create a rule to allow ICMP traffic to pass through the TOE. Initiate continuous ICMP Pings while rebooting the TOE. Verify no ICMP pings are forward through the TOE while the TOE is being initialized.
Findings: PASS – This test is satisfied by FFW_RUL_EXT.1.1 Test 2.

10.3.1.2 FPF_RUL_EXT.1.2

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

10.3.1.3 FPF_RUL_EXT.1.3

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

10.3.1.4 FPF_RUL_EXT.1.4

10.3.1.4.1 TSS

556 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)
  - Source address
  - Destination Address
  - Protocol
- IPv6 (RFC 2460)
  - Source Address
  - Destination Address
  - Next Header (Protocol)
- TCP (RFC 793)
  - Source Port
  - Destination Port
- UDP (RFC 768)
  - Source Port
  - Destination Port

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

Findings:	[ST] Section 6.13 claims conformance to RFC 791, 2460, 793 and 768. The TSS states that Compliance testing is performed as part of the development and release process with changes being made as required to ensure conformance.

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

Findings:	[ST] Section 6.13 states that rules can be configured to permit or drop traffic and generate audit logs for each action.

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

Findings:	[ST] Section 6.13 states that rules can be tied to a specific interface and how each packet is processed on the interface.

10.3.1.4.2 Operational Guidance

560 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)
  - Source address
  - Destination Address
  - Protocol
- IPv6 (RFC 2460)
  - Source Address
  - Destination Address
  - Next Header (Protocol)
- TCP (RFC 793)
  - Source Port
  - Destination Port
- UDP (RFC 768)
  - Source Port
  - Destination Port

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

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

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

Findings:	The [CLI] document in the “firewall > config firewall service custom” starting on page 150 describes the process by which each of the protocol properties can be configured

	for use in the firewall policy table. Once the object is configured, specifying the action is described under “Policy and Objects > Policies” in the [ADMIN] document starting on page 1050 and in [CLI] section “config firewall policy” starting on page 309. Policies can be set to “ACCEPT” or “DENY”. Independently, policies can be set to log the traffic and optionally capture specific packets associated with the rule. In the “Policy and Objects > Policies” section in the [ADMIN] document starting on page 1050, firewall rules are associated with specific incoming and outcoming network interfaces.

10.3.1.4.3 Tests

564 The evaluator shall perform the following tests:

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

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

High-Level Test Description

Configure firewall rules that filter based on the following criteria:
• IPv4
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
• IPv6
  ○ Source address
  ○ Destination address
  ○ Transport layer protocol
• TCP
  ○ Source port
  ○ Destination port
• UDP
  ○ Source port
  ○ Destination port

Findings: PASS – This test is a subset of FFW_RUL_EXT.1.2/FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4 Test 1, so it is satisfied by FFW_RUL_EXT.1.2/FFW_RUL_EXT.1.3/FFW_RUL_EXT.1.4 Test 1.

566 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.

High-Level Test Description
Section 6.13 of the [ST] indicates rules apply to and are assigned to specific interfaces, so interface type does affect how stateful traffic filtering operates. There are no additional interface types to test.
Findings: N/A

567 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.

10.3.1.5 FPF_RUL_EXT.1.5

10.3.1.5.1 TSS

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

Findings:	[ST] Section 6.13 states that packet rules are enforced in the order defined by the administrator.

10.3.1.5.2 Operational Guidance

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

Findings:	The [ADMIN] document describes the order of Packet filtering rules in the “Firewall policy parameters” section starting on page 1051. The [ADMIN] document describes the commands necessary to adjust the precedence with move command in the “Getting started > Using the CLI > Subcommands” starting on page 36.

10.3.1.5.3 Test

570 The evaluator shall perform the following tests:

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

High-Level Test Description
Configure two rules, one that allows traffic to pass from the WAN to the LAN, and a second that denies traffic from the WAN to the LAN. With the allow rule ordered before the deny rule, attempt a connection from the WAN to the LAN. Verify the connection succeeds. With the deny rule ordered before the allow rule, attempt a connection from the WAN to the LAN. Verify the connection fails.
Findings: PASS – This test is satisfied by FFW_RUL_EXT.1.8 Test 1.

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

High-Level Test Description
Configure two rules, one that allows traffic to pass from the WAN to the LAN, and a second (more specific rule) that denies traffic from the WAN VM to the LAN VM. With the allow rule ordered before the deny rule, attempt a connection from the WAN to the LAN. Verify the connection succeeds. With the deny rule ordered before the allow rule, attempt a connection from the WAN to the LAN. Verify the connection fails.
Findings: PASS – This test is satisfied by FFW_RUL_EXT.1.8 Test 2.

10.3.1.6 FPF_RUL_EXT.1.6

10.3.1.6.1 TSS

NIAP TD0597

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

Findings:	[ST] Section 6.13 describes the process for applying traffic filtering. The TSS states the TOE automatically deny any packet that don't match any rule and generate a log.

10.3.1.6.2 Operational Guidance

NIAP TD0597

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

Findings:	The [ADMIN] document in the "Policy and Objects > Policies" section starting on page 1050 describes that packets are denied by default. This behaviour is not configurable. The [CLI] document in the "firewall > config firewall service custom" starting on page 150 shows the protocol number range supported.

10.3.1.6.3 Tests

575 The evaluator shall perform the following tests:

NIAP TD0597

576 Test 1: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported

IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit and log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE permits and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE can permit and log traffic based on each IPv4 Transport Layer Protocol.

NIAP TD0597

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

High-Level Test Description
Configure the TOE to deny and log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE can deny and log traffic based on each IPv4 Transport Layer Protocol.

NIAP TD0597

578 Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit/log and deny/log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport

High-Level Test Description
layer protocol but not matching any of the source/destination address combinations and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE denies IPv4 traffic when it does not match any of the configured rules.

NIAP TD0597

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

High-Level Test Description
Configure the TOE to permit and log each supported IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE permits and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE permits and logs each supported IPv6 Transport Layer Protocol and that the TOE denies UDP-Lite which is not supported.

NIAP TD0597

580 Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
Configure the TOE to deny and log each supported IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE denies and logs each supported IPv6 Transport Layer Protocol and that the TOE denies UDP-Lite which is not supported.

NIAP TD0597

581 Test 6: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address

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

High-Level Test Description
Configure the TOE to permit/log and deny/log each supported IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol but not matching any of the source/destination address combinations and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed the TOE denies IPv6 traffic when it does not match any configured rules.

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

High-Level Test Description
Configure the TOE to permit a specific TCP source and destination port combination. Send traffic matching the configured source and destination port combination and verify the TOE permits and logs the traffic.
Findings: PASS – Filtering based on a selected TCP source port and a selected TCP destination port is performed as part of the FFW_RUL_EXT.1.9 TCP test. The evaluator confirmed the TOE can permit and log traffic based on a TCP source and destination port combination.

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

High-Level Test Description
Configure the TOE to deny a specific TCP source and destination port combination. Send traffic matching the configured source and destination port combination and verify the TOE denies and logs the traffic.
Findings: PASS – Filtering based on a selected TCP source port and a selected TCP destination port is performed as part of the FFW_RUL_EXT.1.9 TCP test. The evaluator confirmed the TOE can deny and log traffic based on a TCP source and destination port combination.

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

High-Level Test Description
Configure the TOE to permit a specific UDP source and destination port combination. Send traffic matching the configured source and destination port combination and verify the TOE permits and logs the traffic.
Findings: PASS – A selected source port and a selected destination port are tested in the UDP testing of FFW_RUL_EXT.1.9. The evaluator confirmed the TOE can permit and log traffic based on a UDP source and destination port combination.

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

High-Level Test Description
Configure the TOE to deny a specific TCP source and destination port combination. Send traffic matching the configured source and destination port combination and verify the TOE denies and logs the traffic.
Findings: PASS – A selected source port and a selected destination port are tested in the UDP testing of FFW_RUL_EXT.1.9. The evaluator confirmed the TOE can deny and log traffic based on a UDP source and destination port combination.

586 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:

Protocol	**Defined Attributes**
IPv4	• Transport Layer Protocol 1 - Internet Control Message • Transport Layer Protocol 2 - Internet Group Management • Transport Layer Protocol 3 - Gateway-to-Gateway • Transport Layer Protocol 4 - IP in IP (encapsulation) • Transport Layer Protocol 5 - Stream • Transport Layer Protocol 6 - Transmission Control • Transport Layer Protocol 7 - UCL • Transport Layer Protocol 8 - Exterior Gateway Protocol • Transport Layer Protocol 9 - any private interior gateway • Transport Layer Protocol 10 - BBN RCC Monitoring • Transport Layer Protocol 11 - Network Voice Protocol • Transport Layer Protocol 12 - PUP • Transport Layer Protocol 13 - ARGUS • Transport Layer Protocol 14 - EMCON • Transport Layer Protocol 15 - Cross Net Debugger

	• Transport Layer Protocol 16 - Chaos • Transport Layer Protocol 17 - User Datagram • Transport Layer Protocol 18 - Multiplexing • Transport Layer Protocol 19 - DCN Measurement Subsystems • Transport Layer Protocol 20 - Host Monitoring • Transport Layer Protocol 21 - Packet Radio Measurement • Transport Layer Protocol 22 - XEROX NS IDP • Transport Layer Protocol 23 - Trunk-1 • Transport Layer Protocol 24 - Trunk-2 • Transport Layer Protocol 25 - Leaf-1 • Transport Layer Protocol 26 - Leaf-2 • Transport Layer Protocol 27 - Reliable Data Protocol • Transport Layer Protocol 28 - Internet Reliable Transaction • Transport Layer Protocol 29 - ISO Transport Protocol Class 4 • Transport Layer Protocol 30 - Bulk Data Transfer Protocol • Transport Layer Protocol 31 - MFE Network Services Protocol • Transport Layer Protocol 32 - MERIT Internodal Protocol • Transport Layer Protocol 33 - Sequential Exchange Protocol • Transport Layer Protocol 34 - Third Party Connect Protocol • Transport Layer Protocol 35 - Inter-Domain Policy Routing Protocol • Transport Layer Protocol 36 - XTP • Transport Layer Protocol 37 - Datagram Delivery Protocol • Transport Layer Protocol 38 - IDPR Control Message Transport Protocol • Transport Layer Protocol 39 - TP++ Transport Protocol • Transport Layer Protocol 40 - IL Transport Protocol • Transport Layer Protocol 41 - Simple Internet Protocol • Transport Layer Protocol 42 - Source Demand Routing Protocol • Transport Layer Protocol 43 - SIP Source Route • Transport Layer Protocol 44 - SIP Fragment • Transport Layer Protocol 45 - Inter-Domain Routing Protocol • Transport Layer Protocol 46 - Reservation Protocol • Transport Layer Protocol 47 - General Routing Encapsulation • Transport Layer Protocol 48 - Mobile Host Routing Protocol • Transport Layer Protocol 49 - BNA • Transport Layer Protocol 50 - SIPP Encap Security Payload • Transport Layer Protocol 51 - SIPP Authentication Header • Transport Layer Protocol 52 - Integrated Net Layer Security TUBA • Transport Layer Protocol 53 - IP with Encryption • Transport Layer Protocol 54 - NBMA Next Hop Resolution Protocol • Transport Layer Protocol 61 - Any host internal protocol • Transport Layer Protocol 62 - CFTP • Transport Layer Protocol 63 - Any local network • Transport Layer Protocol 64 - SATNET and Backroom EXPAK • Transport Layer Protocol 65 - Kryptolan • Transport Layer Protocol 66 - MIT Remote Virtual Disk Protocol • Transport Layer Protocol 67 - Internet Pluribus Packet Core • Transport Layer Protocol 68 - any distributed file system • Transport Layer Protocol 69 - SATNET Monitoring • Transport Layer Protocol 70 - VISA Protocol • Transport Layer Protocol 71 - Internet Packet Core Utility

	• Transport Layer Protocol 72 - Computer Protocol Network Executive • Transport Layer Protocol 73 - Computer Protocol Heart Beat • Transport Layer Protocol 74 - Wang Span Network • Transport Layer Protocol 75 - Packet Video Protocol • Transport Layer Protocol 76 - Backroom SATNET Monitoring • Transport Layer Protocol 77 - SUN ND PROTOCOL-Temporary • Transport Layer Protocol 78 - WIDEBAND Monitoring • Transport Layer Protocol 79 - WIDEBAND EXPAK • Transport Layer Protocol 80 - ISO Internet Protocol • Transport Layer Protocol 81 - VMTP • Transport Layer Protocol 82 - SECURE-VMTP • Transport Layer Protocol 83 - VINES • Transport Layer Protocol 84 - TTP • Transport Layer Protocol 85 - NSFNET-IGP • Transport Layer Protocol 86 - Dissimilar Gateway Protocol • Transport Layer Protocol 87 - TCF • Transport Layer Protocol 88 - IGRP • Transport Layer Protocol 89 - OSPFIGP • Transport Layer Protocol 90 - Sprite RPC Protocol • Transport Layer Protocol 91 - Locus Address Resolution Protocol • Transport Layer Protocol 92 - Multicast Transport Protocol • Transport Layer Protocol 93 - AX.25 Frames • Transport Layer Protocol 94 - IP-within-IP Encapsulation Protocol • Transport Layer Protocol 95 - Mobile Internetworking Control Protocol • Transport Layer Protocol 96 - Semaphore Communications Security Protocol • Transport Layer Protocol 97 - Ethernet-within-IP Encapsulation • Transport Layer Protocol 98 - Encapsulation Header • Transport Layer Protocol 99 - Any private encryption scheme • Transport Layer Protocol 100 - GMTP
IPv6	• Transport Layer Protocol 1 - Internet Control Message • Transport Layer Protocol 2 - Internet Group Management • Transport Layer Protocol 3 - Gateway-to-Gateway • Transport Layer Protocol 4 - IPv4 encapsulation • Transport Layer Protocol 5 - Stream • Transport Layer Protocol 6 - Transmission Control • Transport Layer Protocol 7 - CBT • Transport Layer Protocol 8 - Exterior Gateway Protocol • Transport Layer Protocol 9 - any private interior gateway • Transport Layer Protocol 10 - BBN RCC Monitoring • Transport Layer Protocol 11 - Network Voice Protocol • Transport Layer Protocol 12 - PUP • Transport Layer Protocol 13 - ARGUS • Transport Layer Protocol 14 - EMCON • Transport Layer Protocol 15 - Cross Net Debugger • Transport Layer Protocol 16 - Chaos • Transport Layer Protocol 17 - User Datagram • Transport Layer Protocol 18 - Multiplexing • Transport Layer Protocol 19 - DCN Measurement Subsystems • Transport Layer Protocol 20 - Host Monitoring • Transport Layer Protocol 21 - Packet Radio Measurement

	• Transport Layer Protocol 22 - XEROX NS IDP • Transport Layer Protocol 23 - Trunk-1 • Transport Layer Protocol 24 - Trunk-2 • Transport Layer Protocol 25 - Leaf-1 • Transport Layer Protocol 26 - Leaf-2 • Transport Layer Protocol 27 - Reliable Data Protocol • Transport Layer Protocol 28 - Internet Reliable Transaction • Transport Layer Protocol 29 - Transport Protocol Class 4 • Transport Layer Protocol 30 - Bulk Data Transfer Protocol • Transport Layer Protocol 31 - MFE Network Services Protocol • Transport Layer Protocol 32 - MERIT Internodal Protocol • Transport Layer Protocol 33 - Datagram Congestion Control Protocol • Transport Layer Protocol 34 - Third Party Connect Protocol • Transport Layer Protocol 35 - Inter-Domain Policy Routing Protocol • Transport Layer Protocol 36 - XTP • Transport Layer Protocol 37 - Datagram Delivery Protocol • Transport Layer Protocol 38 - IDPR Control Message Transport Protocol • Transport Layer Protocol 39 - TP++ Transport Protocol • Transport Layer Protocol 40 - IL Transport Protocol • Transport Layer Protocol 41 - IPv6 encapsulation • Transport Layer Protocol 42 - Source Demand Routing Protocol • Transport Layer Protocol 43 - Intentionally blank • Transport Layer Protocol 44 - Intentionally blank • Transport Layer Protocol 45 - Inter-Domain Routing Protocol • Transport Layer Protocol 46 - Reservation Protocol • Transport Layer Protocol 47 - General Routing Encapsulation • Transport Layer Protocol 48 - Dynamic Source Routing Protocol • Transport Layer Protocol 49 - BNA • Transport Layer Protocol 50 - Intentionally Blank • Transport Layer Protocol 51 - Intentionally Blank • Transport Layer Protocol 52 - Integrated Net Layer Security • Transport Layer Protocol 53 - IP with Encryption • Transport Layer Protocol 54 - NBMA Address Resolution Protocol • Transport Layer Protocol 55 - Mobility • Transport Layer Protocol 56 - Transport Layer Security Protocol using Kryptonet key management • Transport Layer Protocol 57 - SKIP • Transport Layer Protocol 58 - ICMP for IPv6 • Transport Layer Protocol 59 - No Next Header for IPv6 • Transport Layer Protocol 60 - Intentionally Blank • Transport Layer Protocol 61 - any host internal protocol • Transport Layer Protocol 62 - CFTP • Transport Layer Protocol 63 - any local network • Transport Layer Protocol 64 - SATNET and Backroom EXPAK • Transport Layer Protocol 65 - Kryptolan • Transport Layer Protocol 66 - MIT Remote Virtual Disk Protocol • Transport Layer Protocol 67 - Internet Pluribus Packet Core • Transport Layer Protocol 68 - any distributed file system • Transport Layer Protocol 69 - SATNET Monitoring • Transport Layer Protocol 70 - VISA Protocol

	• Transport Layer Protocol 71 - Internet Packet Core Utility • Transport Layer Protocol 72 - Computer Protocol Network Executive • Transport Layer Protocol 73 - Computer Protocol Heart Beat • Transport Layer Protocol 74 - Wang Span Network • Transport Layer Protocol 75 - Packet Video Protocol • Transport Layer Protocol 76 - Backroom SATNET Monitoring • Transport Layer Protocol 77 - SUN ND PROTOCOL-Temporary • Transport Layer Protocol 78 - WIDEBAND Monitoring • Transport Layer Protocol 79 - WIDEBAND EXPAK • Transport Layer Protocol 80 - ISO Internet Protocol • Transport Layer Protocol 81 - VMTP • Transport Layer Protocol 82 - SECURE-VMTP • Transport Layer Protocol 83 - VINES • Transport Layer Protocol 84 - TTP • Transport Layer Protocol 85 - Internet Protocol Traffic Manager • Transport Layer Protocol 86 - NSFNET-IGP • Transport Layer Protocol 87 - Dissimilar Gateway Protocol • Transport Layer Protocol 88 - TCF • Transport Layer Protocol 89 - EIGRP • Transport Layer Protocol 90 - OSPFIGP • Transport Layer Protocol 91 - Sprite RPC Protocol • Transport Layer Protocol 92 - Locus Address Resolution Protocol • Transport Layer Protocol 93 - Multicast Transport Protocol • Transport Layer Protocol 94 - AX.25 Frames • Transport Layer Protocol 95 - IP-within-IP Encapsulation Protocol • Transport Layer Protocol 96 - Mobile Internetworking Control Pro. • Transport Layer Protocol 97 - Semaphore Communications Sec. Pro. • Transport Layer Protocol 98 - Ethernet-within-IP Encapsulation • Transport Layer Protocol 99 - Encapsulation Header • Transport Layer Protocol 100 - GMTP • Transport Layer Protocol 101 - Ipsilon Flow Management Protocol • Transport Layer Protocol 102 - PNNI over IP • Transport Layer Protocol 103 - Protocol Independent Multicast • Transport Layer Protocol 104 - ARIS • Transport Layer Protocol 105 - SCPS Transport Layer Protocol • Transport Layer Protocol 106 – QNX • Transport Layer Protocol 107 - Active Networks • Transport Layer Protocol 108 - Payload Compression Protocol • Transport Layer Protocol 109 - Sitara Networks Protocol • Transport Layer Protocol 110 - Compaq Peer Protocol • Transport Layer Protocol 111 - IPX in IP • Transport Layer Protocol 112 - Virtual Router Redundancy Protocol • Transport Layer Protocol 113 - PGM Reliable Transport Protocol • Transport Layer Protocol 114 - any 0-hop protocol • Transport Layer Protocol 115 - Layer Two Tunneling Protocol • Transport Layer Protocol 116 - D-II Data Exchange (DDX)

	• Transport Layer Protocol 117 - Interactive Agent Transfer Protocol • Transport Layer Protocol 118 - Schedule Transfer Protocol • Transport Layer Protocol 119 - SpectraLink Radio Protocol • Transport Layer Protocol 120 - UTI • Transport Layer Protocol 121 - Simple Message Protocol • Transport Layer Protocol 122 - SM • Transport Layer Protocol 123 - Performance Transparency Protocol • Transport Layer Protocol 124 - ISIS over IPv4 • Transport Layer Protocol 125 - FIRE • Transport Layer Protocol 126 - Combat Radio Transport Protocol • Transport Layer Protocol 127 - Combat Radio User Datagram • Transport Layer Protocol 128 - SSCOPMCE • Transport Layer Protocol 129 - IPLT • Transport Layer Protocol 130 - Secure Packet Shield • Transport Layer Protocol 131 - Private IP Encapsulation within IP • Transport Layer Protocol 132 - Stream Control Transmission Protocol • Transport Layer Protocol 133 - Fibre Channel • Transport Layer Protocol 134 - RSVP-E2E-IGNORE • Transport Layer Protocol 135 - Mobility Header • Transport Layer Protocol 136 - UDPLite • Transport Layer Protocol 137 - MPLS-in-IP • Transport Layer Protocol 138 - MANET Protocols • Transport Layer Protocol 139 - Host Identity Protocol • Transport Layer Protocol 140 - Shim6 Protocol • Transport Layer Protocol 141 - Wrapped Encapsulating Security Payload • Transport Layer Protocol 142 - Robust Header Compression

10.4 Protection of the TSF (FPT)

10.4.1 FPT_FLS.1/SelfTest Fail Secure (Self-test Failures) (MOD VPNGW)

10.4.1.1 TSS

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

Findings:	[ST] Section 6.10 states the cryptographic functionality and any operation of the TOE supported by this functionality will not be available if the cryptographic tests fail. The TOE will not complete bootup if the CPU, BIOS tests or boot loader image verification fail. If the noise source tests fail, the boot operation will fail and not be completed.

10.4.1.2 Operational Guidance

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

Findings:	The [SUPP] describes the FIPS Error Mode which can occur and how to resolve the issue if encountered. FIPS Error Mode can occur on bootup in response to failed KATs which run at startup. In addition, the [SUPP] also describes in sections “Installing the CC Certified Firmware > Potential Firmware issues” and “Installing the CC Certified Firmware > Potential hardware issues” errors that may occur as a result of the BIOS, hardware or firmware being corrupted. Information is provided on how to get support for these advanced topics. Finally, if the entropy seeding mechanism is unable to gather enough entropy, the [SUPP] describes ways in which this can be troubleshooted in the “Entropy” section.

10.4.1.3 Test

589 There are no test Evaluation Activities for this SFR.

10.4.2 FPT_TST_EXT.3 Self-Test with Defined Methods (MOD VPNGW)

10.4.2.1 TSS

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

Findings:	[ST] Section 6.10 states the TOE performs FIPS 140-2 KATs upon initialization. The initialization process also includes bootstrap, boot loader, verification of the kernel, firmware and software images. The TSS states that the KATs include a comparison of a number of cryptographic functions against an expected set of values.

10.4.2.2 Operational Guidance

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

10.4.2.3 Test

592 There are no test Evaluation Activities for this SFR.

10.5 Trusted Path/Channels (FTP)

10.5.1 FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications) (MOD VPNGW)

10.5.1.1 TSS

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

10.5.1.2 Operational Guidance

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

10.5.1.3 Test

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

11 Evaluation Activities for Selection-Based Requirements defined in the VPN Gateway PP-Module

11.1 Identification and Authentication (FIA)

11.1.1 FIA_PSK_EXT.1 Pre-Shared Key Composition (MOD VPNGW)

11.1.1.1 TSS

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

Findings:	[ST] Section 6.7 states the TOE allows both text-based and bit-based pre-shared keys and supports 6 to 128 characters. Pre-shared keys are conditioned using SHA-1or the PRF that is configured as the hash algorithm. This is consistent with the last selection in the FIA_PSK_EXT.1.3 requirement.

11.1.1.2 Operational Guidance

597 The evaluator shall examine the operational guidance to determine that it provides guidance to administrators on the composition of strong text-based pre-shared keys, and (if the selection indicates keys of various lengths can be entered) that it provides information on the merits of shorter or longer pre-shared keys. The guidance must specify the allowable characters for pre-shared keys, and that list must be a super-set of the list contained in FIA_PSK_EXT.1.2.

Findings:	The [SUPP] section “VPN specific certificate settings > Preshared keys” provides guidance to administrators on the composition of strong text-based pre-shared keys, specifies the allowable lengths for pre-shared keys, and specifies the allowable characters for pre-shared keys. The evaluator confirmed the list of allowable characters is the same as the list given in FIA_PSK_EXT.1.2.

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

Findings:	The TOE supports entering bit-based pre-shared keys only. The [CLI] document describes that bit-based PSKs are entered by using a leading “0x” indicator to type out hexadecimal-based keys in the “config vpn ipsec phase1-interface” section for the “psksecret” value (page 1300)

11.1.1.3 Test

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

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

High-Level Test Description
Modify the pre-shared key to be 22 characters in length. Verify that the IPsec connection can be established using the 22 character PSK.
Findings: PASS - The evaluator confirmed the TOE can establish an IPsec connection using a 22 character PSK.

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

High-Level Test Description
For each key, configure the TOE with edge case length PSKs (minimum and maximum). Verify the TOE successfully establishes an IPsec connection when a valid length is used and fails to establish an IPsec connection when an invalid length is used.
Findings: PASS – The evaluator confirmed the TOE can establish an IPsec connection using the minimum and maximum length PSKs but fails to establish a connection when a longer or shorter PSK is used.

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

High-Level Test Description
Configure the TOE with a bit-based key by entering it in hexadecimal format. Verify that the IPsec connection can be established using the bit-based PSK.
Findings: PASS – The evaluator confirmed the TOE can establish an IPsec connection using a bit-based PSK.

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

High-Level Test Description
The TOE does not generate bit-based keys.
Findings: N/A

12 Evaluation Activities for SARs defined in the VPN Gateway PP-Module

604 To evaluate the SARs specified by NDcPP and this PP-Module, the evaluator shall perform the SAR Evaluation Activities defined in the NDcPP SD against the entire TOE (i.e., both the network device portion and the VPN gateway portion). In particular, the evaluator shall ensure that the vulnerability testing defined in section A.1.4 of the NDcPP SD is applied to the TOE's VPN interface(s) in addition to any other security-relevant network device interfaces that the TOE may have.