www.GossamerSec.com

Assurance Activity Report for Forcepoint NGFW 6.10.9

Version 0.4
04/14/23

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

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

Document: AAR-VID11343

Revision History

Revision	Date	Authors	Summary
Version 0.1	03/06/23	Cummins Catterton	Initial draft
Version 0.2	04/06/23	Cummins	Addressed ECR comments
Version 0.3	04/12/23	Cummins	Addressed ECR comments
Version 0.4	04/14/23	Cummins	Addressed ECR comments

The TOE Evaluation was Sponsored by:

Forcepoint
10900-A Stonelake Blvd.
Austin, TX 78759

Evaluation Personnel:

- Cody Cummins
- Tyler Catterton

Common Criteria Versions:

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

Common Evaluation Methodology Versions:

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

Table of Contents

1. Introduction

This document presents evaluations results of the Forcepoint NGFW 6.10.9 NDcPP22e /STFFW14e/VPNGW12 evaluation. This document contains a description of the assurance activities and associated results as performed by the evaluators.

1.1 Equivalence

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

1.1.1 Evaluated Platform Equivalence

The TOE is Forcepoint Next Generation Firewall (NGFW) 6.10.9 consisting of the following components.

- Forcepoint NGFW Security Management Center (SMC) Virtual Appliance running software version 6.10.9:
  - ESXi 7.0
- Forcepoint NGFW Engine running software version 6.10.9 and including the following models:
  - Desktop models: N60, N120, N120W, N120WL
  - 1U models: 2201, 2205, 2210
  - 2U models: 3401, 3405, 3410
  - Virtual model: ESXi 7.0

The evaluation team performed testing on the following devices all running software version 6.10.9:

- Virtual SMC Appliance on ESXi 7.0 on Intel Xeon Silver 4208 (Cascade Lake)
- N60 Desktop Atom C3338 (Denverton)
- 2210 1U Xeon D-2177NT (Skylake)
- 3401 2U Xeon 4210 (Cascade Lake)
- Virtual NGFW Engine Appliance (VMWare ESXi 7.0 on an Intel Xeon® Silver 4208 (Cascade Lake)

The evaluation team ran the entire test suite on the Virtual SMC Appliance and the N60 Engine. A subset of tests consisting of all Firewall testing, NTP testing, and vulnerability testing was performed on the 2210, 3401 and the Virtual NGFW Engine appliance.

All security engine models share the exact same software image and update images thus the same crypto-library is used on security engine models. Since the same security engine software is installed on all the platforms and all security functions provided by the TOE are implemented in software; the security behavior of the security engines is the same on all models for each of the SFRs in the evaluation.

The models all share the same processor instruction set architecture (x86_64). The security engine models that comprise the TOE have common hardware characteristics. These differing characteristics affect only non-TSF relevant functionality (such as throughput, processing speed, number and type of network connections supported,

number of concurrent connections supported, and amount of storage) and therefore support security equivalency of the security engines in terms of hardware. The firewall testing of several TOE models with different network ports, with identical results further demonstrates the hardware does not have an effect on TSF relevant functionality. As such, complete testing on only the N60 model is sufficient.

1.1.2 CAVP Equivalence

The NGFW Engine contains a hardened Linux operating system (with a 4.19 kernel) executing on a single or multi-processor Forcepoint hardware platform. The Virtual SMC Appliance (or SMC) contains the Management Server and Log Server. Like the NGFW Engine, the SMC contains a hardened Linux-based operating system (which uses a 4.18 kernel) to support the management capabilities and allow for the operation and configuration of firewall engines.

The TOE utilizes cryptographic support features as part of the TLS, IPsec and NTP protocol mechanism as well as to verify software (both TOE updates and installed software). Each component of the TOE utilizes the cryptographic module available to it as identified in the following tables.

Version 6.10.9 of the NGFW Engine internally includes version 10 of the NGFW OS, which itself includes a Linux 4.19 kernel. The table below identifies the CAVP certificates for the NGFW Engine's **Forcepoint NGFW FIPS Cryptographic Module 1.2.1**:

SFR	Algorithm	NIST/ISO Standard	Cert#
FCS_CKM.1 (Key Gen)	RSA IFC Key Generation 2048/3072/4096 bit	FIPS 186-4, RSA	A2155
	ECDSA ECC Key Generation P-256, P-384, P-521	FIPS 186-4, ECDSA	A2155
FCS_CKM.2 (Key Establishment)	RSA-based Key Exchange	Vendor affirm RSAES-PKCS1-v1_5	N/A
	ECC-based Key Exchange P-256, P-384, P-521	SP 800-56A, KAS ECC	A2155
FCS_COP.1/DataEncryption	AES CBC - 128/192/256 AES GCM - 128/256	ISO 10116 ISO 19772	A2155
FCS_COP.1/SigGen	ECDSA Sign/Verify P-256, P-384, P-521	FIPS 186-4, ECDSA	A2155
	RSA Sign/Verify 2048, 3072, 4096	FIPS 186-4, RSA	A2155
FCS_COP.1/Hash	SHA Hashing SHA-1, SHA-256, SHA-384, SHA-512	ISO/IEC 10118-3:2004	A2155
FCS_COP.1/KeyedHash	Keyed Hash HMAC-SHA-1 HMAC-SHA-256, HMAC-SHA-384	ISO/IEC 9797-2:2011	A2155
FCS_RBG_EXT.1 (CTR) (Random)	CTR_DRBG (AES) 256 bits	ISO/IEC 18031:2011	A2155
FCS_IPSEC	IKEv2 KDF	SP 800-135	A2155
FCS_TLS	TLS KDF	SP 800-135	A2155

The table below identifies the CAVP certificates for the NGFW Engine's **Forcepoint NGFW Cryptographic Kernel Module 3.0**:

SFR	Algorithm	NIST/ISO Standard	Cert#
FCS_COP.1/DataEncryption	AES 128/192/256 CBC, GCM	ISO 10116 ISO 19772	A2166
FCS_COP.1/Hash	SHA Hashing SHA-1, SHA-256, SHA-384, SHA-512	ISO/IEC 10118-3:2004	A2166
FCS_COP.1/KeyedHash	Keyed Hash HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	ISO/IEC 9797-2:2011	A2166

The TOE includes the SMC FIPS Java API 1.0.2.3 library which was deemed equivalent to the 1.0.2.1 version of the library as they differ only because of non-security relevant changes. These non-security relevant changes include a fix for JVM class loader issues manifesting in the October 2020 updates to Java 11, fixes to remove two performance bottlenecks, and a change to allow configuring the library's system properties though java.security. These non-security relevant changes were reviewed by a laboratory and the CMVP and the different versions of the library were added to the library's FIPS 140-2 validation certificate.

The table below identifies the CAVP certificates for the Virtual SMC Appliance's **SMC FIPS Java API 1.0.2.3:**

SFR	Algorithm	NIST Standard	Cert#
FCS_CKM.1 (Key Gen)	RSA IFC Key Generation 2048/3072 bit	FIPS 186-4, RSA	A1973
	ECDSA ECC Key Generation P-256, P-384, P-521	FIPS 186-4, ECDSA	A1973
FCS_CKM.2 (Key Establishment)	RSA-based Key Exchange	Vendor affirm RSAES-PKCS1-v1_5	N/A
	ECC-based Key Exchange P-256, P-384, P-521	SP 800-56A, KAS ECC	A1973
FCS_COP.1/DataEncryption	AES 128/256 CBC, GCM	ISO 10116 ISO 19772	A1973
FCS_COP.1/SigGen	RSA Sign/Verify 2048/3072 bit	FIPS 186-4, RSA	A1973
	ECDSA Sign/Verify P-256, P-384, P-521	FIPS 186-4, ECDSA	A1973
FCS_COP.1/Hash	SHA Hashing SHA-1, SHA-256, SHA-384, SHA-512	ISO/IEC 10118=3:2004	A1973
FCS_COP.1/KeyedHash	Keyed Hash HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384	ISO/IEC 9797-2:2011	A1973
FCS_RBG_EXT.1 (CTR, Hash)(Random)	DRBG Bit Generation CTR_DRBG (AES) -256 bits Hash_DRBG (SHA-512)	ISO/IEC 18031:2011	A1973

Version 6.10.9 of the Virtual SMC Appliance internally includes an ESXi 7.0 hypervisor, running a RHEL 8 OS virtual machine. The RHEL 8 OS (running as a virtual machine) includes a Linux 4.18 kernel. The table below identifies the CAVP certificates for the Virtual SMC Appliance's **SMC FIPS Library 1.1.1**:

SFR	Algorithm	NIST Standard	Cert#
FCS_CKM.1 (Key Gen)	ECDSA ECC Key Generation P-256, P-384, P-521	FIPS 186-4, ECDSA	A1972
FCS_CKM.2 (Key Establishment)	ECC-based Key Exchange P-256, P-384, P-521	SP 800-56A, KAS ECC	A1972
FCS_COP.1/DataEncryption	AES 128/256 CBC, GCM	ISO 10116 ISO 19772	A1972
FCS_COP.1/SigGen	ECDSA Sign/Verify P-256, P-384, P-521	FIPS 186-4, ECDSA	A1972
FCS_COP.1/Hash	SHA Hashing SHA-1, SHA-256, SHA-384, SHA-512	ISO/IEC 10118-3:2004	A1972
FCS_COP.1/KeyedHash	Keyed Hash HMAC-SHA-256, HMAC-SHA-384	ISO/IEC 9797-2:2011	A1972
FCS_RBG_EXT.1(CTR) (Random)	DRBG Bit Generation CTR_DRBG (AES)- 256 bits	ISO/IEC 18031:2011	A1972

The table below identifies the CAVP certificates for the Virtual SMC Appliance's **SMC FIPS Cryptographic Module for NTP 3.79**:

SFR	Algorithm	NIST Standard	Cert#
FCS_COP.1/Hash	SHA Hashing, SHA-1	ISO/IEC 10118-3:2004	A3360

All microarchitecture families included in the evaluation have been tested. As such, there is no equivalency argument applicable as all processor microarchitectures and the evaluated OS version are addressed.

1.2 References

The following evidence was used to complete the Assurance Activities:

- Forcepoint NGFW 6.10.9 Security Target, Version 0.5, 04/14/2023 **(ST)**
- Forcepoint Next Generation Firewall 6.10 Common Criteria Evaluated Configuration Guide, Revision C (**Admin Guide**)

2. Protection Profile SFR Assurance Activities

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

2.1 Security audit (FAU)

2.1.1 Audit Data Generation (NDcPP22e:FAU_GEN.1)

2.1.1.1 NDcPP22e:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.1.2 NDcPP22e:FAU_GEN.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.1 (FAU_GEN.2) of the ST states that the TOE records the key name (the string provided by the administrator) as an identifier of any TOE key generated, imported, changed, or deleted.

Section 7 of the ST provides a table which maps the distributed TOE components to SFRS and associated audit events. The evaluator confirmed that all components that generate audit information for a particular SFR are also mapped to that SFR in the Distributed TOE SFR Allocation (SFR to TOE components mapping).

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

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

The "Review audit events" section of the **Admin Guide** provides a list of all auditable events and a description of the fields in each audit record. The audit events are shown in McAfee ESM format. For further details and instructions on how to set the audit record format to use, a reference to the *Forcepoint Next Generation Firewall Product Guide* is provided as detailed below. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 27 (Configuring the Log Server) and Chapter 29 (Reconfiguring the SMC and engines) provide instructions for configuring the TOE to forward audit data in the selected format from the SMC's Log Server and Management Server to an external syslog server as well as how to enable TLS protection for the forwarding of this data.

The "Review audit events" section of the **Admin Guide** provides several tables which list the audit events corresponding with the requirements claimed in the ST and with the associated audit events for those requirements from the NDcPP22e, STFFW14e, and VPNGW12. The evaluator confirmed that this mapping is complete and found examples of each auditable event required by FAU_GEN.1. Additionally, each of the listed events provides the required content matching the FAU_GEN.1 requirement. As part of testing, the evaluator also verified each audit record in the **Admin Guide** also collected.

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

Component Testing Assurance Activities: The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records for the events listed in the table of audit events and administrative actions listed above. This should include all instances of an event: for instance, if there are several different I&A mechanisms for a system, the FIA_UIA_EXT.1 events must be generated for each mechanism. The evaluator shall test that audit records are generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. If HTTPS is implemented, the test demonstrating the establishment and termination of a TLS session can be combined with the test for an HTTPS session. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the guidance documentation, and that the fields in each audit record have the proper entries.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of auditable events to TOE components in the Security Target. For all events involving more than one TOE component when an audit event is triggered, the evaluator has to check that the event has been audited on both sides (e.g. failure of building up a secure communication channel between the two components). This is not limited to error cases but includes also events about successful actions like successful build up/tear down of a secure communication channel between TOE components.

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

The evaluator created a list of the required audit events. The evaluator then collected each audit event when running the other security functional tests described in this AAR as required by the PPs. The evaluator recorded these audit events in the proprietary Detailed Test Report (DTR). The security management events are handled in a similar manner. When the administrator was required to set a value for testing, the audit record associated with the administrator action was collected and recorded in the DTR.

2.1.2 Security Audit Data Generation (STFFW14E:FAU_GEN.1)

2.1.2.1 STFFW14E:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

No additional Evaluation Activities are specified. Please see NDcPP22e:FAU_GEN.1.

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

This activity has been performed in NDcPP22e:FAU_GEN.1. The evaluator verified that all audit records specified in Table 2 and Table 3 of the PP-Module and those required by the Base-PP are described.

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

This activity was performed as part of NDcPP22e:FAU_GEN.1. The evaluator verified that all audit records specified in Table 2 and Table 3 of the PP-Module are generated.

2.1.3 Audit Data Generation (VPN Gateway) (VPNGW12:FAU_GEN.1/VPN)

2.1.3.1 VPNGW12:FAU_GEN.1.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.2 VPNGW12:FAU_GEN.1.2/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

Section 6.1 of the ST states that the TOE uses the same audit mechanisms for VPN as it does for the all other audits. The TOE does not have an audit mechanism used only for VPN gateway functionality.

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

The "Review audit events" section of the AGD includes all security-relevant auditable events described in the ST's audit table. The TOE does not use multiple audit mechanisms.

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

The evaluator enabled the TOE audit capability and performed testing of security features defined in the [ST]. While establishing IPsec sessions for other tests associated with the FCS_IPSEC_EXT and FIA_X509_EXT requirements, the evaluator observed that these tests triggered the TOE to generate the required audit records.

2.1.4 User identity association (NDcPP22e:FAU_GEN.2)

2.1.4.1 NDcPP22e:FAU_GEN.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Please refer to NDcPP22e:FAU_GEN.1

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

Please refer to NDcPP22e:FAU_GEN.1

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

For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one test on one component where another component instigates an auditable event. The evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

This activity was accomplished in conjunction with the testing of FAU_GEN.1.1.

2.1.5 Security Audit Generation (NDcPP22e:FAU_GEN_EXT.1)

2.1.5.1 NDcPP22e:FAU_GEN_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

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

2.1.6.1 NDcPP22e:FAU_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.6.2 NDcPP22e:FAU_STG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.6.3 NDcPP22e:FAU_STG_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit information to an external IT entity can be done in real-time or periodically. In case the TOE does not perform transmission in real-

time the evaluator needs to verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible acceptable frequency for the transfer of audit data.

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

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

Section 6.1 of the ST states that the NGFW engine transfers audit records to the Virtual SMC appliance Log Server immediately after generation of the record. The Virtual SMC Appliance Log Server stores Engine records and can also send those audits to an external syslog server immediately after they have been received. The Virtual SMC appliance Management Server generates audit records, stores the records locally and can also send them to an external syslog server immediately after storing the records. The Virtual SMC forwards its audit records to an external syslog server using TLS protected syslog configured by an administrator. When a connection to the external syslog server fails, the Management server or Log server will re-establish the connection and send new audit records to the syslog server.

When the NGFW Engine cannot transfer newly generated audit records to the Log Server (irrespective of the cause), the Engine will spool the records on disk. Once the NGFW Engine can again transfer audit records to the Log Server, it transfers the oldest records first and removes those from its spool, continuing until it has transferred all spooled records. If the NGFW Engine cannot transfer its audit records to the Log Server for an extended period of time, it may start to exhaust its available spool disk storage space (the size of this spool depends on the size of the disk in the NGFW Engine model, but the N120/N120W/N120WL/N120L/N60/N60L/2201/2205/2210/3401/3405/3410/Virtual models provide 3/3/8/8/3/3/48/48/48/201/201/201/24 GB of spool space, respectively). In such an event the NGFW Engine first sends alerts to notify the administrator that it has nearly exhausted its log spool, and once it exhausts its spool space, it follows the administrator defined behavior.

The Log Server, which aggregates audit records from NGFW Engines, writes incoming audit entries to the SMC Appliance disk storage (the Log Server's audit storage has its own 180GB logical partition in which to store the records). The proprietary protocol for synchronizing and managing the data between the NGFW Engine and the Virtual SMC Appliance Log Server starts with the Engine notifying the Log Server that there is a new log and then sending the new log entry to the Log Server. The Log Server stores the audit information as database files which are only accessible to a TOE administrator via the SMC Management Client. Only after the Log Server confirms successful receipt and storage of an audit entry does an Engine remove the audit entry from its spool.

The Log Server itself has a limited amount of disk storage in which to hold its database audit records. If the Log Server draws close to exhausting this space (specifically if it has fewer than 300MB of space remaining), it will alert

the administrator (by creating an audit alert that the SMC Client displays to the administrator) warning of the low storage remaining (and the administrator can take action to remove old audit records). Should the Log Server continue to fill up its storage space and have less than 100MB of space remaining, it will stop accepting new audit messages from Engines.

The Management Server, like the Log Server, has a finite amount of space to store management audit records (10GB) on the Virtual SMC Appliance's disk. The Management Server stores its logs in a separate partition (the root partition), and should the Management Server begin to exhaust this space, it behaves similarly to the Log Server. When less than 300MB of space remains, the Management Server generates an administrative alert (displayed to the administrator through the SMC Client GUI), and when less than 100MB of space remains, the Management Server will no longer allow the administrator to make configuration changes (as such a change would result in an audit message that the Management Server no longer has sufficient space to safely store) until action is taken by the administrator to delete local audit data.

As mentioned above, the administrator defines the log spool policy for the NGFW Engines. This specifies the behavior of the NGFW Engines whenever its local log spool fills. The TOE requires that the following setting be used when in an evaluated configuration:

- Stop traffic (required in the evaluated configuration): NGFW Engine automatically goes to an offline state and connections going through NGFW Engine are transferred to other nodes in a cluster (if the Engine were part of a cluster). Once the spool situation has improved, the Engine/node returns automatically to an online state.

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

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

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table refers the reader to specific chapters in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 27 (Configuring the Log Server) provides instructions for configuring the TOE to forward audit

data from the Log Server to an external syslog server as well as how to enable TLS protection for the forwarding of this data.

- Chapter 29 (Reconfiguring the SMC and engines) provides instructions for configuring the TOE to forward audit data from the Management Server to an external syslog server as well as how to enable TLS protection for the forwarding of this data.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table also provides specific options that should be selected and defined when following the instructions in the online guide to set the audit data forwarding in the properties of the Management Server or Log Server. This includes specifying which options should be selected for the TLS certificate to use, defining a TLSv1.2 profile element, defining the TLS server identity and configuring the approved set of TLS ciphersuites.

The "Create a Task to delete log and audit data" section of the **Admin Guide** describes the relationship between local audit data and the audit data sent to the external syslog server. The NGFW Engine stores log data temporarily until the data is sent to the Log Server. The Management Server and Log Server store audit and log data locally, then send the data to an external audit server. Locally-stored data is not deleted automatically. This section also provides the steps for the administrator to schedule the deletion of audit data based on date.

The behavior when remaining audit storage space starts to become low is as follows:

- Log Server - When the remaining audit storage space drops below 300MB, an alert is sent to administrators. When less than 100MB of space remains, the Log Server stops accepting new audit messages from NGFW Engines. The administrator has to take action to remove old audit records.

- Management Server - When less than 100MB of audit storage space remains, the Management Server prevents the administrator from making further changes. The administrator has to take action to remove old audit records.

When the Log Spooling Policy option is set to 'Stop Traffic', the NGFW Engine goes offline when the local storage space is full. This can happen when the Log Server is not available or when the Log Server storage space is becoming full and the Log Server stops the log reception.

The "Create an element for the NGFW Engine" section of the **Admin Guide** provides instructions for configuring "Stop Traffic" as the log spooling policy via the Advanced Settings > Log Handling branch. Further details can be found in the 'Creating and modifying NGFW Engines' chapter in the Forcepoint *Next Generation Firewall Product Guide.* This guide can be accessed via the link found in the "Supporting documentation" section.

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

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

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

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

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

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

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

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

The TOE has three separate components that contain log storage locations, the NGFW Engine, the Management Server (running on the SMC Appliance), and the Log Server (also on the SMC Appliance). All audits from the NGFW Engine are forwarded to the Log Server. The Management Server and the Log Server are configured to forward audit logs to an external syslog server.

Test 1: The evaluator configured the TOE to send audit records via a secure TLS connection to an external server running rsyslog 8.16.0. The evaluator tested the TOE's ability to generate and transmit as a part of every test that includes associated audit messages. The evaluator observed via packet captures that the audit data was not sent in the clear. The evaluator also observed that all audit data was successfully received by the audit server.

Test 2 (1, 2, 3): The TOE has three local log storage locations, the NGFW Engine, the Management Server (running on the SMC Appliance), and the Log Server (also on the SMC Appliance). The evaluator generated audit data until the local storage space in each location was exceeded. The evaluator confirmed the following behavior:

- The NGFW Engine transitioned to an offline state and stopped permitting traffic when it reached its local storage capacity.
- The Log Server generated an audit alert regarding storage capacity and then transitioned to a state where it would no longer accept logs from the NGFW Engine.
- The Management Server transitioned to a state where it would no longer process administrator actions that would result in generation of new audit records.

This behavior complies with the behavior defined in FAU_STG_EXT.1.3.

Test 3: NA - the TOE does not claim FAU_STG_EXT.2/LocSpace.

Test 4: Testing for these components was demonstrated in test 1 and test 2 above.

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

2.1.7.1 NDcPP22e:FAU_STG_EXT.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.9 of the ST states that the TOE utilizes TLS protected communications from Engines to their SMC (log forwarding) and from the SMC to its Engines (management). The TOE has no other communications between

components.

Section 6.1 of the ST states that the NGFW engine transfers audit records to the Virtual SMC appliance Log Server immediately after generation of the record. The Virtual SMC Appliance Log Server stores Engine records and can also send those audits to an external syslog server immediately after they have been received. The Virtual SMC appliance Management server generates audit records, stores the records locally and sends them to an external syslog server immediately after storing the records. The Virtual SMC forwards its audit records to an external syslog server using TLS protected syslog configured by an administrator.

When the NGFW Engine cannot transfer newly generated audit records to the Log Server (irrespective of the cause), the Engine will spool the records on disk. Once the NGFW Engine can again transfer audit records to the Log Server, it transfers the oldest records first and removes those from its spool, continuing until it has transferred all spooled records. If the NGFW Engine cannot transfer its audit records to the Log Server for an extended period of time, it may start to exhaust its available spool disk storage space. In such an event the NGFW Engine first sends alerts to notify the administer that it has nearly exhausted its log spool, and once it exhausts its spool space, it will stop traffic and go into an offline state until the spool situation has improved and audit space is once again available.

For further detail regarding audit storage functionality, see TSS assurance activities for FAU_STG_EXT.1.3.

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

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

This activity has been performed in FAU_STG_EXT.1 and FCO_CPC_EXT.1 where the local storage for audit data and the link between the NFGW Engine and the SMC Log Server are described.

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

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

Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It

is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

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

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

Test 1: The evaluator used the functionality provided by the SMC's Client GUI to view the audit logs locally.

Test 2: This test was performed in NDcPP22e:FAU_STG_EXT.1 where results of encrypted audit log transmission to an external audit server were generated.

Test 3: This test was performed as part of NDcPP22e:FPT_ITT.1_t3 where the evaluator confirmed that the TLS handshake in the ITT channel completed successfully, and application data was seen between the SMC and the Engine.

The evaluator found that the TOE behavior during testing is consistent with the descriptions in the TSS and Guidance documentation.

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

2.1.8.1 NDcPP22e:FAU_STG_EXT.5.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

This activity has been performed in FAU_STG_EXT.1 and FAU_STG_EXT.4.

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

This activity has been performed in FAU_STG_EXT.1 and FCO_CPC_EXT.1.

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

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

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

Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the

following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

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

Test 1: This test was performed as part of NDcPP22e:FAU_STG_EXT.4 where the audit logs, including those sent from the Engine to the Log Server (on the SMC) were viewed locally via the SMC's Client GUI.

Test 2: This test was performed as part of NDcPP22e:FAU_STG_EXT.1, test 1 where the Management Server and Log Server components of the Virtual SMC appliance established a successful connection to the external audit log server in order to pass audit events in a secure manner.

Test 3: This test was performed as part of NDcPP22e:FPT_ITT.1_t3 where the evaluator confirmed that the TLS handshake in the ITT channel completed successfully, and application data was seen between the SMC and the Engine.

2.2 Communication (FCO)

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

2.2.1.1 NDcPP22e:FCO_CPC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.2 NDcPP22e:FCO_CPC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.3 NDcPP22e:FCO_CPC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

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

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

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

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

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

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

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

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

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

The evaluator shall examine the TSS to confirm that it:

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

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

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

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

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

Section 6.2 of the ST states that the TOE implements a registration process using a channel that meets the secure registration channel requirements in FTP_TRP.1/Join. The TOE utilizes TLS to provide a registration channel between a new Engine and the SMC. The TOE requires that an administrator first create a new NGFW Engine

object from within the SMC. Upon completion, the SMC generates and provides the administrator with a 45-character "password" (the SMC generates a unique "password" for each and every registration attempt) as well as the SHA-512 hash of the SMC's server TLS certificate. The administrator then inputs this password into the Engine (via console) and initiates the registration connection from the Engine. The Engine confirms that the SMC's TLS certificate SHA-512 hash either matches the administrator pre-configured value (if the administrator chose to enter it) or displays the hash for the administrator to confirm. Then the SMC authenticates the Engine by requesting the Engine send the SHA-512 hash of the SMC-generated password. Once authenticated, the SMC pushes the SMC's internal CA certificate to the Engine, the Engine creates a CSR that the SMC's internal CA uses to issue the Engine a TLS certificate, and the Engine validates the received TLS certificate. SMC then pushes a policy to the Engine. The policy contains the Log Server reference identifier. After completion, the Engine and SMC subsequently communicate through mutually-authenticated TLS connections.

An administrator can disable communication between an NGFW Engine and the SMC via the SMC Client GUI. This is done from the perspective of the NGFW Engine by performing a factory reset of the Engine and from the perspective of the SMC by deleting the NGFW Engine element in the Client GUI. The TOE, by design, restricts communications between components and only permits communications between the SMC and Engines, and does not support direct communication between two Engines.

Section 6.11 of the ST describes FTP_TRP.1/Join which specifies the registration channel used. It states that the SMC Management Server only accepts join requests from Engines for which the Administrator has already created an object and provided the Engine with the SMC generated password. Furthermore, the SMC protects the registration channel using TLSv1.2 and requires that the registering Engine prove knowledge of the SMC generated password by exchanging a SHA-512 hash. The SMC reserves port 3021 for registration and after registration, the components communicate using mutually-authenticated TLS on different ports (8903, 8907, 8906, 8916, 8917, 3020, 3023), thus preventing reuse of the registration channel. Should a registration attempt fail, the administrator can attempt again, or can generate a new password on the SMC (and input that password into the Engine), and then attempt registration again.

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

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

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

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

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

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

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

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

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

Section “Establishing a security configuration” in the **Admin Guide** provides the steps for configuring the SMC appliance and NGFW Engine into the evaluated configuration.

The “Enable FIPS mode on the SMC Appliance” section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC TLS Client and Server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The user is instructed to use the main network interface for management for the connection to the NGFW Engine. For specific instructions, this section refers the reader to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

Section "Create an element for the NGFW Engine" in the **Admin Guide** provides the steps to use the Management Client to create an NGFW Engine element.

Section "Enabling communication between the SMC and NGFW Engine" in the **Admin Guide** states that after the initial contact is made, subsequent TLS connections between components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

The "Save the initial configuration in the Management Client" section in the **Admin Guide** describes how to save the initial NGFW Engine configuration in the Management Client and then select the Firewall Policy that will be automatically installed on the NGFW Engine after initial contact is made. It also instructs the user to make a note of the one-time generated password, the Management server address, and the SHA-512 certificate fingerprint as this information is needed when installing the NGFW Engine.

Section "Install NGFW Engine in FIPS mode" in the **Admin Guide** indicates that FIPS mode and 256 bit encryption must be enabled when the NGFW engine is installed. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS cipher suite is used for the connection between the NGFW Engine and the Management Server. If the initial contact fails, the user is instructed to restart the appliance, start the NGFW Configuration Wizard again, and verify that the following are correct: the one time password, Management server IP address, certificate fingerprint and that 256 bit encryption is used for the connection to the Management server. Further instructions are also provided in the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode

- The "Installing the SMC Appliance in FIPS mode" section provides instructions for selecting FIPS 140-2 mode during installation.
- The "Install the SMC components" section provides instructions for enabling FIPS restrictions on the Management Server, Log Server and Management Client during the installation.
- The "Create an element for the NGFW Engine" section provides the steps for creating an NGFW Engine object in the Management Client. A note here indicates there is a one-time password created that allows establishing trust with the Management Server.

- The “Install the NGFW Engine in FIPS mode” provides the instructions for upgrading the software and setting the kernel in FIPS mode after reboot. Configuration steps after reboot include selecting FIPS compatible operating mode.

Section “Disabling communication between the SMC and NGFW Engine” in the **Admin Guide** provides instructions for disabling communication from the SMC by deleting the NGFW Engine element in the Management Client, and resetting the NGFW Engine to factory settings.

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

The evaluator shall carry out the following tests:

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

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

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

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

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

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

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

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

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

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

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

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

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

Test 1.1 and 1.2: The evaluator verified that communication between the SMC and the NGFW Engine does not exist until the registration process occurs. Before registration can be initiated from the Engine the evaluator had to configure a new engine on the SMC. The evaluator configured the interfaces and routing for the Engine which would be applied during registration. Once the registration was initiated and the registration attempt was successful, communication between the TOE components was enabled.

Test 2: The evaluator attempted to unregister the NGFW engine from the SMC by sending a factory reset to the NGFW Engine via the SMC and then deleting the NGFW Engine resource from the SMC. The evaluator confirmed that the SMC no longer recognized a connection to the NGFW Engine and there was no longer any traffic flowing between them.

Test 3: This testing requirement was met by NDcPP22e:FTP_TRP.1/Join.

Test 4: The evaluator configured the virtual Engine with the same address as the N60 and used the same

registration parameters that were used in FCO_CPC_EXT.1-t1 to join the N60 to the SMC. The evaluator observed that this attempt to reuse the same registration channel was rejected.

Test 5: According to operational guidance, to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. These steps were performed during the registration process in Test 1 above.

2.3 Cryptographic support (FCS)

2.3.1 Cryptographic Key Generation (NDcPP22e:FCS_CKM.1)

2.3.1.1 NDcPP22e:FCS_CKM.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 in the ST includes tables which identify the TOE's cryptographic functions including cryptographic algorithms and associated key sizes. Section 6.3.3 further describes key generation and key sizes supported by the TOE. The ST specifies both the RSA and ECC schemes.

Section 6.3.3 (FCS_CKM.1) indicates that the Virtual SMC Appliance supports asymmetric key generation for key establishment as part of TLS. This section further provides a table detailing which components act as TLS clients and servers as well as which ones generate RSA or ECDH keys used during TLS cipher suite negotiations. The TOE, when in the evaluated configuration, uses 256-bit encryption mode as the security strength. This setting causes the TOE to generate an ECDSA P-521 TLS server certificate. The TOE also provides the administrator the ability to generate or import either an ECDSA [P-256, P-384, or P-521] or RSA [2048, 3072] key to use for TLS Client or mutual authentication during TLS syslog export). This is consistent with the schemes and key sizes specified in the requirement.

Section 6.3.3 of the ST further identifies where each of these schemes is used by mapping them to the associated TLS SFRs and the services they support. The TOE supports asymmetric key generation for key establishment as part of TLS cipher suite negotiation. The TOE supports the use of RSA with 2048 bit key sizes and ECDSA with a key

size of 256, 384, or 521 bit for cryptographic signatures (using NIST curves P-256, P-384, or P-521).

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed.

When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. Management connections are protected by 256-bit encryption. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

How to install Forcepoint NGFW in FIPS mode:

- The "Installing the SMC Appliance in FIPS mode" section describes how to enable FIPS restrictions on the SMC and the Management client during installation. This includes selecting FIPS 140-2 mode and 256-bit encryption in the security selections when installing the SMC Management Server and Log Server.

- The "Install the Management Client" section describes how to access, download and install the management client on a PC in the operating environment and then ensure that it is operating in FIPS mode by selecting the 'Restricted Cryptographic Algorithms Compatible with FIPS 140-2' operating mode.

- The "Installing the NGFW Engine in FIPS mode" describes how to install and configure the NGFW engine and to select 'FIPS-Compatible Operating Mode' during the process of defining the NGFW engine properties.

The “FIPS mode restrictions” section in the **Admin Guide** states that when FIPS mode is enabled, the following restrictions are enforced:

- The NGFW Engine local console, command line interface, and SSH access are not available
- The available cryptographic algorithms and configuration options in the SMC are restricted:
    - RSA key sizes of 2048 bits or greater are used for digital signature generation
    - ECDSA key sizes of 256 bits or greater are used for digital signature generation
    - SHA-1 cannot be used for digital signature generation

The “Install the Management Client” section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on to the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the “Supporting documentation” section.

The “Create an element for the NGFW Engine” section of the **Admin Guide** provides the steps for defining the properties of an NGFW Engine. It includes some specific settings that should be made in the evaluated configuration including choosing FIPS-Compatible Operating Mode.

The “Configure settings for an evaluated configuration” section of the **Admin Guide** provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server and for HTTPS connections for SMC Web Access. This guide is accessed via the link found in the “Supporting documentation” section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

The “Setup VPN Profile Element” section of the **Admin Guide** includes instructions to allow the administrator to configure the specifics of IPsec (IKEv2/ESP) connections including:
In the IKE SA Tab
Cipher Algorithms
■ **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
■ **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.

Message Digest Algorithms
■ **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Diffie-Hellman Groups
- **14 (2048 bits)** — Diffie-Hellman key exchange with a 2048-bit modulus.
- **15 (3072 bits)** — Diffie-Hellman key exchange with a 3072-bit modulus.
- **16 (4096 bits)** — Diffie-Hellman key exchange with a 4096-bit modulus.
- **17 (6144 bits)** — Diffie-Hellman key exchange with a 6144-bit modulus.
- **18 (8192 bits)** — Diffie-Hellman key exchange with a 8192-bit modulus.
- **19 (ECP 256 bits)** — Diffie-Hellman key exchange with 256-bit elliptic curve.
- **20 (ECP 384 bits)** — Diffie-Hellman key exchange with 384-bit elliptic curve.
- **21 (ECP 521 bits)** — Diffie-Hellman key exchange with 521-bit elliptic curve.

Authentication Method
- **RSA Signatures** — Requires that each Gateway has a valid certificate.
- **ECDSA Signatures** — Requires that each Gateway has a valid certificate.

And in the IPsec SA Tab
Cipher Algorithms
- **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
- **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.
- **AES-GCM-128** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 128-bit key size. Recommended for high-speed networks.
- **AES-GCM-256** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 256-bit key size. Recommended for high-speed networks.

Message Digest Algorithms
- **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

In the “Configure settings for an evaluated configuration” section of the **Admin Guide**, the Audit Server Configuration entry in the table also provides further guidance and specific options that should be selected and defined when following the instructions in the online guide to enable TLS v1.2 protection. It outlines the steps for configuring a trusted Root CA and generating a client certificate request which includes selecting an RSA with key size 2048 bits or greater, or ECDSA with 521 bits for P-521, 384 bits for P-384 or 256 bits for P-256. This section also provides the list of approved TLS cipher suites and instructs the user to ensure that the TLS cipher suites selected match the RSA and ECDSA parameters that are configured. When using an ECDHE cipher suite, P-521, P-384 and P-256 are automatically used in the TLS key establishment.

Component Testing Assurance Activities: Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products.

Generation of long-term cryptographic keys (i.e. keys that are not ephemeral keys/session keys) might be performed automatically (e.g. during initial start-up). Testing of key generation must cover not only administrator invoked key generation but also automated key generation (if supported).

Key Generation for FIPS PUB 186-4 RSA Schemes

The evaluator shall verify the implementation of RSA Key Generation by the TOE using the Key Generation test. This test verifies the ability of the TSF to correctly produce values for the key components including the public verification exponent e, the private prime factors p and q, the public modulus n and the calculation of the private signature exponent d.

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

a) Random Primes:

- Provable primes

- Probable primes

b) Primes with Conditions:

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

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

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

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

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

FIPS 186-4 Public Key Verification (PKV) Test

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

Key Generation for Finite-Field Cryptography (FFC)

The evaluator shall verify the implementation of the Parameters Generation and the Key Generation for FFC by the TOE using the Parameter Generation and Key Generation test. This test verifies the ability of the TSF to correctly produce values for the field prime p, the cryptographic prime q (dividing p-1), the cryptographic group generator g, and the calculation of the private key x and public key y.

The Parameter generation specifies 2 ways (or methods) to generate the cryptographic prime q and the field prime p:

- Primes q and p shall both be provable primes

- Primes q and field prime p shall both be probable primes

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process

- Generator g constructed through an unverifiable process.

The Key generation specifies 2 ways to generate the private key x:

- len(q) bit output of RBG where 1 <=x <= q-1

- len(q) + 64 bit output of RBG, followed by a mod q-1 operation and a +1 operation, where 1<= x<=q-1.

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

To test the cryptographic and field prime generation method for the provable primes method and/or the group generator g for a verifiable process, the evaluator must seed the TSF parameter generation routine with sufficient data to deterministically generate the parameter set.

For each key length supported, the evaluator shall have the TSF generate 25 parameter sets and key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing values generated by the TSF with those generated from a known good implementation. Verification must also confirm

- g != 0,1

- q divides p-1

- g^q mod p = 1

- g^x mod p = y

for each FFC parameter set and key pair.

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

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

2.3.2 Cryptographic Key Generation (for IKE Peer Authentication) (VPNGW12:FCS_CKM.1/IKE)

2.3.2.1 VPNGW12:FCS_CKM.1.1/IKE

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

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

Section 6.3.3 of the ST states that the TOE performs RSA key generation as per FIPS 186-4 Appendix B.3.3 (Generation of Random Primes that are Probably Prime) and B.3.6 (Generation of Probably Primes with Conditions Based on Auxiliary Probably Primes) and ECDSA key generation per FIPS 186-4 Appendix B.4.2 (Key Pair Generation by Testing Candidates). The TOE's implementation complies with the appendices, implementing all shall requirements (omitting none) and does not implement any additional ('shall not', 'should', 'should not', nor TOE-specific extensions) functionality.

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

The "Create a certificate request for a VPN Gateway element" in the **Admin Guide** describes how an administrator invokes the TOE's IKE key generation functionality, along with the inputs and outputs associated with the process. That section also describes the output from the generation process and the subsequent input of the CA issued certificate.

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

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

For all other selections:

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

This requirement is met by the TOE and has been CAVP tested. Refer to the CAVP certificates identified in the table "CAVP Analysis" in Section 1.1.2.

2.3.3 Cryptographic Key Establishment (NDcPP22e:FCS_CKM.2)

2.3.3.1 NDcPP22e:FCS_CKM.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

The intent of this activity is to be able to identify the scheme being used by each service. This would mean, for example, one way to document scheme usage could be:

Scheme | SFR | Service

RSA | FCS_TLSS_EXT.1 | Administration

ECDH | FCS_SSHC_EXT.1 | Audit Server

ECDH | FCS_IPSEC_EXT.1 | Authentication Server

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

(TD0580 applied)

The ST specifies RSA, ECC, and FFC schemes in FCS_CKM.1.1. Section 6.3 of the ST contains tables which identify the TOE's cryptographic functions including cryptographic algorithms and associated key sizes.

Section 6.3.3 (FCS_CKM.2) provides a table which maps the supported key establishment schemes (consistent with those identified in FCS_CKM.1.1) to the associated SFRs and their usage (ie. Service).

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

This activity has been performed in FCS_CKM.1.

Component Testing Assurance Activities: Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

The evaluator shall verify a TOE's implementation of SP800-56A key agreement schemes using the following Function and Validity tests. These validation tests for each key agreement scheme verify that a TOE has implemented the components of the key agreement scheme according to the specifications in the Recommendation. These components include the calculation of the DLC primitives (the shared secret value Z) and the calculation of the derived keying material (DKM) via the Key Derivation Function (KDF). If key confirmation is supported, the evaluator shall also verify that the components of key confirmation have been implemented correctly, using the test procedures described below. This includes the parsing of the DKM, the generation of MACdata and the calculation of MACtag.

Function Test

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

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

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

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

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

Validity Test

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

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

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

RSA-based key establishment

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

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2. For RSA and FFC safe-primes, the TOE was tested with a known good implementation.

2.3.4 Cryptographic Key Destruction (NDcPP22e:FCS_CKM.4)

2.3.4.1 NDcPP22e:FCS_CKM.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Note that where selections involve 'destruction of reference' (for volatile memory) or 'invocation of an interface' (for non-volatile memory) then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory the evaluator

includes in their examination the relevant interface description for each media type on which plaintext keys are stored. The presence of OS-level and storage device-level swap and cache files is not examined in the current version of the Evaluation Activity.

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

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

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

Table 12 in Section 6.3.3 of the ST identifies the Crypto Security Parameters (CSPs), secret keys and private keys provided by the TOE. The table also identifies where each CSP or key is stored and how it is cleared. Keys stored on disk are zeroized upon command by overwriting with pseudo-random data. Keys stored in memory are zeroized when a handshake is complete or at the close of a session by overwriting with zeros.

Section 6.3.3 (FCS_CKM.4) of the ST states that the TOE components clear keys (TLS and IPsec) from memory after those keys are no longer needed. After use on the SMC keys are overwritten with zeros and garbage collector is called. This is performed by the SMC Java Code and SMC FIPS Java API. After use on the NGFW Engine, keys are overwritten with zeros. This is performed by the NGFW FIPS Cryptographic Module.

The TOE uses file system calls to clear persistently stored keys. On the SMC, TLS and IPsec private keys are stored in a Java Keystore. To clear these keys the disk must be wiped. This can be done via the installer by selecting the “Secure wipe with Automatic install”. Data is sourced from /dev/random and then written to the disk. This is done 3 times for the whole disk.

On the NGFW Engine TLS and IPsec private keys are stored in a flat file. To clear these keys the disk must be wiped. This can be done by resetting to factory defaults and choosing a number of overwrites. Data is sourced from /dev/random and then written to disk.

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

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command [Where TRIM is used then the TSS and/or guidance documentation is also expected to describe how the keys are stored such that they are not inaccessible to TRIM, (e.g. they would need not to be contained in a file less than 982 bytes which would be completely contained in the master file table)] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

The ST and the **Admin Guide** do not identify any circumstances or configurations that do not strictly conform to the key destruction requirements.

Component Testing Assurance Activities: None Defined

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

2.3.5.1 NDcPP22e:FCS_COP.1.1/DataEncryption

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 in the ST includes four tables which identify the TOE's cryptographic functions including cryptographic algorithms, modes and key sizes.

Section 6.3.3 (FCS_COP.1/DataEncryption) in the ST states that the TOE performs encryption and decryption using AES in either CBC or GCM mode, and key sizes of either 128 or 256.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

The "Configure settings for an evaluated configuration" section of the **Admin Guide** provides a reference to specific sections of the Forcepoint Next Generation Firewall Product Guide which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server and for HTTPS connections for SMC Web Access. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table provides instructions for configuring the TLS profile and identifies the supported TLS ciphersuites which use AES in CBC or GCM mode and key sizes of either 128 or 256. Similarly, the SMC Web Access entry in the table provides instructions for creating a TLS cryptography suite set and identifies the supported TLS ciphersuites which use AES 256 bit encryption in CBC or GCM mode.

The "Setup VPN Profile Element" section of the **Admin Guide** includes instructions to allow the administrator to configure the specifics of IPsec (IKEv2/ESP) connections including:
In the IKE SA Tab
Cipher Algorithms
■ **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
■ **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key

size.

Message Digest Algorithms
■ **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Diffie-Hellman Groups
■ **14 (2048 bits)** — Diffie-Hellman key exchange with a 2048-bit modulus.
■ **15 (3072 bits)** — Diffie-Hellman key exchange with a 3072-bit modulus.
■ **16 (4096 bits)** — Diffie-Hellman key exchange with a 4096-bit modulus.
■ **17 (6144 bits)** — Diffie-Hellman key exchange with a 6144-bit modulus.
■ **18 (8192 bits)** — Diffie-Hellman key exchange with a 8192-bit modulus.
■ **19 (ECP 256 bits)** — Diffie-Hellman key exchange with 256-bit elliptic curve.
■ **20 (ECP 384 bits)** — Diffie-Hellman key exchange with 384-bit elliptic curve.
■ **21 (ECP 521 bits)** — Diffie-Hellman key exchange with 521-bit elliptic curve.

Authentication Method
■ **RSA Signatures** — Requires that each Gateway has a valid certificate.
■ **ECDSA Signatures** — Requires that each Gateway has a valid certificate.

And in the IPsec SA Tab
Cipher Algorithms
■ **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
■ **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.
■ **AES-GCM-128** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 128-bit key size. Recommended for high-speed networks.
■ **AES-GCM-256** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 256-bit key size. Recommended for high-speed networks.

Message Digest Algorithms
■ **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Component Testing Assurance Activities: AES-CBC Known Answer Tests

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

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

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

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

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

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

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

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

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

AES-CBC Multi-Block Message Test

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

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

AES-CBC Monte Carlo Tests

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

Input: PT, IV, Key

for i = 1 to 1000:

if i == 1:

CT[1] = AES-CBC-Encrypt(Key, IV, PT)

PT = IV

else:

CT[i] = AES-CBC-Encrypt(Key, PT)

PT = CT[i-1]

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

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

AES-GCM Test

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

128 bit and 256 bit keys

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

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

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

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

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

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

AES-CTR Known Answer Tests

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

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

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

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

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

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

AES-CTR Multi-Block Message Test

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

AES-CTR Monte-Carlo Test

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

Input: PT, Key

for i = 1 to 1000:

CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i]

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

There is no need to test the decryption engine.

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

2.3.6 Cryptographic Operation (AES Data Encryption/Decryption) (VPNGW12:FCS_COP.1/DataEncryption)

2.3.6.1 VPNGW12:FCS_COP.1.1/DataEncryption

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to 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.

See NDcPP22e:FCS_COP.1.1

Component Guidance Assurance Activities: None Defined **Component Testing Assurance Activities**: None Defined

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

2.3.7.1 NDcPP22e:FCS_COP.1.1/Hash

TSS Assurance Activities: None Defined **Guidance Assurance Activities**: None Defined **Testing Assurance Activities**: None Defined
Component TSS Assurance Activities: The evaluator shall check that the association of the hash function with other TSF cryptographic functions (for example, the digital signature verification function) is documented in the TSS.

Section 6.3.3 of the ST states that the TOE supports cryptographic hashing services using SHA-1, SHA-256, SHA-384 and SHA-512. The TOE uses HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384 for keyed-hash message authentication. Keyed hashing is used for the following purposes with these key sizes:

- TLS 1.2 master secret 384 bits,
- RSA premaster secret 384 bits,
- ECDHE premaster secret sizes for 256, 384 and 521 bits for P-256, P-384 and P-521, respectively (note that in TLS_ECDHE_* cipher suites, the TOE offers all three NIST curves and will select based upon what the peer specifies).
- TOE integrity check (the Virtual SMC Appliance checks its file system integrity using HMAC-SHA-256 using a hardcoded 256-bit key), and
- TLS 1.2 will use HMAC-SHA1, HMAC-SHA-256 and HMAC-SHA-384 with a 160/256/384 bit key, respectively.

The TOE provides the ability to synchronize its time with an NTP server and protects its time data with a SHA1 message digest.

Section 6.2 in the ST states that the TOE uses a SHA-512 hash of the SMC's server TLS certificate during the registration process between the SMC and the Engine.

Section 6.3.2 in the ST states that the Virtual SMC appliance uses /dev/random to instantiate its SHA-512 HASH_DRBG and generate keys. The Management Server also generates salts for password hashing and SHA-512 hashes user passwords both when creating a new user or when verifying the password of an existing user.

Section 6.9 in the ST states that Management Server passwords are salted and hashed using SHA-512 when stored. The SMC verifies ECDSA P-521 with SHA-512 signatures on update packages for both the SMC and the NGFW Engine when they are uploaded to the SMC. The NGFW Engine similarly verifies the signature on the update package when it is transferred from to the Engine from the SMC.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC TLS Client and Server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

The "Install the Management Client" section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the "Supporting documentation" section.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table provides instructions for configuring the TLS profile and identifies the supported

TLS ciphersuites for communication between the TOE and an external syslog server. Similarly, the SMC Web Access entry in the table provides instructions for creating a TLS cryptography suite set and identifies the supported TLS ciphersuites for HTTPS connections for SMC Web Access. These approved sets of ciphersuites as defined in the ST and the **Admin Guide** include the use of SHA-1, SHA256 and SHA384. This section also provides a reference to the *Forcepoint Next Generation Firewall Product Guide* which provides detailed instructions for configuring and enabling TLS protection for communications between the TOE and an external syslog server and for HTTPS connections for SMC Web Access. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

The "Setup VPN Profile Element" section of the **Admin Guide** includes instructions to allow the administrator to configure the specifics of IPsec (IKEv2/ESP) connections including:
In the IKE SA Tab
Cipher Algorithms
■ **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
■ **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.

Message Digest Algorithms
■ **SHA-2** — The SHA-2 set of hash functions including SHA-256, SHA-384, and SHA-512

Diffie-Hellman Groups
■ **14 (2048 bits)** — Diffie-Hellman key exchange with a 2048-bit modulus.
■ **15 (3072 bits)** — Diffie-Hellman key exchange with a 3072-bit modulus.
■ **16 (4096 bits)** — Diffie-Hellman key exchange with a 4096-bit modulus.
■ **17 (6144 bits)** — Diffie-Hellman key exchange with a 6144-bit modulus.
■ **18 (8192 bits)** — Diffie-Hellman key exchange with a 8192-bit modulus.
■ **19 (ECP 256 bits)** — Diffie-Hellman key exchange with 256-bit elliptic curve.
■ **20 (ECP 384 bits)** — Diffie-Hellman key exchange with 384-bit elliptic curve.
■ **21 (ECP 521 bits)** — Diffie-Hellman key exchange with 521-bit elliptic curve.

Authentication Method
■ **RSA Signatures** — Requires that each Gateway has a valid certificate.
■ **ECDSA Signatures** — Requires that each Gateway has a valid certificate.

And in the IPsec SA Tab
Cipher Algorithms
■ **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
■ **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key

size.
■ **AES-GCM-128** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 128-bit key size. Recommended for high-speed networks.
■ **AES-GCM-256** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 256-bit key size. Recommended for high-speed networks.

Message Digest Algorithms
■ **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Component Testing Assurance Activities: The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

The evaluator shall perform all of the following tests for each hash algorithm implemented by the TSF and used to satisfy the requirements of this PP.

Short Messages Test - Bit-oriented Mode

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

Short Messages Test - Byte-oriented Mode

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

Selected Long Messages Test - Bit-oriented Mode

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

Selected Long Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m + 8*99*i, where 1 <= i <= m/8. The message text

shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

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

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

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

2.3.8.1 NDcPP22e:FCS_COP.1.1/KeyedHash

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3.3 of the ST states the TOE supports keyed-hash message authentication using HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 using SHA-1/256/384/512 with 160/256/384/512-bit keys to produce a 160/256/384/512 output MAC. The SHA-1/256 and 384/512 algorithms have block sizes of 512 and 1024-bits respectively.

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

The ST states “The HMAC configuration is automatic based on other configuration entries. No administrator action is needed.”

Component Testing Assurance Activities: For each of the supported parameter sets, the evaluator shall compose 15 sets of test data. Each set shall consist of a key and message data. The evaluator shall have the TSF generate

HMAC tags for these sets of test data. The resulting MAC tags shall be compared to the result of generating HMAC tags with the same key and message data using a known good implementation.

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

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

2.3.9.1 NDcPP22e:FCS_COP.1.1/SigGen

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3.3 in the ST states that the TOE supports the use of RSA with 2048, 3072 and 4096 bit key sizes, and ECDSA with a key size of 256 bits or greater for cryptographic signatures (specifically NIST curves P-256, P-384, or P-521). The four tables in section 6.3 identify the crypto modules that provide these cryptographic signature services.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC TLS Client and Server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The “Install the NGFW Engine in FIPS mode” section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the “Supporting documentation” section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

The “FIPS mode restrictions” section in the **Admin Guide** states that when FIPS mode is enabled, the following restrictions are enforced:

- The NGFW Engine local console, command line interface, and SSH access are not available
- The available cryptographic algorithms and configuration options in the SMC are restricted:
    - RSA key sizes of 2048 bits or greater are used for digital signature generation
    - ECDSA key sizes of 256 bits or greater are used for digital signature generation
    - SHA-1 cannot be used for digital signature generation

The “Install the Management Client” section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the “Supporting documentation” section.

In the “Configure settings for an evaluated configuration” section of the **Admin Guide**, the Audit Server Configuration entry in the table provides instructions for generating a certificate request and selecting the RSA or ECDSA key size. It further describes configuring the TLS profile and identifies the supported TLS ciphersuites for communication between the TOE and an external syslog server. When using an ECDHE cipher suite, P-521, P-384 and P-256 are automatically used in the TLS key establishment. Similarly, the SMC Web Access entry in the table provides instructions for creating an ECDSA certificate request, configuring the TLS cryptography suite set and identifies the supported TLS ciphersuites for HTTPS connections for SMC Web Access.

This section also provides a reference to the *Forcepoint Next Generation Firewall Product Guide* which provides detailed instructions for configuring and enabling TLS protection for communications between the TOE and an external syslog server and for HTTPS connections for SMC Web Access. This guide is accessed via the link found in the “Supporting documentation” section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates

to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

The “Setup VPN Profile Element” section of the **Admin Guide** includes instructions to allow the administrator to configure the specifics of IPsec (IKEv2/ESP) connections including:
In the IKE SA Tab
Cipher Algorithms
- **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
- **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.

Message Digest Algorithms
- **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Diffie-Hellman Groups
- **14 (2048 bits)** — Diffie-Hellman key exchange with a 2048-bit modulus.
- **15 (3072 bits)** — Diffie-Hellman key exchange with a 3072-bit modulus.
- **16 (4096 bits)** — Diffie-Hellman key exchange with a 4096-bit modulus.
- **17 (6144 bits)** — Diffie-Hellman key exchange with a 6144-bit modulus.
- **18 (8192 bits)** — Diffie-Hellman key exchange with a 8192-bit modulus.
- **19 (ECP 256 bits)** — Diffie-Hellman key exchange with 256-bit elliptic curve.
- **20 (ECP 384 bits)** — Diffie-Hellman key exchange with 384-bit elliptic curve.
- **21 (ECP 521 bits)** — Diffie-Hellman key exchange with 521-bit elliptic curve.

Authentication Method
- **RSA Signatures** — Requires that each Gateway has a valid certificate.
- **ECDSA Signatures** — Requires that each Gateway has a valid certificate.

And in the IPsec SA Tab
Cipher Algorithms
- **AES-128** — Advanced Encryption Standard CBC Mode algorithm with a 128-bit key size.
- **AES-256** — Advanced Encryption Standard CBC Mode algorithm with a 256-bit key size.
- **AES-GCM-128** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 128-bit key size. Recommended for high-speed networks.
- **AES-GCM-256** — Advanced Encryption Standard Galois/Counter Mode encryption algorithm with a 256-bit key size. Recommended for high-speed networks.

Message Digest Algorithms
- **SHA-2** — The SHA-2 hash function. Define the minimum length according to the security requirements for the VPN.

Component Testing Assurance Activities: ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

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

ECDSA FIPS 186-4 Signature Verification Test

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

RSA Signature Algorithm Tests

Signature Generation Test

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

The evaluator shall verify the correctness of the TOE's signature using a trusted reference implementation of the signature verification algorithm and the associated public keys to verify the signatures.

Signature Verification Test

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

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

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

2.3.10 HTTPS Protocol (NDcPP22e:FCS_HTTPS_EXT.1)

2.3.10.1 NDcPP22e:FCS_HTTPS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.10.2 NDcPP22e:FCS_HTTPS_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.10.3 NDcPP22e:FCS_HTTPS_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the ST states that the TOE provides an HTTPS/TLS interface for SMC Client GUI administration. The TOE implements the HTTPS protocol in accordance with RFC 2818.

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

Section “Configure settings for an evaluated configuration” in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the TLS ciphersuite set.

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

Tests are performed in conjunction with the TLS evaluation activities.

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

This test is satisfied by FTP_TRP.1-t1.

2.3.11 IPsec Protocol - per TD0633 (NDcPP22e:FCS_IPSEC_EXT.1)

2.3.11.1 NDcPP22e:FCS_IPSEC_EXT.1.1

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

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

Section 6.3 of the ST states that the TOE processes (in conjunction with its firewall rules) both incoming and outgoing packets according to its administrator configured Security Policy Descriptors (SPDs) in order to apply (or not apply) IPsec processing. The TOE provides both site-to-site as well as client IPsec encryption. The administrator can configure, for specific traffic, whether the TOE will discard (or drop), protect (encrypt with ESP), or bypass (forward without encryption) packets destined for specific peers, in compliance with RFC 4301. The administrator can configure the priority of the IPsec policies, and thus determine to order in which the TOE inspects (and applies) IPsec encryption policies (with the TOE applying the first matching policy) and the TOE recognizes packets from an established SA and automatically applies the encryption (or decryption) belonging to that SA.

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

Section “Setup VPN access rules” in the **Admin Guide** provides detailed instructions on how to create new rules to either Allow Discard or Refuse. To add rules select Configuration, then click NGFW, navigate to Policies then Firewall Policies. From firewall policies, one can select a policy to add or edit a rule into.

To set up a rule to encrypt traffic from the firewall policy create a rule and give it the Allow action, then open the action options. Set VPN action to Enforce VPN, then select a Policy-based VPN. Then save the policy and refresh the policies of all firewalls involved in the VPN to activate the new configuration. The process to set up a decrypt rule is the same as the encrypt rule, create a rule and give it the Allow action, then open the action options. Set VPN action to Enforce VPN, then select a Policy-based VPN.

To set up a rule to drop traffic select configuration then navigate to NGFW, Policies and then Firewall Policies. From there select the desired firewall policy. Add a rule and give it the Discard action.

To set a rule to allow traffic to flow through the TOE without being encrypted, add a rule and give it the "Allow option without any VPN Action" action.

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

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

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

The evaluator used to the TOE to configure a PROTECT/Encrypt, a DISCARD/Drop, and a BYPASS/Plaintext rule and then performed positive tests (where the traffic matched the rule) and negative tests (where the traffic did not match the rule) and observed that the TOE processed the traffic appropriately.

2.3.11.2 NDcPP22e:FCS_IPSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

See Test Case FCS_IPSEC_EXT.1.1 above.

2.3.11.3 NDcPP22e:FCS_IPSEC_EXT.1.3

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

Section 6.3 of the ST states that the TOE's IPsec implementation supports X.509 certificates (both RSA and ECDSA) for IKE authentication. For IKEv2 only tunnel mode is supported.

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

Section "VPN Configuration" in the **Admin Guide** provides the necessary steps to be completed to set up a VPN connection, including setting up IPsec VPN and a remote access VPN.

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

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

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

Test 1: The evaluator configured a test server to require only tunnel mode. The evaluator then attempted to

connect the IPsec VPN between the test server and the TOE expecting the connection to be successful only if the configured mode is supported by the TOE.

Test 2: NA - transport mode is not claimed.

2.3.11.4 NDcPP22e:FCS_IPSEC_EXT.1.4

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

Section 6.3 of the ST states that the TOE supports tunnel mode alone, supports AES-CBC (128/192/256-bit keys) for IKEv2, supports AES-CBC and GCM (128/192/256-bit keys) for ESP, supports all HMAC with SHA-1 (truncated to 96-bits as per RFC 2404) and SHA2 (256, 384, 512, not truncated) hashes for SA integrity, supports only IKEv2, supports administrator configuration of the maximum lifetime for IKEv2 and Child/ESP SAs, and also supports a maximum number of bytes for Child/ESP SAs.

Section 6.3 of the ST also includes tables describing the CAVP algorithms for the NGFW Engine, which includes algorithms for the HMAC and SHA hashing algorithms used in IPsec.

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

Section “VPN Profile Properties dialog box” in the **Admin Guide** explains how to configure the various setting of the TOE’s VPN profile including IPsec SA Cipher Algorithms, Message Digest Algorithms, Diffie-Hellman groups and Authentication method.

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

The evaluator attempted to establish an IPsec connection between a test server and the TOE using each of the supported encryption algorithms. The evaluator confirmed that each claimed encryption algorithm was successful.

2.3.11.5 NDcPP22e:FCS_IPSEC_EXT.1.5

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

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

Section 6.3 of the ST states that the TOE’s IPsec implementation supports IKEv2 in tunnel mode.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure it instructs the administrator how to configure the TOE to use IKEv1 and/or IKEv2 (as selected), and how to configure the TOE to perform NAT traversal for the following test (if selected).

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

Section "VPN Profile Properties dialog box" in the **Admin Guide** shows the IKE SA selections where IKEv2 is selected by default. The TOE does not support IKEv1. The "Define endpoints for External VPN Gateways" section explains how to configure the TOE to perform NAT traversal.

Testing Assurance Activities: Tests are performed in conjunction with the other IPsec evaluation activities.

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

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

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

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

2.3.11.6 NDcPP22e:FCS_IPSEC_EXT.1.6

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

Section 6.3 of the ST indicates that the encrypted payload for IKEv2 supports AES-CBC (128/192/256-bit keys). This is consistent with the FCS_IPSEC_EXT.1.6 requirement.

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

Section "VPN Profile Properties dialog box" in the **Admin Guide** explains how to configure the various setting of the TOE's VPN profile including IKE SA Cipher Algorithms and Message Digest Algorithms.

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

Test 1: The evaluator attempted to establish an IPsec connection between the TOE and a test server with each payload encryption algorithm supported by the TOE. The evaluator confirmed that all supported algorithms were successful.

2.3.11.7 NDcPP22e:FCS_IPSEC_EXT.1.7

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

Section 6.3 of the ST states that the TOE supports tunnel mode alone, supports AES-CBC (128/256-bit keys) for IKEv2, supports AES-CBC and GCM (128/256-bit keys) for ESP, supports all HMAC with SHA-1 (truncated to 96-bits as per RFC 2404) and SHA2 (256, 384, 512, not truncated) hashes for SA integrity, supports only IKEv2, supports administrator configuration of the maximum lifetime for IKEv2 and Child/ESP SAs, and also supports a maximum number of bytes for Child/ESP SAs.

This is consistent with the FCS_IPSEC_EXT.1.7 requirement.

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

Section “VPN Profile Properties dialog box” in the **Admin Guide** explains how to configure the various setting of the TOE’s VPN profile including a section labeled SA lifetime in minutes. The default IKE SA lifetime is 1440 minutes or 24 hours.

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

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

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

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

Test 1: NA - number of bytes is not claimed.

Test 2: The evaluator configured the TOE to have a 24 hour IKE and 8 hour ESP limits and the test server was configured to have 25 hour IKE and 9 hour ESP limits. The evaluator then connected the IPsec VPN between the test server and the TOE and waited for over 24 hours before terminating the test. The evaluator confirmed that an ESP rekey occurred every 8 hours and an IKE rekey occurred 24 hours after the initial connection was established.

2.3.11.8 NDcPP22e:FCS_IPSEC_EXT.1.8

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

Section 6.3 of the ST states that the TOE supports tunnel mode alone, supports AES-CBC (128/256-bit keys) for IKEv2, supports AES-CBC and GCM (128/256-bit keys) for ESP, supports all HMAC with SHA-1 (truncated to 96-bits as per RFC 2404) and SHA2 (256, 384, 512, not truncated) hashes for SA integrity, supports only IKEv2, supports administrator configuration of the maximum lifetime for IKEv2 and Child/ESP SAs, and also supports a maximum number of bytes for Child/ESP SAs.

This is consistent with the FCS_IPSEC_EXT.1.8 requirement.

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

Section "VPN Profile Properties dialog box" in the **Admin Guide** explains how to configure the various setting of the TOE's VPN profile including a section labeled IPsec Tunnel Lifetime. The default is 480 minutes with no limit on the amount of transferred data. The lowest limit that can be configured for the amount of data is 40000 KB.

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

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

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

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

Test 1: The evaluator configured the TOE to use the TOE's minimum data limit of 40,000 KB and then connected a test server a transferred data in excess of the configured amount. The evaluator confirmed than the TOE rekeyed the ESP.

Test 2: The evaluator configured the TOE to have a 24 hour IKE and 8 hour ESP limits and the test server was configured to have 25 hour IKE and 9 hour ESP limits. The evaluator then connected the IPsec VPN between the

test server and the TOE and waited for over 24 hours before terminating the test. The evaluator confirmed that an ESP rekey occurred every 8 hours and an IKE rekey occurred 24 hours after the initial connection was established.

2.3.11.9 NDcPP22e:FCS_IPSEC_EXT.1.9

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

Section 6.3 of the ST states that the TOE supports DH groups 14 through 21 and, using its DRBG, generates a secret key ("x") that has a length of twice the security strength of the negotiated DH group (thus because the TOE supports DH groups 14-21, the secret key length ranges between 224 bits and 256 bits). Additionally, the TOE generates nonces during IKE key exchange with a minimum size of either 128-bits or half the size of the negotiated PRF hash (whichever is less).

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.11.10 NDcPP22e:FCS_IPSEC_EXT.1.10

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

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

Section 6.3 of the ST states that the TOE supports DH groups 14 through 21 and, using its DRBG, generates a secret key ("x") that has a length of twice the security strength of the negotiated DH group (thus because the TOE supports DH groups 14-21, the secret key length ranges between 224 bits and 256 bits). Additionally, the TOE generates nonces during IKE key exchange with a minimum size of either 128-bits or half the size of the negotiated PRF hash (whichever is less).

Guidance Assurance Activities: None Defined

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

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

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

Test 1: NA, the first selection is not chosen for this SFR.

Test 2: See the TSS assurance activities above.

2.3.11.11 NDcPP22e:FCS_IPSEC_EXT.1.11

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

Section 6.3 in the ST indicates that the IKEv2 protocols supported by the TOE implement the following DH groups: 14 (2048-bit MODP), 15 (3072-bit MODP), 16 (4096-bit MODP), 17 (6144-bit MODP), 18 (8192-bit MODP), 19 (256-bit Random ECP), 20 (384-bit Random ECP) and 21 (521-bit Random ECP). The administrator can specify the DH Group in the IKEv2 policy. As a responder the TOE selects the first configured DH group proposed by the peer.

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

Section “VPN Profile Properties dialog box” in the **Admin Guide** explains how to configure the various setting of the TOE’s VPN profile including IPsec SA Cipher Algorithms, Message Digest Algorithms, Diffie-Hellman groups and Authentication method. The Diffie-Hellman groups include groups 14 through 21.

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

The evaluator attempted to establish an IPsec connection between a test server and the TOE using each of the supported DH groups. The evaluator confirmed that each claimed DH group was successful.

2.3.11.12 NDcPP22e:FCS_IPSEC_EXT.1.12

TSS Assurance Activities: The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges. The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to

ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

Section 6.3 of the ST states that the TOE prevents an administrator from misconfiguring the IKEv2 and Child/ESP SA cipher key lengths and ensures that the Child ESP SA's AES key length exceeds or equals that of the IKEv2 SA.

Guidance Assurance Activities: None Defined

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

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

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

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

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

Test 1: The evaluator attempted to establish an IPsec connection between a test server and the TOE using each of the supported hash functions. The evaluator confirmed that each claimed hash function was successful.

Test 2: The evaluator configured the TOE to accept both 128 and 256 bit key sizes for IKE and ESP and then initiated a connection from a test peer proposing a 128 bit key size for the IKE SA and a 256 bit key size for the ESP SA. The TOE refused to accept the configuration.

Test 3: The evaluator attempted to establish an IPsec connection between a test server and the TOE using unsupported algorithms and hash functions. In each case, the connection failed.

Test 4: The evaluator attempted to establish an IPsec connection between a test server and the TOE using an encryption algorithm not identified in FCS_IPSEC_EXT.1.4. The connection failed.

2.3.11.13 NDcPP22e:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1(2)/SigGen Cryptographic Operations (for cryptographic signature).

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

Section 6.3 of the ST states that the TOE's IPsec implementation supports X.509 certificates (both RSA and ECDSA) for IKE authentication.

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

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

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

Section "VPN Certificates" of the **Admin Guide** provides a high-level explanation of how to configure VPN certificates. Section "Create a certificate request for a VPN Gateway element" explains how to request an RSA or ECDSA certificate. The "VPN Profile Properties dialog box" section demonstrates where to select the appropriate certificate type..

The TOE does not support pre-shared keys.

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

The testing of IPsec authentication using Public Key was demonstrated throughout FCS_IPSEC_EXT.1 (including, for example, FCS_IPSEC_EXT.1.4), FIA_X509_EXT.1, and FIA_X509_EXT.2.

2.3.11.14 NDcPP22e:FCS_IPSEC_EXT.1.14

TSS Assurance Activities: The evaluator shall ensure that the TSS describes how the TOE compares the peer's presented identifier to the reference identifier. This description shall include which field(s) of the certificate are used as the presented identifier (DN, Common Name, or SAN). If the TOE simultaneously supports the same identifier type in the CN and SAN, the TSS shall describe how the TOE prioritizes the comparisons (e.g. the result of comparison if CN matches but SAN does not). If the location (e.g. CN or SAN) of non-DN identifier types must

explicitly be configured as part of the reference identifier, the TSS shall state this. If the ST author assigned an additional identifier type, the TSS description shall also include a description of that type and the method by which that type is compared to the peer's presented certificate, including what field(s) are compared and which fields take precedence in the comparison.

Section 6.3 in the ST states the TOE supports both RSA and ECDSA certificates for IKE peer authentication and can use SAN fields (IP, FQDN, or user FQDN) in addition to a peer certificate's Distinguished Name as a reference identifier.

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

The "Define endpoints for External VPN Gateways" section in the **Admin Guide** states the accepted identifiers including

- Distinguished Name to use Distinguished Name (DN)
- IP Address to use SAN: IP address
- DNS Name to use SAN: Fully Qualified Domain Name (FQDN)
- Email to use SAN: user FQDN

Testing Assurance Activities: In the context of the tests below, a valid certificate is a certificate that passes FIA_X509_EXT.1 validation checks but does not necessarily contain an authorized subject.

The evaluator shall perform the following tests:

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

Test 2: (conditional) For each SAN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes SAN checking over CN (through explicit specification of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the CN so it contains an incorrect identifier formatted to be the same type (e.g. the

reference identifier on the TOE is DNS-ID; identify certificate has an identifier in SAN with correct DNS-ID, CN with incorrect DNS-ID (and not a different type of identifier)) and verify that IKE authentication succeeds.

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

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

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

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

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

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

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

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

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

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

Test 1: Not applicable: the ST selected no CN/identifier type combinations.

Test 2:

(Part 1) The evaluator configured strongswan on a test peer to use an authentication certificate with the correct SAN: IP address, DNS address (FQDN), and user FQDN. The evaluator then attempted to connect the IPsec VPN between the test peer and observed the TOE accept each connection.

(Part 2), the test does not apply as the TOE does not support the Common Name (CN) for reference identifier,

only Subject Alternative Name (SAN).

Test 3: Not applicable: the ST selected no CN/identifier type combinations.

Test 4:

(Part 1) For this test, the evaluator alternately configured the TOE to look for each of the supported SAN reference identifiers. The evaluator then configured the strongswan VPN peer to use a certificate that would present an incorrect SAN reference identfier and a correct CN reference identifier. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE expecting the connection to be rejected.

(Part 2) – the test does not apply as the TOE does not support the Common Name (CN) for reference identifier, only Subject Alternative Name (SAN).

Test 5: The evaluator configured a test peer to send a certificate with a valid DN. The evaluator then initiated a connection between the TOE and the test peer and confirmed that the connection was successful.

Test 6a: The evaluator configured a test server to first send a certificate with an authorized DN, and then a nearly identical certificate but with a DN containing a duplicate CN. In each case, the evaluator attempted to establish an IPsec connection and confirmed that the TOE rejected the certificate with the duplicate CN.

Test 6b: The evaluator configured a test server to first sent a certificate with an authorized DN, and then a nearly identical certificate in which the evaluator appended '\0' to a non-CN field of an otherwise authorized DN. In each case, the evaluator attempted to establish an IPsec connection and confirmed that the TOE rejected the certificate with the DN containing a null character.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.12 IPsec Protocol - per TD0657 (VPNGW12:FCS_IPSEC_EXT.1)

2.3.12.1 VPNGW12:FCS_IPSEC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.2 VPNGW12:FCS_IPSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.3 VPNGW12:FCS_IPSEC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.4 VPNGW12:FCS_IPSEC_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.5 VPNGW12:FCS_IPSEC_EXT.1.5

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.6 VPNGW12:FCS_IPSEC_EXT.1.6

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.7 VPNGW12:FCS_IPSEC_EXT.1.7

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.8 VPNGW12:FCS_IPSEC_EXT.1.8

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.9 VPNGW12:FCS_IPSEC_EXT.1.9

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.10 VPNGW12:FCS_IPSEC_EXT.1.10

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.11 VPNGW12:FCS_IPSEC_EXT.1.11

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.12 VPNGW12:FCS_IPSEC_EXT.1.12

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.13 VPNGW12:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.14 VPNGW12:FCS_IPSEC_EXT.1.14

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

See NDcPP22e:FCS_IPSEC_EXT.1.14.

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

Pre-shared Keys was not selected.

Component Testing Assurance Activities: None Defined

2.3.13 NTP Protocol (NDcPP22e:FCS_NTP_EXT.1)

2.3.13.1 NDcPP22e:FCS_NTP_EXT.1.1

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

Section 6.3.3 of the ST states that the TOE provides the ability to synchronize its time with a NTP server using NTPv4. The time data is protected by a SHA1 message digest.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Time Management table entry provides instructions for configuring NTP time synchronization on both the SMC appliance and the NGFW Engines. By default, the NGFW Engine receives the time from the SMC, but can be optionally configured to synchronize directly with an NTP server.

The NTP implementation for both the SMC and the NGFW Engine is an NTP client only. NTP clients support communication with NTPv4 servers without any further configuration. For redundancy, it is possible to configure multiple NTP servers by creating one NTP server element for each NTP server. When enabling NTP time synchronization on the SMC appliance, all NTP server elements should be selected. For specific instructions, the reader is referred to the "Configuring system communications" chapter in the *Forcepoint Next Generation Firewall Product Guide* document which is accessed via the link found in the "Supporting documentation" section.

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

This test was performed as part of FCS_NTP_EXT.1.2, where the evaluator successfully observed that the TOE establishes a connection to the external NTP server using NTP version 4.

2.3.13.2 NDcPP22e:FCS_NTP_EXT.1.2

TSS Assurance Activities: None Defined

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

Assurance Activity Note:

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Time Management table entry provides instructions for configuring the NTP server to use a SHA-1 authentication key. This is performed during the configuration of the NTP server element by selecting SHA1 from the Key Type drop-down list. For specific instructions, the reader is referred to the "Configuring system communications" chapter in the *Forcepoint Next Generation Firewall Product Guide* document which is accessed via the link found in the "Supporting documentation" section.

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

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

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

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

This test was performed with both the SMC and Engine components of the TOE. On the SMC, time can be set manually by an administrator or by enabling NTP time synchronization. The Engine can either synchronize its time with the SMC or directly with an NTP server.

The evaluator configured a SHA1 key on the TOE and configured the NTP server with a different key and verified that the authentication failed and the TOE did not synchronize with the NTP server time source and the time did not change.

The evaluator then configured the NTP server to use the valid SHA1 key and verified that the authentication was successful and the TOE correctly synchronized its time with the NTP server. The evaluator also confirmed from the packet capture that the SHA1 message digest algorithm and NTP version 4 were used.

2.3.13.3 NDcPP22e:FCS_NTP_EXT.1.3

TSS Assurance Activities: None Defined

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Time Management table entry states that by default, the SMC appliance and NGFW Engine NTP clients do not update their times based on multicast or broadcast NTP packets. There is no configuration needed.

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

The evaluator configured the NTP server to support periodic time updates to broadcast and multicast addresses and then changed the time on the NTP server and viewed that the SMC/Engines did not sync after receiving the broadcast/multicast packets.

2.3.13.4 NDcPP22e:FCS_NTP_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

(TD0528 applied)

These tests were performed with both the SMC and Engine components of the TOE. On the SMC, time can be set manually by an administrator or by enabling NTP time synchronization. The Engine can either synchronize its time with the SMC or directly with an NTP server.

Test 1: The evaluator configured the TOE with 3 valid NTP connections. The evaluator changed the time on the NTP server and observed that the TOE updated its time and synched with one of the three valid NTP servers.

Test 2: The evaluator kept the TOE configured with the same 3 NTP servers as in test 1. The evaluator collected network traffic while monitoring the time on the TOE while an untrusted NTP server was configured to broadcast to the TOE. The evaluator confirmed via packet capture that the TOE ignored the NTP packets and could not update time using traditional authenticated updates with the invalid NTP server.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.14 Random Bit Generation (NDcPP22e:FCS_RBG_EXT.1)

2.3.14.1 NDcPP22e:FCS_RBG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.14.2 NDcPP22e:FCS_RBG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

Section 6.3 in the ST identifies the DRBG type for the SMC and the NGFW Engine. The SMC uses both an AES-256 CTR_DRBG and a SHA-512 Hash_DRBG while the Engines use an AES-256 CTR_DRBG.

Section 6.3.3 (FCS_RBG_EXT.1) of the ST states that the TOE components perform random bit generation in support of the cryptographic functions. The SMC uses both an AES-256 CTR_DRBG and a SHA-512 Hash_DRBG while the Engines use an AES-256 CTR_DRBG. The TOE components use a software-based noise source and draw from /dev/random to instantiate both the CTR_DRBGs and Hash_DRBG.

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

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

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

The “Evaluated Products” section of the **Admin Guide** describes the TOE including the cryptographic modules that provide all TOE cryptographic security functions. This section states that no other cryptographic modules have been evaluated or tested during the CC evaluation.

The “Establishing a security configuration” section of the **Admin Guide** provides the specific configuration steps for configuring the TOE into its evaluated configuration. This includes configuring the TOE components into FIPS mode and ensuring that 256-bit encryption as the security strength is enabled. Enabling FIPS mode configures the TOE to use the proper DRBG methods.

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

If the RNG has prediction resistance enabled, each trial consists of (1) instantiate DRBG, (2) generate the first block of random bits (3) generate a second block of random bits (4) uninstantiate. The evaluator verifies that the second block of random bits is the expected value. The evaluator shall generate eight input values for each trial. The first is a count (0 - 14). The next three are entropy input, nonce, and personalization string for the instantiate operation. The next two are additional input and entropy input for the first call to generate. The final two are additional input and entropy input for the second call to generate. These values are randomly generated. 'generate one block of random bits' means to generate random bits with number of returned bits equal to the Output Block Length (as defined in NIST SP800-90A).

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

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

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

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

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

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.

2.3.15 TLS Client Protocol Without Mutual Authentication - per TD0634 & TD0670 (NDcPP22e:FCS_TLSC_EXT.1)

2.3.15.1 NDcPP22e:FCS_TLSC_EXT.1.1

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

Section 6.3.3 (FCS_TLSC_EXT.1) of the ST states that the TOE communicates with both remote audit servers and other distributed TOE components using the TLS protocol. Section 6.3.2.1 and 6.3.2.2 of the ST indicate that the Management Server and the Log Server communicate with the external syslog server using the TLSv1.2 protocol.

The Management Server and Log Server use the following cipher suites to communicate with an external syslog server:

TLS_RSA_WITH_AES_128_CBC_SHA
TLS_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
TLS_RSA_WITH_AES_128_CBC_SHA256
TLS_RSA_WITH_AES_256_CBC_ SHA256
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_128_GCM_SHA256
TLS_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384

The TOE uses the following ciphersuite for all inter-TOE communication among its distributed components:

TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,

This is consistent with the FCS_TLSC_EXT.1 requirement. While FCS_TLSC_EXT.1 identifies two other ciphersuites supported by the SMC as Client (ITT), section 6.3.2.2 of the ST explains that there is only the one ciphersuite supported by both the SMC and the Engine. The two additional SMC ciphersuites were included and tested for completeness.

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

The “Enable FIPS mode on the SMC Appliance” section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS Mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

The "FIPS mode restrictions" section in the **Admin Guide** states that when FIPS mode is enabled, the following restrictions are enforced:

- The NGFW Engine local console, command line interface, and SSH access are not available
- The available cryptographic algorithms and configuration options in the SMC are restricted:
    - RSA key sizes of 2048 bits or greater are used for digital signature generation
    - ECDSA key sizes of 256 bits or greater are used for digital signature generation
    - SHA-1 cannot be used for digital signature generation

The "Configure settings for an evaluated configuration" section of the **Admin Guide** provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server

Configuration entry in the table also provides further guidance and specific options that should be selected and defined when following the instructions in the online guide to enable TLS v1.2 protection. It outlines the steps for configuring a trusted Root CA and generating a client certificate request which includes selecting an RSA with key size 2048 bits or greater, or ECDSA with 521 bits for P-521, 384 bits for P-384 or 256 bits for P-256. This section also provides the list of approved TLS cipher suites and instructs the user to ensure that the TLS cipher suites selected match the RSA and ECDSA parameters that are configured. When using an ECDHE cipher suite, P-521, P-384 and P-256 are automatically used in the TLS key establishment.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide,** the set of ciphersuites listed is consistent with those listed in the FCS_TLSC_EXT.1 requirement in the ST for communication between the TOE and an external syslog server and communication between the distributed TOE components.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

The registration of the NGFW Engine to the SMC to establish distributed TOE communication over TLS was demonstrated in FCO_CPC_EXT.1-t1.

The tests below were iterated for 3 variations as follows: Engine ITT Communication, SMC ITT Communication and SMC Syslog.

Test 1: The evaluator established a TLS connection between the TOE and a test server with each of the claimed cipher suites in turn and confirmed that successful connections were negotiated and that the packet capture identified the expected TLS cipher.

Test 2: The evaluator attempted to establish a TLS connection between the TOE and a test server using a valid certificate with the Server Authentication purpose in the extendedKeyUsage field. The connection was successful. Next the evaluator attempted to establish a TLS connection between the TOE and the external server using a certificate that did not include the Server Authentication purpose in the extendedKeyUsage field. A packet capture identified that the connection was not successful.

Test 3: The evaluator attempted to establish a TLS connection between the TOE and a test server with a certificate type that did not match the server-selected ciphersuite (e.g. an RSA certificate with an TLS_ECDHE_ECDSA_* ciphersuite). The connection attempt was rejected.

Test 4a: The evaluator attempted to establish a TLS connection between the TOE and a test server which sent only a TLS_NULL_WITH_NULL_NULL ciphersuite in the server hello. The connection was rejected.

Test 4b: The evaluator configured the TOE to connect to a test server. During the connection the evaluator caused the server to choose a ciphersuite that the TOE did not offer in its Client Hello handshake message. The connection was rejected.

Test 4c: The evaluator configured the TOE to connect to a test server using TLS with a TOE supported ECDHE key exchange method. The evaluator also configured the test server to accept that same ECHDE key exchange method, but to require a curve that was not supported by the TOE (i.e., P-192). The evaluator then caused the TOE to attempt the connection. The connection was rejected.

Test 5a: The evaluator attempted to establish a TLS connection between the TOE and a test server using TLS. During the connection, the evaluator caused the server to use a TLS version in the server hello that is a non-supported version. The connection was rejected.

Test 5b: The evaluator attempted to establish a TLS connection between the TOE and a test server. During the connection the evaluator caused the server to modify the signature block in the Server's Key Exchange handshake message. The connection attempt was rejected.

Test 6a: The evaluator attempted to establish a TLS connection between the TOE and a test server. During the connection the evaluator modified a byte in the Finished handshake message and verified that the TOE rejected the connection attempt after receiving the modified Finished message and no application data was exchanged.

Test 6b: The evaluator attempted to establish a TLS connection between the TOE and a test server. A garbled message was sent from the server after the server sent the ChangeCipherSpec message. The connection attempt was rejected.

Test 6c: The evaluator attempted to establish a TLS connection between the TOE and a test server. During the connection the evaluator caused the server to modify one byte in the server's nonce in the Server hello handshake. The connection was rejected.

2.3.15.2 NDcPP22e:FCS_TLSC_EXT.1.2

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

Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a 'Gatekeeper' discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the 'joining' component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the

client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

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

Section 6.3.3 of the ST states that the TOE communicates with both remote audit servers and other distributed TOE components using the TLS protocol. For TLS communication with remote audit servers, the administrator can configure a reference identifier of DNS name or IPv4 address. When configured with a DNS Name, the TOE will check the administrator configured value against the certificate's CN and SAN:DNS identifiers fields by first comparing the expected value against each SAN:DNS extension present in the certificate (if present), and if the TOE finds no SAN:DNS extensions, it will then compare the expected value against the certificate's CN. When configured with an IPv4 address, the TOE will check the IPv4 address against the certificate's SAN identifier field. For communication with distributed TOE components, the TOE mandates a numerical identifier to be found in the SAN:DNS extension. This expected numerical identifier is negotiated at the time of registration of the NGFW Engine to the SMC. The TOE does not support certificate pinning. The administrator need not (and cannot) explicitly configure anything regarding the ECDHE curves, the TOE always presents P-256, P-384, and P-521 curves in its client hello. The TOE supports wildcards for TLS communication with a remote audit server. The TOE does not support wildcards and will always reject them for Distributed TOE communication.

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

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

The "Configure settings for an evaluated configuration" section of the **Admin Guide** provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table also provides further guidance and specific options that should be selected and defined when following the instructions in the online guide to enable TLS v1.2 protection. Step 5 includes setting the reference identifier as the 'DNS Name' or 'IP Address' when configuring the TLS Server Identity.

Section "Enabling communication between the SMC and NGFW Engine" describes how after initial contact is made between the NGFW Engine and the SMC, subsequent TLS connections between the components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

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

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

or

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

or

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

Note that for some tests additional conditions apply.

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

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

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

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

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

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

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

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

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

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

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

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

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

Test 6 [conditional]: If IP address identifiers 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 192.168.0.1.20, CN=2001:0DB8:0000:0000:0008:0800:200C:*when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6). (TD0634 applied)

This negative test corresponds to the following section of the Application Note 64: '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.

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

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

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

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

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

The tests below were iterated for 3 variations as follows: Engine ITT Communication, SMC ITT Communication and SMC Syslog. The identifier type, DNS names, was tested in all 3 variations. In the SMC Syslog variation, IP address in the SAN was also tested.

Test 1: The evaluator attempted to establish a TLS connection between the TOE and a test server. The evaluator configured the server to provide a server certificate that contains a CN that did not match the reference identifier

and does not contain the SAN extension. The connection was rejected.

Test 2: The evaluator attempted to establish a TLS connection between the TOE and a test server. The evaluator configured the server to provide a server certificate that contains a CN that does match the reference identifier, contains the SAN extension, but does not contain an identifier in the SAN That matches the reference identifier. The connection was rejected.

Test 3: For syslog communication, the evaluator attempted to establish a TLS connection between the TOE and a test server. The evaluator configured the server to send a server certificate that contains a CN that matches the reference identifier and does not contain the SAN extension. The connection attempt was successful. For Distributed TOE communication, the SAN is mandated, so this test was not applicable.

Test 4: The evaluator attempted to establish a TLS connection between the TOE and a test server. The evaluator configured the server to 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 connection attempt was successful. This test was repeated between the distributed TOE components.

Test 5, part 1 & part 2: Wildcard is not supported for Distributed TOE communication, so all wildcard tests demonstrate a rejected connection. For communication with an external syslog server, the evaluator presented a server certificate containing a wildcard that was not in the left-most label of the presented identifier and verified that the connection failed. Next the evaluator presented a server certificate containing a wildcard in the left-most label and verified that the connection succeeded. The evaluator configured the reference identifier without a left-most label as in the certificate and verified that the connection failed. Lastly, the evaluator configured the reference identifier with two left-most labels and verified that the connection failed.

Test 6 - IPv4 addresses in the SAN are only supported by SMC syslog communications. For communication with an external syslog server, the evaluator presented a server certificate that contains a CN that matches the reference identifier, except one of the groups is replaced with an asterisk (ie. Wildcard CN) and does not contain the SAN extension. The connection was rejected.

Test 7 - Not applicable, RFC 5280 is not claimed.

2.3.15.3 NDcPP22e:FCS_TLSC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

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

Test 1: Successful connections with a valid certificate chain were demonstrated in FCS_TLSC_EXT.1.1-t1.

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

- Match the reference identifier -- Corresponds to FCS_TLSC_EXT.1.2 Tests 1 through 7.
- Validate certificate path -- Corresponds to FIA_X509_EXT.1.1 Test 1 for both Rev and ITT
- Validate expiration date -- Corresponds to FIA_X509_EXT.1.1 Test 2 both Rev and ITT
- Determine the revocation status -- Corresponds to FIA_X509_EXT.2 Test 1 For syslog only

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

2.3.15.4 NDcPP22e:FCS_TLSC_EXT.1.4

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

Section 6.3.3 of the ST states that the TOE administrator need not (and cannot) explicitly configure anything regarding the ECDHE curves, the TOE always presents P-256, P-384, and P-521 curves in its client hello.

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

As noted in the TSS assurance activity above, section 6.3.3 of the ST indicates that in the evaluated configuration, the TOE administrator need not (and cannot) explicitly configure anything regarding the ECDHE curves, the TOE always presents P-256, P-384, and P-521 curves in its client hello.

The “Enable FIPS mode on the SMC Appliance” section of the **Admin Guide** states that in order to comply with

Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. Both 256-bit and 128-bit encryption can be used for audit export.

For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table provides instructions for generating a certificate request and selecting the RSA or ECDSA key size. It further describes configuring the TLS profile and identifies the supported TLS ciphersuites for communication between the TOE and an external syslog server. When using an ECDHE cipher suite, P-521, P-384 and P-256 are automatically used in the TLS key establishment.

This section also provides a reference to the *Forcepoint Next Generation Firewall Product Guide* which provides detailed instructions for configuring and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

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

The test below was iterated for 3 variations as follows: Engine ITT Communication, SMC ITT Communication and SMC Syslog.

Test 1: The evaluator attempted to establish a connection between the TOE and the test server with a TOE supported ECDHE key exchange method using P-256, P-384 and P-521 curves and verified successful connections. The evaluator configured the server to accept that same ECHDE key exchange method, but to require a curve that was not supported by the TOE (i.e., P-192). The connection was rejected.

Component TSS Assurance Activities: None Defined **Component Guidance Assurance Activities**: None Defined **Component Testing Assurance Activities**: None Defined

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

2.3.16.1 NDcPP22e:FCS_TLSC_EXT.2.1

TSS Assurance Activities: None Defined **Guidance Assurance Activities**: None Defined **Testing Assurance Activities**: None Defined
Component TSS Assurance Activities: The evaluator shall ensure that the TSS description required per FIA_X509_EXT.2.1 includes the use of client-side certificates for TLS mutual authentication.

Section 6.3.3 of the ST states that mutual authentication using client-side x.509v3 certificates is supported by the SMC TLS client for syslog over TLS and for the TLS communication between the distributed TOE components.

The TOE provides the administrator the ability to generate or import either an ECDSA [P-256, P-384, or P-521] or RSA [2048, 3072] key to use for TLS Client or mutual authentication during TLS syslog export.

The TOE also provides a mutually authenticated (using internal CA certificates) TLS server interface on the SMC and NGFW Engines to allow for secure, distributed TOE communications between those two components. When the components exchange certificates each side compares the received SAN:DNS to ensure that it matches the expected identifier of the component. The TOE's components require the presence of the SAN:DNS and do not rely upon CN.

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

In section "Configure settings for an evaluated configuration", the Audit Server Configuration table entry describes how to configure the client cert for mutual authentication for syslog by generating the client certificate request, exporting the certificate request and importing a signed certificate. A reference is provided to the *Forcepoint Next Generation Firewall Product Guide* which provides detailed instructions for configuring and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities including creating and exporting a certificate request and importing an externally signed certificate.

There is no configuration needed for the client side certificates used in mutual authentication between the TOE's distributed components as this is all performed automatically using the SMC's internal CA during the registration process. Section "Enabling communication between the SMC and NGFW Engine" in the **Admin Guide** indicates that after initial contact is made, subsequent TLS connections between the components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

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

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

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

All of the TLS client implementation support mutual authentication. All tests for FCS_TLSC_EXT.1 and FIA_X509_EXT.* were performed with mutual authentication enabled.

The test below was iterated for 3 variations as follows: Engine ITT Communication, SMC ITT Communication and SMC Syslog.

Test 1: The evaluator attempted to establish a connection between the TOE and the test server. During the first connection the server did not request a client certificate. During the second connection, the server requested the TOE/client provide a certificate chaining to a common RootCA. Both connections succeeded but the second

connection ensured that the TOE TLS client sent both client Certificate (type 11) and client Certificate Verify (type 15) messages during its negotiation of a TLS channel and that Application Data was sent.

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

2.3.17.1 NDcPP22e:FCS_TLSS_EXT.1.1

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

Section 6.3.2.1 of the ST indicates that the Management Server accepts incoming administrative sessions (SMC Client connections) that are protected by TLS and use the following ciphersuites:

TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,

TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

Section 6.3.2.2 of the ST indicates that for all inter-TOE communication, the following cipher is used by both the SMC and the NGFW Engine to protect the communication between the distributed TOE components:

TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The SMC supports the following additional ciphersuites which were tested for completeness:

SMC (as both Client and Server): TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

These ciphersuites are consistent with those listed for FCS_TLSS_EXT.1.

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

The “Enable FIPS mode on the SMC Appliance” section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server. For specific instructions, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength when installing the TOE.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the TLS ciphersuite set.

The "Install the Management Client" section of the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* guide which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode:

- The "Install the Management Client using a file" subsection describes how to access, download and install the management client on a PC in the operating environment and then ensure that it is operating in FIPS mode by selecting the 'Restricted Cryptographic Algorithms Compatible with FIPS 140-2' operating mode.

The "Install the Management Client" section of the **Admin Guide** also references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the "Supporting documentation" section.

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

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

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

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

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

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

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

The tests below were iterated for 3 variations as follows: Engine ITT Communication, SMC GUI and SMC ITT Communication.

Test 1: The evaluator attempted to establish a TLS connection on the TOE with each of the ciphersuites identified by the Protection Profile. A packet capture identified the expected TLS ciphers are negotiated.

Test 2: The evaluator attempted to establish a TLS connection with the TOE using a list of cipher suites not claimed in the ST and also attempted to establish a connection using TLS_NULL_WITH_NULL_NULL cipher suite. A packet capture identified that all of the connections were rejected.

Test 3a: The evaluator attempted to establish a TLS connection with the TOE where the evaluator modified a byte in the Finished handshake message sent to the TOE. The connection was rejected.

Test 3b: The evaluator made connection attempts from a client to the TOE. The evaluator established a successful connection with the TOE, captured the negotiation and observed that the TOE sent an encrypted Finished message

and that the first byte of the encrypted Finished message did not equal 0x14.

2.3.17.2 NDcPP22e:FCS_TLSS_EXT.1.2

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

Section 6.3.3 of the ST indicates that the TOE does not provide the ability for the administrator to specify the versions of TLS that the TOE's server will negotiate, the TOE only negotiates TLSv1.2 with clients.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed.

When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server.

For more detailed instructions and steps, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength on both the SMC and the NGFW Engine when installing the TOE.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections.

The instructions include configuring the TLS ciphersuite set for which a reference is provided to the *Forcepoint Next Generation Firewall Product Guide.*

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) - sub section "Create TLS Cryptography Suite Set elements" provides instructions for defining the TLS ciphersuites for TLS 1.2 only.

The "Install the Management Client" section of the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* and the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. These guides are both accessed via the link found in the "Supporting documentation" section.

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

The tests below were iterated for 3 variations as follows: Engine ITT Communication, SMC GUI and SMC ITT Communication.

Test 1: The evaluator alternately attempted to connect to the TOE using SSL2.0, SSL3.0, TLSv1.0, TLSv1.1 and TLSv1.2. The connection attempt using TLSv1.2 was successful. All other attempts failed.

2.3.17.3 NDcPP22e:FCS_TLSS_EXT.1.3

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

Section 6.3.3 of the ST states that the TOE's TLS server acts similarly to its TLS client in that the administrator need not (and cannot) explicitly configure anything regarding the ECDHE curves, the TOE will negotiate P-256, P-384, or P-521 curves based upon what the peer/client specifies it supports in its hello.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed.

When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The "Install the NGFW Engine in FIPS mode" section in the **Admin Guide** similarly states that in order to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 cipher suite is used for the connection between the NGFW Engine and the Management Server.

For more detailed instructions and steps, the reader is referred to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document provides the instructions for enabling FIPS mode and 256-bit encryption as the security strength on both the SMC and the NGFW Engine when installing the TOE.

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

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

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

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

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

The tests below were iterated for 3 variations as follows: Engine ITT Communication, SMC GUI and SMC ITT Communication.

Test 1: The evaluator attempted to establish a TLS session with the TOE when the evaluator's client specified only one key exchange method in the Client Hello. The evaluator observed that the P-256, P-384 and P-521 elliptic curves were all supported as claimed. The evaluator then attempted a connection using an unsupported elliptic curve (ie. P-192) and verified that the connection failed.

Test 2: Not applicable. The TOE does not support DHE ciphers.

Test 3: Not applicable. The TOE does not support RSA ciphers.

2.3.17.4 NDcPP22e:FCS_TLSS_EXT.1.4

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

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

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

Section 6.3.3 in the ST states that the TOE does not support session resumption or session tickets.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: Test Objective: To demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).

Test 1 [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps: Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

d) The client completes the TLS handshake and captures the SessionID from the ServerHello.

e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).

f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).

b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 3 [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with a ServerHello with an empty SessionTicket extension, NewSessionTicket, ChangeCipherSpec and Finished messages (as seen in figure 2 of RFC 5077).

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

Test 1 was iterated for 3 variations as follows: Engine ITT Communication, SMC GUI and SMC ITT Communication.

Test 1: The evaluator attempted to establish a TLS session with the TOE when the evaluator's client initiated a connection attempt in which the Client Hello has a zero-length Session ID and a zero-length Session Ticket extension. The TOE's Engine TLS server sent a zero-length session ID, indicating that the server does not support session resumption based on session IDs and did not send a session ticket indicating it does not support session resumption based on session tickets. The two SMC variations returned non-zero session IDs and no session ticket indicating they do not support session resumption based on session tickets. The evaluator then attempted to reuse the prior session ID resulting in the SMC servers providing a new sessionID (indicating that the SMC servers did not reuse the old session and instead required a new session).

Test 2 – Not applicable

Test 3 - Not applicable

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.18 TLS Server Support for Mutual Authentication (NDcPP22e:FCS_TLSS_EXT.2)

2.3.18.1 NDcPP22e:FCS_TLSS_EXT.2.1

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

The evaluator shall verify the TSS describes how the TSF uses certificates to authenticate the TLS client. The evaluator shall verify the TSS describes if the TSF supports any fallback authentication functions (e.g. username/password, challenge response) the TSF uses to authenticate TLS clients that do not present a certificate. If fallback authentication functions are supported, the evaluator shall verify the TSS describes whether the fallback authentication functions can be disabled.

Section 6.3.3 (FCS_TLSS_EXT.1/2) of the ST states that the TOE provides a mutually authenticated TLS server interface on the SMC and Engines which allows for secure, distributed TOE communications between these two components. Mutual authentication between the SMC and Engines is performed using internal CA certificates. When the components exchange certificates as part of distributed TOE TLS handshake authentication, each side compares the received SAN:DNS to ensure that it matches the expected identifier of the component (the TOE's components require the presence of the SAN:DNS and do not rely upon CN).

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

The evaluator shall verify the guidance describes how to configure the TLS client certificate authentication function. If the TSF supports fallback authentication functions, the evaluator shall verify the guidance provides instructions for configuring the fallback authentication functions. If fallback authentication functions can be disabled, the evaluator shall verify the guidance provides instructions for disabling the fallback authentication functions.

There is no configuration needed for the client side certificates used in mutual authentication between the TOE's distributed components as this is all performed automatically using the SMC's internal CA during the registration process. Section "Enabling communication between the SMC and NGFW Engine" in the **Admin Guide** indicates that after initial contact is made, subsequent TLS connections between the components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

Testing Assurance Activities: Test 1a [conditional]: If the TOE requires or can be configured to require a client certificate, the evaluator shall configure the TOE to require a client certificate and send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 1b [conditional]: If the TOE supports fallback authentication functions and these functions cannot be disabled. The evaluator shall configure the fallback authentication functions on the TOE and configure the TOE to send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify the TOE authenticates the connection using the fallback authentication functions as described in the TSS.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

Test 2 [conditional]: If TLS 1.2 is claimed for the TOE, the evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the connection is denied.

Test 3: The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.

Test 4: The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.

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

a) Configure the server to require mutual authentication and then connect to the server with a client configured to send a client certificate that is signed by a Certificate Authority trusted by the TOE. The evaluator shall verify that the server accepts the connection.

b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message (see RFC5246 Sec 7.4.8). The evaluator shall verify that the server rejects the connection.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

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

Test 6: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds, and a trusted channel can be established.

Test 7: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 8 [conditional]: The purpose of this test is to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

The tests below were iterated for 2 variations as follows: Engine ITT Communication and SMC ITT Communication. The TOE does not support fallback authentication so the connection should fail in each case.

Test 1a: The evaluator configured a test client to connect to the TOE using TLS without providing a certificate as requested by the TOE. The client was unable to connect when the server required a certificate and none was provided. The evaluator confirmed that there was no flow of application data.

Test 1b: Not applicable. The TOE does not support fallback authentication functions.

Test 2: The evaluator configured a test client to connect to the TOE using TLS with an unsupported signature algorithm (md5). The connection failed.

Test 3: The evaluator configured a test client to connect to the TOE using TLS with a certificate issued by an imposter root CA (i.e, one with an issuer CA distinguished name matching a trusted CA specified by the TOE in its Certificate Request message, but where the imposter CA's signing key differs). The evaluator observed the TOE reject the connection attempt.

Test 4: The evaluator configured a test client to connect to the TOE using TLS. The first connection attempt is made with a valid certificate including the client auth EKU. This connection is accepted. During the second connection attempt the testy client provides a certificate without the ClientAuthentication EKU. This connection attempt is rejected.

Test 5a: The successful connection with a client cert signed by the trusted root CA was demonstrated in FIA_X509_EXT.1.1-t1.

Test 5b: The evaluator used an openssl s_client to attempt to connect to the TOE, but a modified openssl library causes the client to modify the client Certificate Verify handshake message. The evaluator observed that the TOE rejected the connection.

Test 6: This test has been performed in FIA_X509_EXT.1.1-t1 where a successful connection with a valid cert chain is tested.

Test 7: This test has been performed in several other test activities. Specifically, this test repeats the assurance activities as described here.

- match the reference identifier -- Corresponds to FCS_TLSS_EXT.2.3-t1
- validate certificate path and validate expiration date -- Corresponds to FIA_X509_EXT.1.1/REV Test 2-5
- failed determination of revocation status -- Corresponds to FIA_X509_EXT.2-t1

Test 8: Not applicable. The TOE does not support administrator override.

2.3.18.2 NDcPP22e:FCS_TLSS_EXT.2.2

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

The evaluator shall verify the TSS describes how the TSF uses certificates to authenticate the TLS client. The evaluator shall verify the TSS describes if the TSF supports any fallback authentication functions (e.g. username/password, challenge response) the TSF uses to authenticate TLS clients that do not present a certificate. If fallback authentication functions are supported, the evaluator shall verify the TSS describes whether the fallback authentication functions can be disabled.

This activity has been performed in FCS_TLSS_EXT.2.1.

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

The evaluator shall verify the guidance describes how to configure the TLS client certificate authentication function. If the TSF supports fallback authentication functions, the evaluator shall verify the guidance provides instructions for configuring the fallback authentication functions. If fallback authentication functions can be disabled, the evaluator shall verify the guidance provides instructions for disabling the fallback authentication functions.

This activity has been performed in FCS_TLSS_EXT.2.1.

Testing Assurance Activities: Test 1a [conditional]: If the TOE requires or can be configured to require a client certificate, the evaluator shall configure the TOE to require a client certificate and send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify that the handshake is not finished successfully and no application data flows.

Test 1b [conditional]: If the TOE supports fallback authentication functions and these functions cannot be disabled. The evaluator shall configure the fallback authentication functions on the TOE and configure the TOE to send a Certificate Request to the client. The evaluator shall attempt a connection while sending a certificate_list structure with a length of zero in the Client Certificate message. The evaluator shall verify the TOE authenticates the connection using the fallback authentication functions as described in the TSS.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

Test 2 [conditional]: If TLS 1.2 is claimed for the TOE, the evaluator shall configure the server to send a certificate request to the client without the supported_signature_algorithm used by the client's certificate. The evaluator shall attempt a connection using the client certificate and verify that the connection is denied.

Test 3: The aim of this test is to check the response of the server when it receives a client identity certificate that is signed by an impostor CA (either Root CA or intermediate CA). To carry out this test the evaluator shall configure the client to send a client identity certificate with an issuer field that identifies a CA recognised by the TOE as a trusted CA, but where the key used for the signature on the client certificate does not correspond to the CA certificate trusted by the TOE (meaning that the client certificate is invalid because its certification path does not terminate in the claimed CA certificate). The evaluator shall verify that the attempted connection is denied.

Test 4: The evaluator shall configure the client to send a certificate with the Client Authentication purpose in the extendedKeyUsage field and verify that the server accepts the attempted connection. The evaluator shall repeat this test without the Client Authentication purpose and shall verify that the server denies the connection. Ideally, the two certificates should be identical except for the Client Authentication purpose.

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

a) Configure the server to require mutual authentication and then connect to the server with a client configured to send a client certificate that is signed by a Certificate Authority trusted by the TOE. The evaluator shall verify that the server accepts the connection.

b) Configure the server to require mutual authentication and then modify a byte in the signature block of the client's Certificate Verify handshake message (see RFC5246 Sec 7.4.8). The evaluator shall verify that the server rejects the connection.

Note: Testing the validity of the client certificate is performed as part of X.509 testing.

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

Test 6: Using the administrative guidance, the evaluator shall load a CA certificate or certificates needed to validate the presented certificate used to authenticate an external entity and demonstrate that the function succeeds, and a trusted channel can be established.

Test 7: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

Test 8 [conditional]: The purpose of this test is to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

This activity has been performed in FCS_TLSS_EXT.2.1.

2.3.18.3 NDcPP22e:FCS_TLSS_EXT.2.3

TSS Assurance Activities: The evaluator shall verify that the TSS describes which types of identifiers are supported during client authentication (e.g. Fully Qualified Domain Name (FQDN)). If FQDNs are supported, the evaluator shall verify that the TSS describes that corresponding identifiers are matched according to RFC6125. For all other types of identifiers, the evaluator shall verify that the TSS describes how these identifiers are parsed from the certificate, what the expected identifiers are and how the parsed identifiers from the certificate are matched against the expected identifiers.

Section 6.3.3 (FCS_TLSC_EXT.1/2) of the ST states that for communication with distributed TOE components, the TOE mandates a numerical identifier to be found in the SAN:DNS extension. This expected numerical identifier is negotiated at the time of registration of the NGFW Engine to the SMC. The TOE does not support certificate pinning. The TOE does not support wildcards and will always reject them for Distributed TOE communication.

Section 6.3.3 (FCS_TLSS_EXT.1/2) of the ST states that the TOE provides a mutually authenticated TLS server interface on the SMC and Engines which allows for secure, distributed TOE communications between these two components. Mutual authentication between the SMC and Engines is performed using internal CA certificates. When the components exchange certificates as part of distributed TOE TLS handshake authentication, each side compares the received SAN:DNS to ensure that it matches the expected identifier of the component (the TOE's components require the presence of the SAN:DNS and do not rely upon CN).

Guidance Assurance Activities: The evaluator shall ensure that the AGD guidance describes the configuration of expected identifier(s) for X.509 certificate-based authentication of TLS clients. The evaluator ensures this description includes all types of identifiers described in the TSS and, if claimed, configuration of the TOE to use a directory server.

Section "Enabling communication between the SMC and NGFW Engine" in the **Admin Guide** indicates that after initial contact is made, subsequent TLS connections between the components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

Testing Assurance Activities: The evaluator shall send a client certificate with an identifier that does not match an expected identifier and verify that the server denies the connection.

This test was iterated for 2 variations as follows: Engine ITT Communication and SMC ITT Communication.

The evaluator configured a test client to connect to the TOE using TLS. The evaluator alternately used a valid client certificate (control) and a certificate with an identifier not configured on the TOE but is otherwise valid (e.g., chains to the root configured on the TOE). The evaluator observed that the control connection succeeded, while the TOE rejected the second connection.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: For all tests in this chapter the TLS client used for testing of the TOE shall support mutual authentication.

All tests were performed with a TLS client that supports mutual authentication.

2.4 User data protection (FDP)

2.4.1 Full Residual Information Protection (STFFW14E:FDP_RIP.2)

2.4.1.1 STFFW14E:FDP_RIP.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: 'Resources' in the context of this requirement are network packets being sent through (as opposed to 'to', as is the case when a security administrator connects to the TOE) the TOE. The concern is that once a network packet is sent, the buffer or memory area used by the packet still contains data from that packet, and that if that buffer is re-used, those data might remain and make their way into a new packet. The evaluator shall check to ensure that the TSS describes packet processing to the extent that they can determine that no data will be reused when processing network packets. The evaluator shall ensure that this description at a minimum describes how the previous data are zeroized/overwritten, and at what point in the buffer processing this occurs.

Section 6.4 of the ST states that the TOE has been designed to ensure that no residual information exists in network packets. When the TOE allocates a new buffer for either an incoming or outgoing network packet, the new packet data will be used to overwrite any previous data in the buffer. If an allocated buffer exceeds the size of the packet, and additional space will be overwritten (padded) with zeros before the packet is forwarded (to the external network or delivered to the appropriate, internal application).

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5 Firewall (FFW)

2.5.1 Stateful Traffic Filtering (STFFW14E:FFW_RUL_EXT.1)

2.5.1.1 STFFW14E:FFW_RUL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.1.2 STFFW14E:FFW_RUL_EXT.1.2

TSS Assurance Activities: The evaluator shall verify that the TSS describes a stateful packet filtering policy and the following attributes are identified as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

The evaluator shall verify that each rule can identify the following actions: permit or drop with the option to log the operation. The evaluator shall verify that the TSS identifies all interface types subject to the stateful packet filtering policy and explains how rules are associated with distinct network interfaces.

Section 6.5 of the ST states that the TOE provides an information flow control mechanism using a rule base that comprises a set of security policy rules, i.e., the firewall security policy. The NGFW engine enforces the firewall security policy on all traffic that passes through the engine, via its internal or external network Ethernet interfaces. The traffic is TCP, UDP, ICMPv4, ICMPv6, connections over IPv4 and IPv6. The NGFW engine inspects and filters these protocols based upon their header fields as defined by their corresponding RFC. The table below identifies the protocols and fields filtered by the TOE.

Protocol	Related RFC	Fields Inspected
ICMPv4	RFC 792	Type, Code
ICMPv6	RFC 4443	Type, Code
IPv4	RFC 791	Source Address, Destination Address, Transport layer protocol
IPv6	RFC 2460	Source Address, Destination Address, Transport layer protocol
TCP	RFC 793	Source Port, Destination Port
UDP	RFC 768	Source Port, Destination Port

Each rule comprises matching criteria and target actions. If the matching criteria is verified, the NGFW engine applies the target actions. Possible target actions include: Allow, Discard and Refuse. Access rules with the logging option can create a log or alert entry each time they match. The logging option is in addition to the target action of a rule. An administrator can specify that a rule apply to a specific interface by specifying a zone, adding a given firewall interface to that zone, and then specifying that zone as either a source or destination address for the rule.

Guidance Assurance Activities: The evaluators shall verify that the guidance documentation identifies the following attributes as being configurable within stateful traffic filtering rules for the associated protocols:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

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

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

The “How firewalls process traffic” section of the **Admin Guide** indicates that the TOE supports the following protocols and their attributes:

RFC 792 (ICMPv4)
- Type
- Code

RFC 4443 (ICMPv6)
- Type
- Code

RFC 791 (IPv4)
- Source address
- Destination address
- Transport layer protocol

RFC 2460 (IPv6)

- Source address
- Destination address
- Transport layer protocol

RFC 793 (TCP)

- Source port
- Destination port

RFC 768 (UDP)

- Source port
- Destination port

The "Create a customized Firewall Policy Template" section of the **Admin Guide** indicates that the rules can be configured with the actions to Discard or Allow. It provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for creating a customized firewall policy template.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 48 (Creating and managing policy elements) describes how to add specific access rules to a firewall policy template and then use that template to create security policies.

The "Create a customized Firewall Policy Template" section of the **Admin Guide** further provides specific network elements and access rules that should be configured when following the instructions in the online guide to create a firewall policy template.

The "Create a Firewall Policy" section of the **Admin Guide** provides steps for creating a firewall policy which include adding access rules and configuring logging for a rule. This section states that access rules affect all network interfaces, unless the source interface is specified. A reference is provided to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide information on using Zone elements to specify the source interface.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 50 (Access rules) describes how to use Zone elements to match traffic based on which interfaces it is traveling through. Zone elements are interface references that can combine several network interfaces of an engine into one logical entity. Using Zones in the Source or Destination cells allows you to restrict traffic according to which interfaces the traffic is traveling through.

Testing Assurance Activities: Test 1: The evaluator shall use the instructions in the guidance documentation to test that stateful packet filter firewall rules can be created that permit, drop, and log packets for each of the following attributes:

- ICMPv4

o Type

o Code

- ICMPv6

o Type

o Code

- IPv4

o Source address

o Destination Address

o Transport Layer Protocol

- IPv6

o Source address

o Destination Address

o Transport Layer Protocol and where defined by the ST author, Extension Header Type, Extension Header Fields

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

Test 2: Repeat the test evaluation activity above to ensure that stateful traffic filtering rules can be defined for each distinct network interface type supported by the TOE.

Note that these test activities should be performed in conjunction with those of FFW_RUL_EXT.1.9 where the effectiveness of the rules is tested. The test activities for FFW_RUL_EXT.1.9 define the protocol/attribute combinations required to be tested. If those combinations are configured manually, that will fulfil the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

Test 1: The evaluator configured firewall rules for testing of the other STFFW14E:FFW_RUL_EXT.1 tests (including

STFFW14E:FFW_RUL_EXT.1.9) using instructions provided within the administrative guidance and found all necessary instructions were provided accurately.

Test 2: The evaluator configured firewall rules for testing of the other STFFW14E:FFW_RUL_EXT.1 tests (including STFFW14E:FFW_RUL_EXT.1.9) using instructions provided within the administrative guidance and found all necessary instructions were provided accurately. The evaluator found that these rules can be applied to all types of supported network interfaces.

2.5.1.3 STFFW14E:FFW_RUL_EXT.1.3

TSS Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Guidance Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Testing Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

2.5.1.4 STFFW14E:FFW_RUL_EXT.1.4

TSS Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Guidance Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

Testing Assurance Activities: See FFW_RUL_EXT.1.2

See FFW_RUL_EXT.1.2

2.5.1.5 STFFW14E:FFW_RUL_EXT.1.5

TSS Assurance Activities: The evaluator shall verify that the TSS identifies the protocols that support stateful session handling. The TSS shall identify TCP, UDP, and, if selected by the ST author, also ICMP.

The evaluator shall verify that the TSS describes how stateful sessions are established (including handshake processing) and maintained.

The evaluator shall verify that for TCP, the TSS identifies and describes the use of the following attributes in session determination: source and destination addresses, source and destination ports, sequence number, and individual flags.

The evaluator shall verify that for UDP, the TSS identifies and describes the following attributes in session determination: source and destination addresses, source and destination ports.

The evaluator shall verify that for ICMP (if selected), the TSS identifies and describes the following attributes in session determination: source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5.

The evaluator shall verify that the TSS describes how established stateful sessions are removed. The TSS shall describe how connections are removed for each protocol based on normal completion and/or timeout conditions. The TSS shall also indicate when session removal becomes effective (e.g., before the next packet that might match the session is processed).

Section 6.5 of the ST states that the NGFW engine implements connection tracking to manage the information flow control decisions for connections (i.e., stateful sessions) rather than packets, providing increased performance and support for firewall features that require packet information above the IP level (e.g., ICMP, TCP, UDP). The connection tracking mechanism stores the state information of each connection to allow packets belonging to an established connection to pass. Connection tracking follows the standard TCP handshaking process (SYN, responding SYN-ACK, followed by ACK) to denote establishment of a stateful session, and the TOE's connection tracking will eliminate existing connections immediately, upon completion of the flow (in the case of TCP and FTP) or upon an inactivity timeout for the session. The connection tracking uses the fields shown in the following table when determining whether a packet matches an allowed established session for the corresponding protocol.

Protocol	**Connection Tracking**
TCP	Source & Destination Address, Source & Destination Port, Sequence Number, Flags
UDP	Source & destination address, source & destination port
ICMP	Source and destination address, type, code
FTP	TCP data session attributes

The NGFW engine follows a specific orderly algorithm to traverse the rule base for matching and filtering the traffic between its internal and external networks. Any traffic that is not explicitly accepted by the security policy is rejected by the firewall. The structure of the rule base and the capabilities of its associated protocol agents enable the TSF to make the information flow control decisions.

Each rule comprises matching criteria and target actions. If the matching criteria is verified (i.e., a comparison matches) the NGFW engine applies the target actions. Possible target actions include Allow, Discard and Refuse. Access rules with the logging option, can create a log or alert entry each time they match. The logging option is in addition to the target action of a rule.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes stateful session behaviours. For example, a TOE might not log packets that are permitted as part of an existing session.

The “How firewalls process traffic” section of the **Admin Guide** states that the NGFW Engine permits or denies traffic according to firewall filtering rules that are contained in a firewall policy. Each policy is based on a template policy which contains necessary predefined rules and also enables automatic rules for the NGFW engine to communicate with the SMC. A firewall only passes the traffic that is explicitly allowed in the firewall policy.

Access rules are traffic handling rules that define how the traffic is examined and what action the NGFW Engine takes when a rule is matched. The Source, Destination, and Service options can be used to set the matching criteria for the rule. Network packets are accepted automatically without additional processing when connection tracking is enabled. When Strict connection tracking mode is used, the NGFW Engine checks the sequence numbers of the packets in pre-connection establishment states and for RST and FIN packets, and drops packets that are out of sequence. Connections are closed upon completion of the flow (in the case of TCP and FTP) or if there is an inactivity timeout for the session.

Forcepoint NGFW supports the following protocols and their attributes in a firewall policy:

RFC 792 (ICMPv4)
- Type
- Code

RFC 4443 (ICMPv6)
- Type
- Code

RFC 791 (IPv4)
- Source address
- Destination address
- Transport layer protocol

RFC 2460 (IPv6)
- Source address
- Destination address
- Transport layer protocol

RFC 793 (TCP)
- Source port
- Destination port

RFC 768 (UDP)
- Source port
- Destination port

Within each protocol, certain attributes are subject to firewall filtering rules.

With stateful connections, a log entry is created only for the first packet that is seen in the control connection or data connection. It also notes that TCP traffic on port 21 is by default interpreted as FTP protocol traffic. If this control connection is allowed by Access rules and traffic on port 21 contains valid FTP protocol commands to open a data connection, the NGFW engine allows those related data connections and logs them using the same settings

as configured in Access rules for control connections.

The “Create an element for the NGFW Engine” section of the **Admin Guide** provides the steps for creating an NGFW Engine and defining its properties including setting the Strict connection tracking mode via Advanced Settings -> Traffic Handling -> Layer 3 Connection Tracking Mode, select Strict.

The “Create a customized Firewall Policy Template” section of the **Admin Guide** states that packet validity checks automatically drop invalid IP packets, packets with certain IP options, incomplete IP packets, and invalid IP fragments. These dropped packets are also logged when Packet Filter diagnostics have been enabled. The automatic anti-spoofing drops and logs spoofed packets where the source or the destination address is a loopback address, the source address is an IPv4 broadcast address or an IPv4 multicast address, or the source address does not belong to a connected network. The additional Access rules in the customized template discard IPv4 and IPv6 link local addresses, IPv6 reserved addresses, IPv4 and IPv6 addresses reserved for future use, and packets where the source address is an IPv6 multicast address.

The “Create a customized Firewall Policy Template” section also notes that the IPv6 Neighbor Discovery Protocol can be adversely affected when link local IPv6 addresses are discarded as required in the evaluated configuration. It then provides two rules that should be configured before the Ipv6 link local discard rules in order to allow IPv6 Neighbor solicitation and advertisement messages using IPv6 link local addresses.

Testing Assurance Activities: Test 1: The evaluator shall configure the TOE to permit and log TCP traffic. The evaluator shall initiate a TCP session. While the TCP session is being established, the evaluator shall introduce session establishment packets with incorrect flags to determine that the altered traffic is not accepted as part of the session (i.e., a log event is generated to show the ruleset was applied). After a TCP session is successfully established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports, sequence number, flags) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 2: The evaluator shall terminate the TCP session established per Test 1 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 3: The evaluator shall expire (i.e., reach timeout) the TCP session established per Test 1 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 4: The evaluator shall configure the TOE to permit and log UDP traffic. The evaluator shall establish a UDP session. Once a UDP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, source and destination ports) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 5: The evaluator shall expire (i.e., reach timeout) the UDP session established per Test 4 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 6: If ICMP is selected, the evaluator shall configure the TOE to permit and log ICMP traffic. The evaluator shall establish a session for ICMP as defined in the TSS. Once an ICMP session is established, the evaluator shall alter each of the session determining attributes (source and destination addresses, other attributes chosen in FFW_RUL_EXT.1.5) one at a time in order to verify that the altered packets are not accepted as part of the established session.

Test 7: If applicable, the evaluator shall terminate the ICMP session established per Test 6 as described in the TSS. The evaluator shall then immediately send a packet matching the former session definition in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 8: The evaluator shall expire (i.e., reach timeout) the ICMP session established per Test 6 as described in the TSS. The evaluator shall then send a packet matching the former session in order to ensure it is not forwarded through the TOE without being subject to the ruleset.

Test 1: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. Using guidance, the evaluator configured the TOE to allow and log network traffic for a valid TCP session. The evaluator started transmitting packets from a test server to establish a valid TCP session. The packets being sent were constructed such that they would be permitted to pass through the TOE. During the TCP session negotiation, the evaluator transmitted a packet with incorrect flags. The evaluator ensured by examining the TOE logs and viewing captured packets, that the packet with incorrect flags was discarded as not being part of the session. The evaluator also sent packets with incorrect source and destination addresses, incorrect source and destination ports, and incorrect flags and sequence numbers. The evaluator confirmed (through logs and packet captures) that all non-matching packets are not treated as part of the established session. These steps were performed using both IPv4 and IPv6.

Test 2: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. Using guidance, the evaluator configured the TOE to allow and log network traffic for a valid TCP session. The evaluator started transmitting packets from a test server to establish a valid TCP session. The packets being sent were constructed such that they would be permitted to pass through the TOE. The evaluator then terminated the TCP session, and attempted to send additional packets using the same TCP session information. The evaluator ensured by examining the TOE logs and viewing captured packets, that the packet sent after the session termination, were not passed by the TOE as part of the session. These tests were performed using both IPv4 and IPv6.

Test 3: Repeated Test 2, expiring the session rather than explicitly terminating the session, using both IPv4 and IPv6. The evaluator observed that no packets sent after a session expiration were treated by the TOE as part of the original TCP session.

Test 4: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides

of the TOE. Using guidance, the evaluator configured the TOE to allow and log network traffic for a valid UDP session. The evaluator started transmitting packets from a test server to establish a valid UDP session. The packets being sent were constructed such that they would be permitted to pass through the TOE. The evaluator also sent packets with incorrect source and destination addresses and incorrect source and destination ports. The evaluator confirmed (through logs and packet captures) that all nonmatching packets are not treated as part of the established session. These steps were performed using both IPv4 and IPv6.

Test 5: The evaluator performed test 3 (using an expired session) using a UDP session rather than a TCP session.

Test 6: The evaluator established an ICMP session and then sent ICMP packets that did not match the established session (different src IP, dst IP, icmp ID, etc.). The evaluator confirmed that the TOE does not treat the packets as part of the established ICMP session. The TOE only permits packets matching the established ICMP session and unmatching packets were denied.

Test 7: Not applicable, the TOE does not track ICMP “session termination” and only times out/expires ICMP sessions.

Test 8: The evaluator established an ICMP session, then expired/timed out that session, and then sent ICMP packets that matched the terminated session in order to ensure that the TOE treats the packets as part of a new ICMP session. The evaluator confirmed that the TOE does not treat the ICMP packets as part of the expired ICMP session. The TOE treated the packets as part of a new ICMP session (and applied the configured FW rules as appropriate).

2.5.1.6 STFFW14E:FFW_RUL_EXT.1.6

TSS Assurance Activities: The evaluator shall verify that the TSS identifies the following as packets that will be automatically dropped and are counted or logged:

a) Packets which are invalid fragments, including a description of what constitutes an invalid fragment

b) Fragments that cannot be completely re-assembled

c) Packets where the source address is defined as being on a broadcast network

d) Packets where the source address is defined as being on a multicast network

e) Packets where the source address is defined as being a loopback address

f) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address 'reserved for future use' (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4;

g) The TSF shall reject and be capable of logging network packets where the source or destination address of the network packet is defined as an 'unspecified address' or an address 'reserved for future definition and use' (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;

h) Packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified

i) Other packets defined in FFW_RUL_EXT.1.6 (if any).

Section 6.5 of the ST states that the NGFW engine compares the protocol information attributes with the matching criteria of the rule to determine whether to apply the rule. If applied the target actions are implemented and the additional capabilities and flow control rules defined below are applied.

1) The NGFW engine denies and allows logging packets which are invalid fragments;

2) The NGFW engine denies and allows logging fragmented IP packets which cannot be re-assembled completely;

3) The NGFW engine denies and allows logging packets where the source address of the network packet is defined as being on a broadcast network;

4) The NGFW engine denies and allows logging packets where the source address of the network packet is defined as being on a multicast network;

5) The NGFW engine denies and allows logging network packets where the source address of the network packet is defined as being a loopback address;

6) The NGFW engine denies and allows logging packets where the source or destination address of the network packet is defined as being an address 'reserved for future use' as specified in RFC 5735 for IPv4;

7) The NGFW engine denies and allows logging 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' as specified in RFC 3513 for IPv6;

8) The NGFW engine denies and allows logging packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes packets that are discarded and potentially logged by default. If applicable protocols are identified, their descriptions need to be consistent with the TSS. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

The “Create a customized Firewall Policy Template” section of the **Admin Guide** states that packet validity checks automatically drop invalid IP packets, packets with certain IP options, incomplete IP packets, and invalid IP fragments. These dropped packets are also logged when Packet Filter diagnostics have been enabled. The automatic anti-spoofing drops and logs spoofed packets where the source or the destination address is a loopback address, the source address is an IPv4 broadcast address or an IPv4 multicast address, or the source address does not belong to a connected network. The additional Access rules in the customized template discard IPv4 and IPv6

link local addresses, IPv6 reserved addresses, IPv4 and IPv6 addresses reserved for future use, and packets where the source address is an IPv6 multicast address.

The "Create a Firewall Policy" section of the **Admin Guide** provides the steps for creating a firewall policy which include adding access rules and configuring logging for a rule. It states that packets that are automatically rejected are not logged by default and provides specific instructions for enabling the logging of all packets. It also includes a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for creating a customized firewall policy.

The "How firewalls process traffic" section of the **Admin Guide** notes that with stateful connections, a log entry is created only for the first packet that is seen in the control connection or data connection. It also notes that TCP traffic on port 21 is by default interpreted as FTP protocol traffic. If this control connection is allowed by Access rules and traffic on port 21 contains valid FTP protocol commands to open a data connection, the NGFW engine allows those related data connections and logs them using the same settings as configured in Access rules for control connections.

Testing Assurance Activities: Both IPv4 and IPv6 shall be tested for items a), b), c), d), and e) of the SFR element. Both IPv4 and IPv6 shall be tested for item i) unless the rule definition is specific to IPv4 or IPv6. Note: f), g), and h) are specific to IPv4 or IPv6 and shall be tested accordingly.

Test 1: The evaluator shall test each of the conditions for automatic packet rejection in turn. In each case, the TOE should be configured to allow all network traffic and the evaluator shall generate a packet or packet fragment that is to be rejected. The evaluator shall use packet captures to ensure that the unallowable packet or packet fragment is not passed through the TOE.

Test 2: For each of the cases above, the evaluator shall use any applicable guidance to enable dropped packet logging or counting. In each case above, the evaluator shall ensure that the rejected packet or packet fragment was recorded (either logged or an appropriate counter incremented).

Test 1: The evaluator set up a test configuration allowing packets to be captured on both (ingress and egress) sides of the TOE. The evaluator configured firewall rules to allow all network traffic and enabled rejected packet logging. The evaluator tested each condition for automatic packet rejection (as specified in FFW_RUL_EXT.1.6) by generating the types of packets constructed to meet those conditions. The evaluator observed that the TOE rejected and logged all of these packets.

Test 2: This test was performed as part of Test 1 described above.

2.5.1.7 STFFW14E:FFW_RUL_EXT.1.7

TSS Assurance Activities: The evaluator shall verify that the TSS explains how the following traffic can be dropped and counted or logged:

a) Packets where the source address is equal to the address of the network interface where the network packet was received

b) Packets where the source or destination address of the network packet is a link-local address

c) Packets where the source address does not belong to the networks associated with the network interface where the network packet was received, including a description of how the TOE determines whether a source address belongs to a network associated with a given network interface

Section 6.5 of the ST states that the NGFW engine enforces the firewall security policy on all traffic that passes through the engine, via its internal or external network Ethernet interfaces. The traffic is TCP, UDP, ICMPv4, ICPv6, connections over IPv4 and IPv6. The NGFW engine inspects and filters these protocols based upon their header fields as defined by their corresponding RFC (as identified in Table 12 of the ST). The protocol attributes are compared with the matching criteria of the rule to determine whether to apply the rule. Each rule comprises matching criteria and target actions including Allow, Discard and Refuse. If applied the target actions are implemented and additional capabilities and flow control rules are applied. These additional capabilities and flow control rules include the following:

1) The NGFW engine denies and logs network packets where the source address of the network packet is equal to the address of the network interface where the network packet was received;

2) The NGFW engine denies and logs network packets where the source or destination address of the network packet is a link-local address;

3) The NGFW engine denies and logs network packets where the source address of the network packet does not belong to the networks associated with the network interface where the network packet was received, as the Engine has an administrator defined set of networks associated with configured network interfaces.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes how the TOE can be configured to implement the required rules. If logging is configurable, the evaluator shall verify that applicable instructions are provided to configure auditing of automatically rejected packets.

The section “Create a customized Firewall Policy Template” in the **Admin Guide** provides the steps for opening a firewall policy template and creating the required network elements and rules. Packet validity checks automatically drop invalid IP packets, packets with certain IP options, incomplete IP packets, and invalid IP fragments. These dropped packets are also logged when Packet Filter diagnostics have been enabled. The automatic anti-spoofing drops and logs spoofed packets where the source or the destination address is a loopback address, the source address is an IPv4 broadcast address or an IPv4 multicast address, or the source address does not belong to a connected network. The additional Access rules in the customized template discard IPv4 and IPv6 link local addresses, IPv6 reserved addresses, IPv4 and IPv6 addresses reserved for future use, and packets where the source address is an IPv6 multicast address.

The “Create a Firewall Policy” section of the **Admin Guide** includes the steps for creating a firewall policy which includes adding access rules and configuring logging for a rule. It states that packets that are automatically rejected are not logged by default and provides specific instructions for enabling the logging of all packets. It also

includes a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for creating a customized firewall policy.

Testing Assurance Activities: Test 1: The evaluator shall configure the TOE to drop and log network traffic where the source address of the packet matches that of the TOE network interface upon which the traffic was received. The evaluator shall generate suitable network traffic to match the configured rule and verify that the traffic is dropped and a log message generated.

Test 2: The evaluator shall configure the TOE to drop and log network traffic where the source IP address of the packet fails to match the network reachability information of the interface to which it is targeted, e.g. if the TOE believes that network 192.168.1.0/24 is reachable through interface 2, network traffic with a source address from the 192.168.1.0/24 network should be generated and sent to an interface other than interface 2. The evaluator shall verify that the network traffic is dropped and a log message generated.

Test 1: The evaluator configured the TOE to drop and log network traffic where the source address of the packet matches that of the TOE network interface upon which the traffic was received. The evaluator generated suitable traffic to match the configured rule and confirmed via packet capture and logs that the traffic is dropped and a log message is generated.

Test 2: The evaluator configured the TOE to drop and log network traffic where the source IP address of the packet fails to match the network reachability information of the interface to which it is targeted. The evaluator generated suitable traffic to match the configured rule and confirmed via packet capture and logs that the traffic is dropped and a log message is generated. The evaluator then configured the TOE to drop and log network traffic where the source or destination address of the network packet is a link-local address and confirmed that when traffic matching that rule was generated, the TOE dropped the traffic and generated a log message.

2.5.1.8 STFFW14E:FFW_RUL_EXT.1.8

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

Section 6.5 of the ST states that the TOE provides an information flow control mechanism using a rule base that comprises a set of security policy rules, i.e., the firewall security policy. Administrators using the Management Server define the firewall security policy rules.

The NGFW engine enforces the firewall security policy on all traffic that passes through the engine, via its internal or external network Ethernet interfaces. The traffic is TCP, UDP, ICMPv4, ICPv6, connections over IPv4 and IPv6. The NGFW engine inspects and filters these protocols based upon their header fields as defined by their corresponding RFC (as defined in Table 12 of the ST). Any network traffic passed by the NGFW engine must be explicitly allowed by a firewall rule or be part of an established session allowed by a rule, or it is dropped. This is true even in the case of attempts to flood a TOE interface (in which case some packets may be dropped, but are never passed violating policy).

The NGFW engine implements connection tracking to manage the information flow control decisions for connections (i.e., stateful sessions) rather than packets, providing increased performance and support for firewall features that require packet information above the IP level (e.g., ICMP, TCP, UDP). The connection tracking mechanism stores the state information of each connection to allow packets belonging to an established connection to pass. The connection tracking uses protocol attributes when determining whether a packet matches an allowed established session for the corresponding protocol. Connection tracking will eliminate existing connections immediately, upon completion of the flow or upon an inactivity timeout for the session.

The NGFW engine follows a specific orderly algorithm to traverse the rule base for matching and filtering the traffic between its internal and external networks. The rule base is read from top down, and when the first matching rule is encountered the search stops and the TOE executes the matching rule. Any traffic that is not explicitly accepted by the security policy is rejected by the firewall. Each rule comprises matching criteria and target actions. If the matching criteria is verified (i.e., a comparison matches) the NGFW engine applies the target actions. Possible target actions include Allow, Discard and Refuse. Access rules with the logging option, can create a log or alert entry each time they match. The logging option is in addition to the target action of a rule. If applied the target actions are implemented and the additional capabilities and flow control rules as defined in Table 14 of the ST are applied.

During the NGFW Engine boot process, there is a lag between the time when the network interface is operational, and the time that the Stateful Traffic Filtering functionality is fully functioning. During this time, traffic flow through the appliance is disabled; and traffic to and from the appliance is controlled by a Default Filter that drops all external traffic to the appliance.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes how the order of stateful traffic filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

The "How firewalls process traffic" section of the **Admin Guide** refers the reader to the chapter about Access Rules in the *Forcepoint Next Generation Firewall Product Guide* for more information.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 50 (Access Rules) indicates that Rules are read from the top down. The actions Allow, Discard and Refuse stop the processing from continuing down the rule table for any connection that matches the rule. Rules with any of these actions should be placed so that the more limited the rule is in scope, the higher up in the rule table it is. In Firewall policies, traffic that does not match any of the Access rules by the end of the policy is discarded by default.

The "Create a customized Firewall Policy Template" section of the **Admin Guide** describes how to open a Firewall Policy Template, create network elements and then use the protocol attributes to configure the rules to map them to specific network interfaces. Step 3 provides instructions for adding an access rule and also indicates how to move access rules up and down in the policy.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Firewall Policy entry in the table refers the reader to the chapter about creating and managing policy elements in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 48 (Creating and Managing Policy Elements) indicates that the NGFW Engine passes through only traffic that is explicitly allowed in the Firewall Policy. All other traffic is discarded. The NGFW Engine checks a new connection against the policy, rule by rule. The header on each packet arriving on an interface is examined for the source and destination IP address, and protocol-related information, such as the port. The authentication status of the user attempting a connection and the current date and time can also be included as parameters in the examination process.

  Connection tracking means that the engine keeps a record of all currently open connections (stateful inspection). With connection tracking, the engine can verify that the connection proceeds according to the protocol standards. By default, connection tracking is on. The current connection tracking information is checked to see if the packet is part of an established connection (for example, a reply packet to a request that has been allowed). If TCP SYN rate limits or other DoS protection features are enabled, they are enforced at this stage. If the packet is not part of an existing connection, the packet is compared with the Access rules in the installed policy. The processing continues until the packet matches a rule that tells the NGFW Engine to allow or stop the packet. If there is no rule match anywhere else in the policy, the packet is allowed or discarded according to the final action of the policy.

Testing Assurance Activities: Test1: If the TOE implements a mechanism that ensures that no conflicting rules can be configured, the evaluator shall try to configure two conflicting rules and verify that the TOE rejects the conflicting rule(s). It is important to verify that the mechanism is implemented in the TOE but not in the non-TOE environment. If the TOE does not implement a mechanism that ensures that no conflicting rules can be configured, the evaluator shall devise two equal stateful traffic filtering rules with alternate operations - permit and drop. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation. (TD0545 applied)

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

Test 1: The evaluator attempted to configure the TOE (according to the **Admin Guide**) with two firewall rules using the same matching criteria, where one rule would permit while the other deny traffic. The evaluator configured a permit rule first to permit traffic to a specific destination with the second rule (deny). The evaluator confirmed that the connection was successful. Subsequently, the evaluator configured a deny rule first with the second rule (permit). The evaluator observed that the connection failed. The original firewall rules were implemented using IPv4. The test was then repeated using IPv6. Both test cases behaved as expected.

Test 2: Continuing test 1, the evaluator repeated the procedure above, except the evaluator changed the rules to make one a subset of the other, and then tested both orders. The evaluator confirmed that the first is enforced regardless of the specificity of the rule. The original firewall rules were implemented using IPv4. The test was then repeated using IPv6. Both test cases behaved as expected.

2.5.1.9 STFFW14E:FFW_RUL_EXT.1.9

TSS Assurance Activities: The evaluator shall verify that the TSS describes the process for applying stateful traffic filtering rules and also that the behavior (either by default, or as configured by the administrator) is to deny packets when there is no rule match unless another required conditions allows the network traffic (i.e., FFW_RUL_EXT.1.5 or FFW_RUL_EXT.2.1).

Section 6.5 of the ST states that the TOE provides an information flow control mechanism using a rule base that comprises a set of security policy rules, i.e., the firewall security policy. Administrators using the Management Server define the firewall security policy rules.

The NGFW engine enforces the firewall security policy on all traffic that passes through the engine, via its internal or external network Ethernet interfaces. The traffic is TCP, UDP, ICMPv4, ICPv6, connections over IPv4 and IPv6. The NGFW engine inspects and filters these protocols based upon their header fields as defined by their corresponding RFC (as defined in Table 12 of the ST). Any network traffic passed by the NGFW engine must be explicitly allowed by a firewall rule or be part of an established session allowed by a rule, or it is dropped. This is true even in the case of attempts to flood a TOE interface (in which case some packets may be dropped, but are never passed violating policy).

The NGFW engine implements connection tracking to manage the information flow control decisions for connections (i.e., stateful sessions) rather than packets, providing increased performance and support for firewall features that require packet information above the IP level (e.g., ICMP, TCP, UDP). The connection tracking mechanism stores the state information of each connection to allow packets belonging to an established connection to pass. The connection tracking uses protocol attributes when determining whether a packet matches an allowed established session for the corresponding protocol. Connection tracking will eliminate existing connections immediately, upon completion of the flow or upon an inactivity timeout for the session.

The NGFW engine follows a specific orderly algorithm to traverse the rule base for matching and filtering the traffic between its internal and external networks. The rule base is read from top down, and when the first matching rule is encountered the search stops and the TOE executes the matching rule. Any traffic that is not explicitly accepted by the security policy is rejected by the firewall. Each rule comprises matching criteria and target actions. If the matching criteria is verified (i.e., a comparison matches) the NGFW engine applies the target actions. Possible target actions include Allow, Discard and Refuse. Access rules with the logging option, can create a log or alert entry each time they match. The logging option is in addition to the target action of a rule. If applied the target actions are implemented and the additional capabilities and flow control rules as defined in Table 14 of the ST are applied.

During the NGFW Engine boot process, there is a lag between the time when the network interface is operational,

and the time that the Stateful Traffic Filtering functionality is fully functioning. During this time, traffic flow through the appliance is disabled; and traffic to and from the appliance is controlled by a Default Filter that drops all external traffic to the appliance.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the guidance documentation provides the appropriate instructions to configure the behavior to deny packets with no matching rules.

The "How firewalls process traffic" section of the **Admin Guide** states that NGFW Engines permit or deny traffic according to firewall filtering rules that are contained in a Firewall Policy. Each policy is based on a Template Policy. A Template Policy contains necessary predefined rules and also enables automatic rules for the NGFW Engine to communicate with the SMC. A firewall only passes the traffic that is explicitly allowed in the Firewall Policy.

The "How firewalls process traffic" section of the **Admin Guide** refers the reader to the chapter about Access Rules in the *Forcepoint Next Generation Firewall Product Guide* for more information. In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Firewall Policy entry in the table refers the reader to the chapter about creating and managing policy elements in the *Forcepoint Next Generation Firewall Product Guide.* The *Forcepoint Next Generation Firewall Product Guide* is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 48 (Creating and Managing Policy Elements) indicates that the NGFW Engine passes through only traffic that is explicitly allowed in the Firewall Policy. All other traffic is discarded. The NGFW Engine checks a new connection against the policy, rule by rule. The header on each packet arriving on an interface is examined for the source and destination IP address, and protocol-related information, such as the port. The authentication status of the user attempting a connection and the current date and time can also be included as parameters in the examination process.

  Connection tracking means that the engine keeps a record of all currently open connections (stateful inspection). With connection tracking, the engine can verify that the connection proceeds according to the protocol standards. By default, connection tracking is on. The current connection tracking information is checked to see if the packet is part of an established connection (for example, a reply packet to a request that has been allowed). If TCP SYN rate limits or other DoS protection features are enabled, they are enforced at this stage. If the packet is not part of an existing connection, the packet is compared with the Access rules in the installed policy. The processing continues until the packet matches a rule that tells the NGFW Engine to allow or stop the packet. If there is no rule match anywhere else in the policy, the packet is allowed or discarded according to the final action of the policy. In Firewall policies, traffic that does not match any of the Access rules by the end of the policy is discarded by default.

- Chapter 50 (Access Rules) indicates that Rules are read from the top down. The actions Allow, Discard and

Refuse stop the processing from continuing down the rule table for any connection that matches the rule. Rules with any of these actions should be placed so that the more limited the rule is in scope, the higher up in the rule table it is. In Firewall policies, traffic that does not match any of the Access rules by the end of the policy is discarded by default.

Testing Assurance Activities: For each attribute in FFW_RUL_EXT.1.2, the evaluator shall construct a test to demonstrate that the TOE can correctly compare the attribute from the packet header to the ruleset, and shall demonstrate both the permit and deny for each case. It shall also be verified that a packet is dropped if no matching rule can be identified for the packet. The evaluator shall check the log in each case to confirm that the relevant rule was applied. The evaluator shall record a packet capture for each test to demonstrate the correct TOE behaviour.

The evaluator defined several tests variations to exercise the attributes and rules from FFW_RUL_EXT.1.2. The evaluator generated traffic to match specific aspects of the configured firewall rule set and confirmed that all attributes demonstrated permit, deny and log for each test case. The following variations were tested.

Test Variations
Part 1-ICMPv4 Permit Rule - Match Type and Code
Part 2-ICMPv4 Permit Rule - Mismatch Type
Part 3-ICMPv4 Permit Rule - Mismatch Code
Part 4-ICMPv4 Deny Rule - Match Type and Code
Part 5-ICMPv4 Deny Rule - Mismatch Type
Part 6-ICMPv4 Deny Rule- Mismatch Code
Part 7-ICMP6 Permit Rule - Match Type and Code
Part 8-ICMP6 Permit Rule - Mismatch Type
Part 9-ICMP6 Permit Rule - Mismatch Code
Part 10-ICMP6 Deny Rule - Match Type and Code
Part 11-ICMP6 Deny Rule - Mismatch Type
Part 12-ICMP6 Deny Rule - Mismatch Code
Part 13-IPv4 Permit Rule - Match Source/Destination, Protocol and ports
Part 14-IPv4 Permit Rule - Mismatch Source
Part 15-IPv4 Permit Rule - Mismatch Destination
Part 16-IPv4 Permit Rule - Mismatch Protocol
Part 17-IPv4 Deny Rule - Match Source/Destination, Protocol and ports
Part 18-IPv4 Deny Rule - Mismatch Source
Part 19-IPv4 Deny Rule - Mismatch Destination
Part 20-IPv4 Deny Rule - Mismatch Protocol
Part 21-IPv6 Permit Rule - Match Source/Destination, Protocol and ports
Part 22-IPv6 Permit Rule - Mismatch Source
Part 23-IPv6 Permit Rule - Mismatch Destination
Part 24-IPv4 Permit Rule - Mismatch Protocol
Part 25-IPv6 Deny Rule - Match Source/Destination, Protocol and ports

Part 26-IPv6 Deny Rule - Mismatch Source
Part 27-IPv6 Deny Rule - Mismatch Destination
Part 28-IPv6 Deny Rule - Mismatch Protocol
Part 29-TCP Permit Rule - Match IPv4 ports
Part 30-TCP Permit Rule - Mismatch IPv4 Source
Part 31-TCP Permit Rule - Mismatch IPv4 Destination
Part 32-TCP Deny Rule- Match IPv4 ports
Part 33-TCP Deny Rule - Mismatch IPv4 Source
Part 34-TCP Deny Rule - Mismatch IPv4 Destination
Part 35-TCP Permit Rule - Match IPv6 ports
Part 36-TCP Permit Rule - Mismatch IPv6 Source
Part 37-TCP Permit Rule - Mismatch IPv4 Destination
Part 38-UDP Deny Rule- Match IPv6 ports
Part 39-TCP Deny Rule - Mismatch IPv6 Source
Part 40-TCP Deny Rule - Mismatch IPv6 Destination
Part 41-UDP Permit Rule - Match IPv4 ports
Part 42-UDP Permit Rule - Mismatch IPv4 Source
Part 43-UDP Permit Rule - Mismatch IPv4 Destination
Part 44-UDP Deny Rule - Match IPv4 ports
Part 45-UDP Deny Rule - Mismatch IPv4 Source
Part 46-UDP Deny Rule - Mismatch IPv4 Destination
Part 47-UDP Permit Rule - Match IPv6 ports
Part 48-UDP Permit Rule - Mismatch IPv6 Source
Part 49-UDP Permit Rule - Mismatch IPv6 Destination
Part 50-UDP Deny Rule - Match IPv6 ports
Part 51-UDP Deny Rule - Mismatch IPv6 Source
Part 52-UDP Deny Rule - Mismatch IPv6 Destination

2.5.1.10 STFFW14E:FFW_RUL_EXT.1.10

TSS Assurance Activities: The evaluator shall verify that the TSS describes how the TOE tracks and maintains information relating to the number of half-open TCP connections. The TSS should identify how the TOE behaves when the administratively defined limit is reached and should describe under what circumstances stale half-open connections are removed (e.g. after a timer expires).

Section 6.5 of the ST states that the TOE can also track and maintain the number of half-open TCP connections, and the administrator can define a limit of the number of such connections (either for the Engine as a whole or for a specific rule). When the TOE detects that threshold has been exceeded, the TOE denies additional SYN packets. The TOE will expire such half-open TCP connections after fifteen second by default, and the administrator can change this default by configuring the “TCP syn ack seen” timeout.

Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes the behaviour of imposing TCP half-open connection limits and its default state if unconfigured. The evaluator shall

verify that the guidance clearly indicates the conditions under which new connections will be dropped e.g. perdestination or per-client.

In the “Create an element for the NGFW engine” section of the **Admin Guide**, instructions for creating an NGFW Engine element via the GUI are provided and include configuring the “Rate-Based DoS Protection Mode” via the Advanced Settings and then setting a value in the “Limit for Half-Open TCP Connections” option. The limit applies per destination IP address. This option is enabled for all permitted traffic on the NGFW engine but can be overridden for some traffic in the Access rule option in a firewall policy.

Testing Assurance Activities: Test 1: The evaluator shall define a TCP half-open connection limit on the TOE. The evaluator shall generate TCP SYN requests to pass through the TOE to the target system using a randomised source IP address and common destination IP address. The number of SYN requests should exceed the TCP half-open threshold defined on the TOE. TCP SYN-ACK messages should not be acknowledged. The evaluator shall verify through packet capture that once the defined TCP half-open threshold has been reached, subsequent TCP SYN packets are not transmitted to the target system. The evaluator shall verify that when the configured threshold is reached that, depending upon the selection, either a log entry is generated or a counter is incremented.

Test 1: The TOE was configured with a SYN threshold of 200. The evaluator sent a stream of 200 SYN packets and after a 1 second delay, sent another 200 packets. The evaluator noted that the TOE only passed through a portion of packets and blocked the rest of the traffic once it detected that the SYN packet threshold had been reached. This test was performed with both IPv4 and IPv6 SYN packets.

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

The evaluator shall verify that the TSS also include a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describe the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets. The description shall also include a description how the TOE behaves in the situation where the traffic exceeds the amount of traffic the TOE can handle and how it is ensured that also in this condition stateful traffic filtering rules are still applied so that traffic does not pass that shouldn't pass according to the specified rules.

Section 6.5 of the ST states that the NGFW Engine obtains time values from the local hardware clock when making the security policy decisions associated with time-based information flows. During the NGFW Engine boot process, there is a lag between the time when the network interface is operational, and the time that the Stateful Traffic Filtering functionality is fully functioning. During this time, traffic flow through the appliance is disabled; and traffic to and from the appliance is controlled by a Default Filter that drops all external traffic to the appliance.

All received network packets are processed by the NGFW Engine software module before transmission. The NGFW

software module does stateful filtering of the received network packets according to the configured traffic filtering rules. Protocol Agents are used for advanced processing of traffic that require special handling such as permitting an FTP data connection dynamically. The NGFW Engine software denies the traffic if the Protocol Agent cannot process the traffic. Incoming packets are dropped if a network packet cannot be processed due to insufficient memory. All incoming network packets are also discarded before the NGFW Engine software module has been loaded, and the NGFW Engine software module denies all traffic until the module has been configured. Network interfaces and routing are configured after the NGFW Engine software module has been loaded. If the configured firewall rules cannot be applied during startup, only the management network interface will be available and traffic through the firewall will be denied.

Any network traffic passed by the NGFW Engine must be explicitly allowed by a firewall rule or be part of an established session allowed by a rule, or it is dropped. This is true even in the case of attempts to flood a TOE interface (in which case some packets may be dropped, but are never passed violating policy).

Component Guidance Assurance Activities: The guidance documentation associated with this requirement is assessed in the subsequent test evaluation activities.

Please refer to the subsequent test assurance activities below.

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

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

Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test evaluation activities.

Test 1: The evaluator configured the TOE to block network traffic directed at a specific destination. The evaluator started transmitting packets from a test server. The packets being sent were constructed such that they should be blocked by the configured rule. The evaluator rebooted the TOE and stopped the packet capture once the TOE was booted up and presented the login prompt. The evaluator observed that during the reboot the TOE did not permit any of the generated network traffic through the firewall.

Test 2: The evaluator configured the TOE to permit network traffic directed at a specific destination. The evaluator started transmitting packets from a test server. The packets being sent were constructed such that they

should be permitted by the configured rule. The evaluator rebooted the TOE and stopped the packet capture once the TOE was booted up and presented the login prompt. The evaluator observed that during the reboot the TOE did not permit any of the generated network traffic through the firewall. Once initialization was complete, the evaluator observed in the packet captures that traffic flowed through the TOE and that the rules were properly enforced.

2.5.2 Stateful Filtering of Dynamic Protocols (STFFW14E:FFW_RUL_EXT.2)

2.5.2.1 STFFW14E:FFW_RUL_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS identifies the protocols that can cause the automatic creation of dynamic packet filtering rules. In some cases rather than creating dynamic rules, the TOE might establish stateful sessions to support some identified protocol behaviors.

The evaluator shall verify that the TSS explains the dynamic nature of session establishment and removal. The TSS also shall explain any logging ramifications.

The evaluator shall verify that for each of the protocols selected, the TSS explains the dynamic nature of session establishment and removal specific to the protocol.

Section 6.5 of the ST states that the NGFW engine implements connection tracking to manage the information flow control decisions for connections (i.e., stateful sessions) rather than packets, providing increased performance and support for firewall features that require packet information above the IP level (e.g., ICMP, TCP, UDP). The connection tracking mechanism stores the state information of each connection to allow packets belonging to an established connection to pass. The connection tracking uses protocol attributes when determining whether a packet matches an allowed established session for the corresponding protocol. Connection tracking will eliminate existing connections immediately, upon completion of the flow or upon an inactivity timeout for the session.

Connection tracking works closely with the protocol agents to manage the information flow control decisions based on information attributes at the different networking layers through the application layer to decide whether a packet should be granted access or not. The following protocol agents and their security function are within the scope of the evaluation: FTP (RFC 959).

The FTP Protocol Agent keeps track of the ports used in File Transfer Protocol (FTP) sessions. An FTP session starts

with a control connection (by default, TCP port 21), and the communications continue using a dynamically allocated port. The FTP Protocol Agent opens the actual ports used in FTP sessions as needed so that the whole range of possible dynamic ports does not need to be allowed in the policy.

The NGFW Engine follows a specific orderly algorithm to traverse the rule base for matching and filtering the traffic between its internal and external networks. Each rule comprises matching criteria and target actions. If the matching criteria is verified (i.e., a comparison matches) the NGFW Engine applies the target actions. Possible target actions include Allow, Discard and Refuse. Access rules with the logging option, can create a log or alert entry each time they match. The logging option is in addition to the target action of a rule.

Component Guidance Assurance Activities: The evaluator shall verify that the guidance documentation describes dynamic session establishment capabilities.

The evaluator shall verify that the guidance documentation describes the logging of dynamic sessions consistent with the TSS.

The "How firewalls process traffic" section of the **Admin Guide** describes how access rules are used to define how the traffic is examined and what action the NGFW engine will take when a rule is matched. With stateful connections, a log entry is created only for the first packet that is seen in the control connection or data connection. TCP traffic on port 21 is by default interpreted as FTP protocol traffic. If this control connection is allowed by Access rules and traffic on port 21 contains valid FTP protocol commands to open a data connection, the NGFW engine allows those related data connections and logs them using the same settings as configured in Access rules for control connections.

The "How firewalls process traffic" section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Product Guide* for further information on the FTP Protocol Agent and multi-layer inspection.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 2, section "Support for multi-layer inspection" - this section states that Multi-Layer inspection combines application layer inspection, stateful inspection, and packet filtering technologies flexibly for optimal security and system performance. Like stateful inspection, the Forcepoint NGFW in the Firewall/VPN role uses state tables to track connections and judge whether a packet is a part of an established connection. Forcepoint NGFW also features application-layer inspection through specific Protocol Agents for enhanced security to inspect all data up to the application layer. Forcepoint NGFW in the Firewall/VPN role can also act as a packet filter for types of connections that do not require the security considerations of stateful inspection. By default, all Firewall Access rules implement stateful inspection.

  Forcepoint NGFW in the Firewall/VPN role applies application level inspection with or without proxying the connections, depending on what is required. Protocol Agents provide protocol validation for specific protocols (e.g. FTP). Protocol Agents are also used to handle protocols that generate complex connection patterns, to redirect traffic to proxy services, and to change data payload if necessary.

- Chapter 56, section “Define FTP Protocol parameters” - this section provides instructions for configuring the FTP Protocol Agent. It states that the FTP Protocol Agent keeps track of the ports used in File Transfer Protocol (FTP) sessions. An FTP session starts with a control connection (by default, TCP port 21), and the communications continue using a dynamically allocated port. The FTP Protocol Agent can open the actual ports used in FTP sessions as needed so that the whole range of possible dynamic ports does not need to be allowed in the policy.

Component Testing Assurance Activities: Test 1: The evaluator shall define stateful traffic filtering rules to permit and log traffic for each of the supported protocols and drop and log TCP and UDP ports above 1024. Subsequently, the evaluator shall establish a connection for each of the selected protocols in order to ensure that it succeeds. The evaluator shall examine the generated logs to verify they are consistent with the guidance documentation.

Test 2: Continuing from Test 1, the evaluator shall determine (e.g., using a packet sniffer) which port above 1024 opened by the control protocol, terminate the connection session, and then verify that TCP or UDP (depending on the protocol selection) packets cannot be sent through the TOE using the same source and destination addresses and ports.

Test 3: For each additionally supported protocol, the evaluator shall repeat the procedure above for the protocol. In each case the evaluator must use the applicable RFC or standard in order to determine what range of ports to block in order to ensure the dynamic rules are created and effective.

These tests were iterated for two variations as follows: IPv4 and IPv6.

Test 1: The evaluator first configured the TOE and created two firewall rules - the first rule allows FTP traffic (TCP, port 21) for a specific destination. A second rule (to block TCP traffic for ports above 1024) was not needed for this TOE (as the default reject rule covered this). The evaluator then used a test server to attempt FTP connections through the TOE to an FTP server. An FTP session was successfully established through the TOE. Packet captures were obtained for this connection. The packet captures show that the TOE did not block any of the FTP related connections, even though the TOE was configured with a rule to block TCP packets with a destination port great than 1024. In other words, the TOE correctly, dynamically allowed establishment of FTP data sessions.

Test 2: Continuing from test 1, after closing the FTP session, the evaluator attempted to send TCP packets through the FTP session data ports identified. The evaluator observed that the TOE correctly blocked the attempts to reuse the FTP data ports.

Test 3: Not applicable. The TOE does not claim support for any other protocols.

2.6 IDENTIFICATION AND AUTHENTICATION (FIA)

2.6.1 AUTHENTICATION FAILURE MANAGEMENT (NDCPP22E:FIA_AFL.1)

2.6.1.1 NDCPP22E:FIA_AFL.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.1.2 NDCPP22E:FIA_AFL.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.6 of the ST states that the TOE allows the administrator to specify a maximum number of incorrect logins as well as lock out period for an administrator who exceeds the maximum configured value. The administrator can set the number of failed attempts to a value from 1-1000 and the lockout duration from 1-1000 (and choose from minutes, hours, days). The TOE defaults to 6 incorrect attempts and a 30 minute lock out period. The local CLI remains available when the remote account is locked out.

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

The evaluator shall examine the guidance documentation to confirm that it describes, and identifies the importance of, any actions that are required in order to ensure that administrator access will always be

maintained, even if remote administration is made permanently or temporarily unavailable due to blocking of accounts as a result of FIA_AFL.1.

The Administrative Logins table entry in the "Configure settings for an evaluated configuration" section of the **Admin Guide** provides instructions to enable temporarily locking administrator accounts after a certain amount of failed logon attempts. This is performed on the Password Policy tab via Menu ->System Tools -> Global System Properties. If the administrator account is locked, it is still possible to log on to the SMC Appliance through the local console.

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

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

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

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

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

Test 1: The evaluator performed lockout testing with two different values to verify the configuration works correctly. The first case was performed with an invalid attempt threshold of 3 attempts and a lockout time of 1 minute. The second case was performed with an invalid attempt threshold of 5 attempts and a lockout time of 3 minutes. The evaluator confirmed in each case that once the authentication attempts limit is reached, authentication attempts with valid credentials are no longer successful.

Test 2: The evaluator performed this test after the lockouts occurred in test 1 above. In the first case, the evaluator waited for one minute after the lockouts and then attempted to logon to the TOE with valid credentials. The logon was successful demonstrating that the user was no longer locked out. In the second case, evaluator waited for three minutes after the lockouts and then attempted to logon to the TOE with valid credentials. The logon was successful demonstrating that the user was no longer locked out.

2.6.2 Password Management (NDcPP22e:FIA_PMG_EXT.1)

2.6.2.1 NDcPP22e:FIA_PMG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that it contains the lists of the supported special character(s) and minimum and maximum number of characters supported for administrator passwords.

Section 6.6 of the ST states that the TOE authenticates local and remote administrative users by means of a local password mechanism. Passwords can be composed of upper or lower case letters, numbers, and special characters including “!”, “@”, “#”, “$”,“%”, “^”, “&”, “*”, “(” and “)”. Administrators can specify a minimum length of between 1 and 80 characters for passwords. By default, the TOE enforces a minimum password length of 10 characters. When operating in a Common Criteria evaluated configuration, the recommended minimum password length is 15 characters.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to determine that it:

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

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

The “Configure settings for an evaluated configuration” section of the **Admin Guide** states that when setting a password, you should select a password that meets the following requirements:

- Minimum ten characters long
- At least one uppercase character
- At least one number
- At least one special character: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")

- Cannot be the same as the user name

By default, Forcepoint NGFW enforces a minimum password length of 10 characters. The minimum password length is configurable from 1 to 80 characters. The recommended minimum length is 15 characters. The 'Minimum Number of Required Characters' setting is used to configure the minimum password length.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Password Guidelines entry in the table refers the reader to further information on the topic about enabling and defining password policy settings in the Forcepoint *Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) - subsection "Enable and define password policy settings" provides the steps for configuring and/or changing password policy settings. After accessing the Global System Properties Menu and clicking on the Password Policy tab, the user is instructed to click on *Enforce Password Settings for All the Administrators and Web Portal Users* and then select the password policy settings. At this point, the product guide refers back to the **Admin Guide** for the options that must be selected in the evaluated configuration. As noted above, the **Admin Guide** recommends that the "Minimum Number of Required Characters" setting should be used to configure a minimum password length of 15.

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

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

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

Test 1: The evaluator successfully set valid user passwords with compositions including the following:

-Min configurable length (10 characters)
-Sample min length (30 characters)
-Sample maximum configurable length (80 characters)
-Demonstrating special character set
-Demonstrating number set
-Demonstrating uppercase set

-Demonstrating lowercase set

Test 2: The evaluator attempted to set invalid user passwords with compositions including the following. All invalid password attempts were rejected.

-short password
-no uppercase letters
-no lowercase letters
-no numbers
-no special characters

2.6.3 Protected Authentication Feedback (NDcPP22e:FIA_UAU.7)

2.6.3.1 NDcPP22e:FIA_UAU.7.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

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

Section “Enable FIPS mode on the SMC Appliance” in the **Admin Guide** indicates that the password is not shown while logging on to the local console. No configuration or preparatory steps are required.

Component Testing Assurance Activities: The evaluator shall perform the following test for each method of local login allowed:

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

Test 1- The evaluator observed during testing that passwords are obscured on the console login. Actual results were provided in FIA_UIA_EXT.1-t1.

2.6.4 Password-based Authentication Mechanism (NDcPP22e:FIA_UAU_EXT.2)

2.6.4.1 NDcPP22e:FIA_UAU_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

See FIA_UIA_EXT.1

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

See FIA_UIA_EXT.1

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

See FIA_UIA_EXT.1

2.6.5 User Identification and Authentication (NDcPP22e:FIA_UIA_EXT.1)

2.6.5.1 NDcPP22e:FIA_UIA_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.5.2 NDcPP22e:FIA_UIA_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

Section 6.7 of the ST states that the SMC Appliance offers two administrative interfaces - command line and GUI. The NGFW Engine provides no administrator access at all. The SMC Appliance offers command line functions which are accessible via the CLI, a text based interface which can be accessed from the virtual machine console in the VMware Host client. The TOE also offers access through the GUI client using TLS v1.2. The GUI client provides all management functionality except the limited commands available through the CLI.

The TOE requires that the administrator perform all configuration through the SMC which then communicates with the Engines under its control). The Engines provide no direct interface for administrators in the evaluated configuration.

Section 6.6 of the ST states that the TOE authenticates local and remote administrative users by means of a local password mechanism. Prior to login, the TOE displays a warning banner on both the GUI and the local console

interface. The TOE also supports the filtering and forwarding of network traffic through the NGFW Engine prior to an administrative user being authenticated. The TOE requires login prior to allowing any TOE configuration actions. The SMC Management Server only accepts TLSv1.2 connections for management operations.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document describes how to enable FIPS mode and 256-bit encryption as the security strength during TOE installation, in order to restrict the cryptographic algorithms available in the TOE to those that are compatible with FIPS 140-2.

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** also references the instructions to install the SMC Appliance software on a virtualization platform. It instructs the administrator to access the appliance using the virtual machine console in the VMware Host Client.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table states that the Management client should be used to manage users and passwords in the SMC. The local console user accounts are synchronized with the user accounts used in the SMC and are managed from the SMC. The local console accounts and passwords are managed from the SMC. Only SMC user accounts with unrestricted permissions are available on the SMC Appliance local console. This section also refers the reader to the chapter about Administrator accounts in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide

- Chapter 21 (Administrator Accounts) provides instructions for creating administrative users under the "Adding administrator accounts" subsection. This includes configuring administrator authentication to use a password stored in the internal database of the SMC and configuring an administrator with unrestricted permissions which allows logon via the local console.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the TLS ciphersuite set. The reader is referred to the *Forcepoint Next Generation Firewall Product Guide* for instructions to enable SMC Web Access.

Forcepoint Next Generation Firewall Product Guide

- Chapter 25 (Using the Management Client in a web browser) provides the detailed steps for enabling SMC Web Access on the Management Server in order to run the Management Client in a web browser.

In the "Configure settings for an evaluated configuration" section in the **Admin Guide,** the Password Guidelines entry in the table refers the reader to further information on the topic about enabling and defining password policy settings in the Forcepoint *Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) - subsection "Enable and define password policy settings" provides the steps for configuring and/or changing password policy settings. After accessing the Global System Properties Menu and clicking on the Password Policy tab, the user is instructed to click on *Enforce Password Settings for All the Administrators and Web Portal Users* and then select the password policy settings. At this point, the product guide refers back to the **Admin Guide** for the options that must be selected in the evaluated configuration. The **Admin Guide** indicates that the "Minimum Number of Required Characters" setting should be used to configure a minimum password length of 15.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Logon Banner entry in the table refers to the chapter about using the Management Client in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 7 (Using the Management Client) -The "Create logon banners for administrators" subsection provides instructions for creating a banner text showing all administrators information about the selected Management Server. The text from the banner also appears in the logon window of the local console.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins

entry in the table provides the local console command line command to specify the timeout to terminate an inactive local administrative session. For information about setting timeouts via the Client GUI, it provides a reference to the specific section in the *Forcepoint Next Generation Firewall Product Guide* where the instructions can be found. This document is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) provides instructions for enabling and defining the password policy which includes defining session limits and idle timeouts.

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

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

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

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

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

Test 1 - The evaluator configured the TOE for local console access and for remote access. The evaluator then performed an unsuccessful and successful logon of each type using bad and good credentials respectively.

Test 2 - The evaluator observed that there were no other services available prior to administrator login other than the ability to enter the username and password, display and acknowledge the banner and passing network traffic through the firewall.

Test 3 - This test was performed as part of Test 2 above where the evaluator confirms this behavior. The TOE does not allow any activity prior to login locally or remotely except the operations specified in the requirement.

Test 4 - An administrator is able to log on to the TOE in two ways, through the virtual machine console in the VMware Host client, and through the Management Client GUI interface using TLS (tested as a part of FCS_TLSS). These methods were demonstrated as part of test 1 above. There is no administrator access to the NGFW Engines, thus the evaluator has no ability to locally or remotely connect with the NGFW Engine in FIPS mode.

2.6.6 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/ITT)

2.6.6.1 NDcPP22e:FIA_X509_EXT.1.1/ITT

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the device. The evaluator shall perform the following tests for FIA_X509_EXT.1.1/ITT:

These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols.:

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function, and shall use this chain to demonstrate that the function succeeds.

Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store).

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

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

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the TOE certificate and revocation of the TOE intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function fails. No testing is required if no revocation method is selected. Revocation checking is only applied to certificates

that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

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

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

f) Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

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

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

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

Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed

by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

(TD0527 12/2020 update applied)

The tests below were iterated as follows:

Client Implementation

- Engine ITT Communication
- SMC ITT Communication.

Server Implementation

- Engine ITT Communication
- SMC ITT Communication.

Test 1a: The successful connection with a valid cert chain was demonstrated in FCS_TLSC_EXT.1.

Test 1b: The evaluator configured the TOE without the trusted root CA used by the test server to anchor all of its certificates. The evaluator then attempted to connect the TOE client to the test server and confirmed that the connection was rejected.

Test 2: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection only succeeded if there are no expired certificates. Next, the evaluator alternately configured a test client to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the test client to the TLSS TOE server and confirmed that the connection only succeeded if there are no expired certificates.

Test 3: Not applicable. Revocation checking is not supported as part of distributed TOE communication.

Test 4: Not applicable. Revocation checking is not supported as part of distributed TOE communication.

Test 5, 6, 7: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection only succeeded if the certificate is not modified/corrupted. Next, the evaluator alternately configured a test client to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the test client to the TLSS TOE server and confirmed that the connection only succeeded if the

certificate is not modified/corrupted.

Test 8: For this test, the evaluator alternately configured stunnel on a test server to send an authentication certificate issued by a Sub CA with a valid elliptic curve and an explicitly defined elliptic curve. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection was rejected in the second case using the explicitly defined elliptic curve.

2.6.6.2 NDcPP22e:FIA_X509_EXT.1.2/ITT

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall perform the following tests for FIA_X509_EXT.1.2/ITT. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/ITT. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted. The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation). For each of the following tests the evaluator shall create a chain of at least two certificates: a self-signed root CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

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

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

The tests below were iterated as follows:

Client Implementation

- Engine ITT Communication

- SMC ITT Communication.

Server Implementation

- Engine ITT Communication
- SMC ITT Communication.

Test 1 and Test 2: For this test, the evaluator alternately configured a test server to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection was rejected. Next, the evaluator alternately configured a test client to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false. In each case, the evaluator then attempted to connect the test client to the TLSS TOE server and confirmed that the connection was rejected.

Component TSS Assurance Activities: The evaluator shall examine the TSS to ensure it describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). If selected, the TSS shall describe how certificate revocation checking is performed. It is not sufficient to verify the status of a X.509 certificate only when it's loaded onto the device.

Section 6.6 in the ST states that certificates are validated as part of the authentication process when they are presented to the TOE and when they are loaded into the TOE. The following fields are verified as appropriate: signature, validity period, extended key usage, issuer's name, basic constraints (for CA certs). The TOE does not support revocation for internal TOE communications between distributed TOE components.

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

Section "Forcepoint NGFW System" in the **Admin Guide** states that the Forcepoint NGFW system does not support revocation for internal target of evaluation communications between distributed Forcepoint NGFW system components. Certificates are validated as part of the authentication process when they are presented to the Forcepoint NGFW system and when they are loaded into the Forcepoint NGFW system. The following fields are verified as appropriate: signature, validity period, extended key usage, issuer's name, basic constraints (for CA certificates). The TOE does not support certificate revocation checking between distributed TOE components.

Component Testing Assurance Activities: None Defined

2.6.7 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/Rev)

2.6.7.1 NDcPP22e:FIA_X509_EXT.1.1/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

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

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

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

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

f) Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

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

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

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

Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

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

(TD0527 12/2020 update applied)

These tests were repeated for TLSC (syslog) and for IPsec (IKEv2 auth)

Test 1a: For TLS, the successful connection with a valid cert chain was demonstrated in FCS_TLSC_EXT.1. For IPsec, the evaluator configured the TOE with the trusted root CA used by the test server to anchor all of its certificates. The evaluator then attempted to make an IPsec connection between the TOE and test server and observed the TOE accept the connection.

Test 1b: The evaluator configured the TOE without the trusted root CA used by the test server to anchor all of its certificates. The evaluator then attempted to connect the TOE to the test server and confirmed that the connection was rejected.

Test 2: For this test, the evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection only succeeded if there are no expired certificates.

Test 3: For this test, the evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that is revoked, and 3) issued by an intermediate CA that is revoked. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection succeeded only if there are no revoked certificates. This test was performed for 2 variations as follows: CRL and OCSP.

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

Next, the evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that has a root that refers to an OCSP revocation server where the signer lacks OCSPSigning, 3) issued by an intermediate CA whose issuer CA refers to an OCSP revocation server where the signer lacks OCSPSigning, and 3) issued by an intermediate CA referring to an OCSP revocation server where the signer lacks OCSPSigning. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection only succeeded if all retrieved OCSP responses are signed using certificates with OCSPSigning.

Test 5: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection only succeeded if the certificate is not modified/corrupted.

Test 6 - This test was performed in Test 5 above.

Test 7- This test was performed in Test 5 above.

Test 8a-b: For this test, the evaluator alternately configured a test server to send an authentication certificate issued by a Sub CA with a valid elliptic curve and an explicitly defined elliptic curve. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection was rejected in the second case using the explicitly defined elliptic curve.

Test 8c: The evaluator attempted to load subordinate CA certificate with the elliptic curve parameters specified as a named curve and signed by a trusted root CA. The evaluator observed that the certificate was accepted into the TOE's trust store. Next the evaluator attempted to load a subordinate CA certificate with an explicit format version of the elliptic curve parameters and signed by a trusted root CA. The evaluator observed that the certificate was rejected and not added to the TOE's trust store.

2.6.7.2 NDcPP22e:FIA_X509_EXT.1.2/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation). For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

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

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

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

These tests were repeated for TLSC (syslog) and for IPsec (IKEv2 auth)

Test 1 - The evaluator alternately configured a test server to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection was rejected in each case.

Test 2- This test was performed in Test 1 above.

Component TSS Assurance Activities: The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

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

Section 6.6 of the ST states the TOE supports OCSP and CRL revocations for X509v3 certificate validation during negotiation of TLS protected syslog and during IKEv2 authentication. The TOE does not support revocation for internal TOE communications between distributed TOE components. Certificates are validated as part of the authentication process when they are presented to the TOE and when they are loaded into the TOE. The following fields are verified as appropriate: signature, validity period, extended key usage, issuer's name, basic constraints (for CA certs).

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

In section "Configure settings for an evaluated configuration" of the **Admin Guide**, the Audit Server Configuration table entry states that Forcepoint NGFW supports OCSP and CRL revocations for X509v3 certificate validation during negotiation of TLS protected syslog. When handling a certificate bearing OCSP revocation but where Forcepoint NGFW cannot establish a connection with the OCSP responder, Forcepoint NGFW will not accept the certificate (and thus not establish the connection). When handling certificates bearing CRL information but where Forcepoint NGFW cannot establish a connection to the CRL Distribution Point location, Forcepoint NGFW will not accept the certificate as valid. Forcepoint NGFW constructs the certificate path to a trusted certificate, and then verifies the signature, checks the revocation status, validity period, issuer's name, extended key usage and basic constraints for each certificate starting from the trusted certificate.

The "Define additional VPN certificate authorities" section of the **Admin Guide** describes the TOE's support for OSCP and CRL revocations checking when validating X.509v3 certificates and chains.

Forcepoint NGFW supports OCSP and CRL revocations for X.509v3 certificate validation during negotiation of IPsec VPN.

Forcepoint NGFW constructs the certificate path to a trusted certificate, and then verifies the signature, checks the revocation status, validity period, issuer's name, extended key usage and basic constraints for each certificate starting from the trusted certificate.

- Revocation status checks using OCSP and CRLs can be enabled independently for each trusted CA.
- The settings are applied to the whole certificate chain excluding the trust anchor.
- When OCSP is enabled but the certificate is not bearing OCSP responder information, or Forcepoint NGFW cannot establish a connection with the OCSP responder, revocation status cannot be determined using OCSP.
- When CRLs are enabled but the certificate is not bearing CRL information, or Forcepoint NGFW cannot establish a connection to the CRL Distribution Point location, revocation status cannot be determined using a CRL.
- When both OCSP and CRLs are enabled but Forcepoint NGFW cannot determine revocation status using OCSP, revocation status is checked using a CRL.
- When either OCSP or CRLs are enabled and Forcepoint NGFW cannot determine the revocation status, Forcepoint NGFW will not accept the certificate as valid

Component Testing Assurance Activities: None Defined

2.6.8 X.509 Certificate Validation (VPNGW12:FIA_X509_EXT.1/Rev)

2.6.8.1 VPNGW12:FIA_X509_EXT.1.1/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

See NDcPP22e:FIA_X509_EXT.1/Rev

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.6.9 X.509 Certificate Authentication (NDcPP22e:FIA_X509_EXT.2)

2.6.9.1 NDcPP22e:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.9.2 NDcPP22e:FIA_X509_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.6 of the ST states that the TOE supports OCSP and CRL revocations for X509v3 certificate validation

during negotiation of TLS protected syslog and during IKEv2 authentication. The TOE does not support revocation for internal TOE communications between distributed TOE components. Certificates are validated as part of the authentication process when they are presented to the TOE and when they are loaded into the TOE. The following fields are verified as appropriate: signature, validity period, extended key usage, issuer's name, basic constraints (for CA certs).

The TOE performs revocation checking when validating a server certificate during TLS establishment with a remote syslog server or when authenticating an IKEv2 peer's certificate. The TOE performs no revocation checking as part of distributed TOE TLS communications. When handling a certificate bearing OCSP revocation but where the TOE cannot establish a connection with the OCSP responder, the TOE will not accept the certificate (and thus not establish the connection). When handling certificates bearing CRL information but where the TOE cannot establish a connection to the CRL Distribution Point location, the TOE will not accept the certificate as valid. The TOE constructs the certificate path to a trusted certificate, and then verifies the signature, checks the revocation status, validity period, issuer's name, extended key usage and basic constraints for each certificate starting from the trusted certificate.

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

The "Configure settings for an evaluated configuration" section of the **Admin Guide** provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table also provides further guidance and specific options that should be selected and defined when following the instructions in the online guide to enable TLS v1.2 protection with revocation enabled. It outlines the steps for configuring a trusted Root CA and generating a client certificate request which includes selecting an RSA with key size 2048 bits or greater, or ECDSA with 521 bits for P-521, 384 bits for P-384 or 256 bits for P-256. This section also provides the list of approved TLS cipher suites and instructs the user to ensure that the TLS cipher suites selected match the RSA and ECDSA parameters that are configured. When using an ECDHE cipher suite, P-521, P-384 and P-256 are automatically used in the TLS key establishment.

To use certificate revocation checks, peer certificates must contain the correct CRL Distribution Points extension that refers to a valid CRL Distribution point. The environment must be configured so that the SMC can access the referenced CRL distribution points. If a TLS connection cannot be established because the connection to the CRL server fails, the administrator is instructed to verify the network path to the CRL server and the status of the server, and fix any issues.

The "Define additional VPN certificate authorities" section of the **Admin Guide** describes the TOE's support for OSCP and CRL revocations checking when validating X.509v3 certificates and chains.

Forcepoint NGFW supports OCSP and CRL revocations for X.509v3 certificate validation during negotiation of IPsec VPN.

Forcepoint NGFW constructs the certificate path to a trusted certificate, and then verifies the signature, checks the revocation status, validity period, issuer's name, extended key usage and basic constraints for each certificate starting from the trusted certificate.

- Revocation status checks using OCSP and CRLs can be enabled independently for each trusted CA.
- The settings are applied to the whole certificate chain excluding the trust anchor.
- When OCSP is enabled but the certificate is not bearing OCSP responder information, or Forcepoint NGFW cannot establish a connection with the OCSP responder, revocation status cannot be determined using OCSP.
- When CRLs are enabled but the certificate is not bearing CRL information, or Forcepoint NGFW cannot establish a connection to the CRL Distribution Point location, revocation status cannot be determined using a CRL.
- When both OCSP and CRLs are enabled but Forcepoint NGFW cannot determine revocation status using OCSP, revocation status is checked using a CRL.
- When either OCSP or CRLs are enabled and Forcepoint NGFW cannot determine the revocation status, Forcepoint NGFW will not accept the certificate as valid.

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

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

These tests were repeated for TLSC (syslog) and for IPsec (IKEv2 auth)

Test 1: The evaluator alternately configured a test server to send an authentication certificate with valid/accessible revocation servers and an authentication certificate with revocation information referring to an inaccessible

revocation server. In each case, the evaluator then attempted to connect the TOE to the test server and confirmed that the connection is only successful when the revocation server is accessible and unsuccessful when the revocation server is not accessible.

2.6.10 X.509 Certificate Authentication (VPNGW12:FIA_X509_EXT.2)

2.6.10.1 VPNGW12:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.10.2 VPNGW12:FIA_X509_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to support its use for IPsec at a minimum. The evaluator shall ensure that all evaluation of this SFR is performed against its use in IPsec communications as well as any other supported usage.

See NDcPP22e:FIA_X509_EXT.2

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.6.11 X.509 Certificate Requests (NDcPP22e:FIA_X509_EXT.3)

2.6.11.1 NDcPP22e:FIA_X509_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.11.2 NDcPP22e:FIA_X509_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the ST author selects 'device-specific information', the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

Section 6.6 of the ST states that the TOE generates certificate requests and validates the CA used to sign the certificate. The TOE generates certificate requests which include public key, Common Name, Organization, Organizational Unit, Country and device-specific information in the form of Subject Alternative Name.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server configuration entry in the table provides the steps for enabling TLS protection for communications with the external syslog server. For generating a client certificate request, step 3 references the *Forcepoint Next Generation Firewall Product Guide* for detailed instructions, but does provide the specific options which should be selected for the key sizes and the TLS cipher suite. The *Forcepoint Next Generation Firewall Product Guide* can be accessed via the link found in the "Supporting Documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity. The subsection 'Create a certificate request' provides the instructions for generating a client certificate request including establishing the 'Common Name' and other fields.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server configuration entry in the table states that the target of evaluation generates certificate requests that include a public key, Common Name, Organization, Organizational Unit, Country and device-specific information in the form of Subject Alternative Name.

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

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

b) Test 2: The evaluator shall demonstrate that validating a response message to a Certification Request without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the response message, and demonstrate that the function succeeds.

These tests were repeated for TLSC (syslog) and for IPsec (IKEv2 auth)

Test 1- The evaluator generated a certificate signing request on the TOE as part of testing. The evaluator followed the instructions in the **Admin Guide** when generating the request. The evaluator viewed the contents of the CSR and confirmed that it provided public key and other required information including the information that the evaluator supplied during the generation process.

Test 2 - The evaluator successfully installed the certificate resulting from the CSR request in test 1. Next the evaluator attempted to import a certificate that was signed by a root CA that was not installed on the TOE and attempted to import the resulting certificate to the TOE. The attempt failed.

2.7 Security management (FMT)

2.7.1 Management of Security Functions Behaviour (NDcPP22e:FMT_MOF.1/Functions)

2.7.1.1 NDcPP22e:FMT_MOF.1.1/Functions

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

For non-distributed TOEs, the evaluator shall ensure the TSS 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).

Section 6.7 of the ST states that the SMC Appliance offers two administrative interfaces - command line and GUI (the NGFW Engine provides no administrator access at all). The SMC Appliance offers command line functions which are accessible via the CLI. The CLI is a text based interface which can be accessed from the virtual machine console in the VMware Host Client. These command line functions can be used to query the current SMC Appliance firmware version and update the SMC Appliance's firmware (an administrator must use the GUI to query and update NGFW Engine software), to manually set the SMC Appliance's time, and to configure the SMC CLI session time out. The SMC Appliance also offers a non-CLI, remote interface for management. This remote interface offers access through the GUI client using TLS v1.2, and provides all management functionality except those commands available through the CLI.

Section 6.7 (FMT_MOF.1/Functions) of the ST indicates that the TOE allows only authenticated administrators to configure TLS protected syslog export and to configure the TOE's behavior when the local audit storage space becomes full.

Section 6.7 (FMT_MOF.1/ManualUpdate) of the ST indicates that only administrators can initiate TOE updates.

Section 6.7 (FMT_MTD.1/CoreData) of the ST indicates that only the administrator can manage TSF data and configure TOE services. TSF data includes audit data, cryptographic data, authentication data, configuration data, security attributes, session timeouts and updates. The TOE requires that the administrator perform all configuration through the SMC (which then communicates with the Engines under its control)-the Engines provide no direct interface for administrators in their evaluated configuration.

Section 6.7 (FMT_MTD.1/CryptoKeys) of the ST indicates that the TOE allows only authenticated administrators to configure (import, generate, delete, change) cryptographic keys.

Section 6.7 (FMT_SMF.1) of the ST indicates that Administrators can configure operations of the TOE through the GUI, including configuring cryptographic functionality, audit behavior, authentication failure parameters, cryptographic keys, the reference identifier for peers (external syslog server and IKEv2 peers), services available prior to login and configuring NTP. The local command line interface is used by administrators to query the current SMC firmware version, install SMC updates, manually set the time, and configure the CLI session timeout. The administrator can enable the interaction between TOE components (the TOE only allows communications between the SMC and each of the Engines under its control) as part of the setup process. The administrator can disable the interaction between TOE components by removing the Engine from the SMC's control. Administrators can import X.509v3 certificates to the TOE's trust store and designate X509.v3 certificates as trusted certificate authorities.

Section 7 of the ST describes the requirement allocation and maps the SFRs to TOE components to show how the security management SFRs are shared between the TOE components.

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

For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation 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.

The management functions performed via the SMC appliance to manage both TOE components (SMC and NGFW Engine) are identified or referenced throughout this AAR with the requirement to which they apply. In each case, the Guidance has been verified and rationale provided for the required guidance assurance activities. For the management of the behavior of the audit functions, see guidance assurance activities for FAU_STG_EXT.1, FAU_STG_EXT.4 and FAU_STG_EXT.5.

Component Testing Assurance Activities: 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.

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 without prior authentication as Security Administrator. The effects of the modifications should be confirmed.

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.

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.

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.

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.

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.

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.

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.

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

Test 1: The evaluator configured a user as a non-administrator and attempted to configure external audit and behavior when the log space is full. The evaluator confirmed that these actions were not available to the user.

Test 2: The successful configuration of audit export by an authorized administrator and the behavior of the TOE when the log is full was demonstrated in NDcPP22e:FAU_STG_EXT.1.

Test 3: As demonstrated in FIA_UIA_EXT.1 there are no settings or commands available to an administrator to modify any functions at all prior to login. An attempt by an unauthorized user was demonstrated in FMT_MOF.1/Functions-t1.

Test 4: Determining the audit export configuration can be performed at the same time of configuration. The successful configuration of audit export by an authorized administrator and the behavior of the TOE when the log is full was demonstrated in NDcPP22e:FAU_STG_EXT.1.

2.7.2 MANAGEMENT OF SECURITY FUNCTIONS BEHAVIOUR (NDCPP22E:FMT_MOF.1/MANUALUPDATE)

2.7.2.1 NDCPP22E:FMT_MOF.1.1/MANUALUPDATE

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

There are no specific requirements for non-distributed TOEs.

See FMT_MOF.1/Functions above where this same activity has been performed.

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

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

The “Secure the update process” section of the **Admin Guide** provides the steps and commands used to update the SMC Appliance. It also instructs the reader to review and follow the guidance in the chapters about managing SMC Appliance patches and upgrading NGFW Engines in the *Forcepoint Next Generation Firewall Product Guide* to ensure that the update is secure.

- Chapter 87 (SMC Appliance Maintenance) - the “Patch or upgrade the SMC Appliance in the Management Client” section provides the steps for using the Management client to install SMC appliance patches. The “Patching or Upgrading the SMC Appliance” section states that it is important to upgrade the SMC Appliance before upgrading the engines. An old SMC version might not be able to recognize the new version engines and can generate an invalid configuration for them. The Management Server can control several older versions of engines. Instructions are also provided for upgrading via the command line in section “Patch or upgrade the SMC Appliance on the command line”.

- Chapter 85 (Upgrading NGFW Engines) - the “How engine upgrades work” section states that the upgrade package is imported to the Management Server manually or automatically. The “What do I need to know before I begin” section states that the SMC must be up to date before you upgrade the engines. An old SMC version might not be able to recognize the new version engines and can generate an invalid configuration for them. The “Upgrading the NGFW engines configuration overview” section indicates that the Engines can be updated using the Management Client. The “Obtain and import NGFW engine upgrade files” provides the steps for downloading the installation files. The “Upgrade engines remotely” section provides the steps for upgrading the Engine remotely from the Management Server.

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

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

Test 1 - The evaluator configured a user as a non-administrator and then attempted to import an update as the non-administrator. The evaluator confirmed that this action was not available to the non-administrative user.

Test 2 - See FPT_TUD_EXT.1 for a successful update.

2.7.3 Management of TSF Data (NDcPP22e:FMT_MTD.1/CoreData)

2.7.3.1 NDcPP22e:FMT_MTD.1.1/CoreData

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.7 of the ST states that the SMC Appliance offers two administrative interfaces - command line and GUI (the NGFW Engine provides no administrator access at all). The SMC Appliance offers command line functions which are accessible via the CLI. The CLI is a text based interface which can be accessed the virtual machine console in the VMware Host Client. These command line functions can be used to query the current SMC Appliance firmware version and update the SMC Appliance's firmware (an administrator must use the GUI to query and update NGFW Engine software), to manually set the SMC Appliance's time, and to configure the SMC CLI session time out. The SMC Appliance also offers a non-CLI, remote interface for management. This remote interface offers access through the GUI client using TLS v1.2, and provides all management functionality except those commands available through the CLI.

Section 6.7 (FMT_MOF.1/Functions) of the ST indicates that the TOE allows only authenticated administrators to configure TLS protected syslog export and to configure the TOE's behavior when the local audit storage space becomes full.

Section 6.7 (FMT_MOF.1/ManualUpdate) of the ST indicates that only administrators can initiate TOE updates.

Section 6.7 (FMT_MTD.1/CoreData) of the ST indicates that only the administrator can manage TSF data and configure TOE services. TSF data includes audit data, cryptographic data, authentication data, configuration data, security attributes, session timeouts and updates. The TOE requires that the administrator perform all configuration through the SMC (which then communicates with the Engines under its control)-the Engines provide no direct interface for administrators in their evaluated configuration.

Section 6.7 (FMT_MTD.1/CryptoKeys) of the ST indicates that the TOE allows only authenticated administrators to configure (import, generate, delete, change) cryptographic keys as well as the TOE's trust store and designate X509.v3 certificates.

Section 6.7 (FMT_SMF.1) of the ST indicates that Administrators can configure operations of the TOE through the GUI, including configuring cryptographic functionality, audit behavior, authentication failure parameters, cryptographic keys, the reference identifier for peers (external syslog server and IKe prior to login. The local command line interface is used by administrators to verify and install TOE updates, manually set the time, and configure the CLI session timeout. The administrator can enable the interaction between TOE components (the TOE only allows communications between the SMC and each of the Engines under its control) as part of the setup process. The administrator can disable the interaction between TOE components by removing the Engine from the SMC's control.

Section 6.6 of the ST states that the TOE displays a banner, and filters network traffic prior to administrative login. The TOE also requires login prior to all administrative actions.

Section 7 describes the requirement allocation and maps the SFRs to TOE components to show how the security management SFRs are shared between the TOE components.

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

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

The ST indicates that only security administrators can login to manage TSF data and configure TOE services. TSF data include audit data, cryptographic data, authentication data, configuration data, security attributes, session timeouts, and updates. The TSF data manipulating functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

The TOE defines an administrator role. User accounts that are associated with the administrator role are considered to fulfill the TOE Security Administrator role.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table states that the Management client (GUI) should be used to manage users and passwords in the SMC. The local console user accounts are synchronized with the user accounts used in the SMC. The local console accounts and passwords are managed from the SMC. Only SMC user accounts with unrestricted permissions are available on the SMC Appliance local console. This section also refers the reader to the chapter about Administrator accounts in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide

- Chapter 21 (Administrator Accounts) provides instructions for creating administrative users under the "Adding administrator accounts" subsection. This includes configuring administrator authentication to use a password stored in the internal database of the SMC and configuring an administrator with unrestricted permissions which allows logon via the local console.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the

TLS ciphersuite set. The reader is referred to the *Forcepoint Next Generation Firewall Product Guide* for instructions to enable SMC Web Access.

The "Configure settings for an evaluated configuration" section of the **Admin Guide** provides a reference to specific sections of the *Forcepoint Next Generation Firewall Product Guide* which provide detailed instructions for configuring a trusted root CA, generating a certificate request, importing a certificate and enabling TLS protection for communications between the TOE and an external syslog server. This guide is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 10 (Managing certificates for system communications) provides instructions for using certificates to secure communications with external entities, creating a TLS profile using TLSv1.2 and configuring a TLS identity. The "Create Trusted Certificate Authority elements" section describes how to create a new Trusted Certificate Authority element in order to import a certificate thereby designating it as a trust anchor.

The "Define additional VPN certificate authorities" section of the **Admin Guide** describes how an administrator can securely load CA certificates into the TOE for use during IPsec/IKEv2 authentication.

Component Testing Assurance Activities: No separate testing for FMT_MTD.1/CoreData is required unless one of the management functions has not already been exercised under any other SFR.

All of the management functions have been exercised under the other SFRs as demonstrated throughout the AAR.

2.7.4 Management of TSF Data (NDcPP22e:FMT_MTD.1/CryptoKeys)

2.7.4.1 NDcPP22e:FMT_MTD.1.1/CryptoKeys

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

See FMT_MOF.1/Functions above where this same activity has been performed.

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

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

See FMT_MOF.1/Functions above where this same activity has been performed.

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

The successful generation of a new CSR and private key, and the subsequent import of CA and signed certificate by an authorized administrator is demonstrated in NDcPP22e:FIA_X509_EXT.3.

The evaluator configured a user as non-administrator and attempted to generate a new CSR as the non-administrator user. The evaluator confirmed that the management of crypto keys and certificates was not available to the non-administrative user.

2.7.5 Management of TSF Data (VPNGW12:FMT_MTD.1/CryptoKeys)

2.7.5.1 VPNGW12:FMT_MTD.1.1/CryptoKeys

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to make it mandatory because of the TOE's required support for IPsec.

Section 6.7 of the ST states that only the authorized administrator can perform operations on cryptographic keys and certificates used for the VPN.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.6 Specification of Management Functions - per TD0631 (NDcPP22e:FMT_SMF.1)

2.7.6.1 NDcPP22e:FMT_SMF.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1(1)/ManualUpdate, FMT_MOF.1(4)/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1(2)/Services, and FMT_MOF.1(3)/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

(containing also requirements on Guidance Documentation and Tests)

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

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

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

See the other requirements in this AAR as referenced. All security management functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

Section 6.7 in the ST states that the Virtual SMC Appliance offers two administrative interfaces - command line and GUI, while the NGFW Engine provides no administrator access at all. The command line functions can be used to query the current Virtual SMC Appliance firmware version, update the Virtual SMC Appliance's firmware, manually set the Virtual SMC Appliance's time and configure the SMC CLI session time out. All security management functions are available through the GUI, except the commands available only through the CLI for manually setting the time and configuring the CLI session time out.

The TOE ensures that only security administrators can login to manage TSF data and configure TOE services. The TOE requires that the administrator perform all configuration through the SMC (which then communicates with the Engines under its control)-the Engines provide no direct interface for administrators in the evaluated configuration. The administrator can enable the interaction between TOE components (the TOE only allows communications between the SMC and each of the Engines under its control) as part of the setup process. The administrator can disable the interaction between TOE components by removing the Engine from the SMC's control.

The "FIPS mode restrictions" section in the **Admin Guide** further confirms that when FIPS mode is enabled, the NGFW Engine local console, command line interface, and SSH access are not available.

The "Enable FIPS mode on the SMC Appliance" section in the **Admin Guide** describes the steps to prepare the appliance for installation and reference the *Forcepoint Next Generation Firewall Installation Guide* which provides instructions for installing the SMC as a virtual appliance in ESXi 7.0. The administrator is instructed to access the appliance using the virtual machine console in the VMware Host Client.

In the "Configure settings for an evaluated configuration" section in the **Admin Guide**, the Administrative Logins table entry provides guidelines for managing users and passwords and indicates that the local console accounts and passwords are managed from the SMC Client GUI. Only SMC user accounts with unrestricted permissions are available on the SMC Appliance local console. This section also provides the commands that are available via the CLI for manually setting the time, setting the local session timeout and logging out of the console.

See FCO_CPC_EXT.1 where the TSS and Guidance assurance activities describe the ability to configure the interaction between TOE components. The evaluator confirmed that the TOE behavior during testing of the registration process matched what is described in the ST and the Guidance.

Component Guidance Assurance Activities: See TSS Assurance Activities

See TSS Assurance Activities.

Component Testing Assurance Activities: The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

All of the management functions were demonstrated throughout the course of testing.

2.7.7 Specification of Management Functions (STFFW14E:FMT_SMF.1/FFW)

2.7.7.1 STFFW14E:FMT_SMF.1.1/FFW

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

This activity has been performed in STFFW14E:FFW_RUL_EXT.1 and NDcPP2e:FMT_SMF.1 where the evaluator verified that the guidance describes and provides instructions for configuring all management functions specified in FMT_SMF.1/FFW.

Component Guidance Assurance Activities: See TSS Activity.

See TSS Assurance Activities.

Component Testing Assurance Activities: The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

All of the management functions were demonstrated throughout the course of testing.

2.7.8 Specification of Management Functions (VPNGW12:FMT_SMF.1/VPN)

2.7.8.1 VPNGW12:FMT_SMF.1.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

See the NDcPP22e:FMT_SMF.1 TSS Assurance Activity.

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

See the NDcPP22e:FMT_SMF.1 TSS Assurance Activity.

Component Testing Assurance Activities: The evaluator tests management functions as part of performing other test EAs. No separate testing for FMT_SMF.1/VPN is required unless one of the management functions in FMT_SMF.1.1/VPN has not already been exercised under any other SFR.

All of the management functions were demonstrated throughout the course of testing. Refer to FAU_GEN.1 where audit records were collected for each of the security management functions selected.

2.7.9 Restrictions on Security Roles (NDcPP22e:FMT_SMR.2)

2.7.9.1 NDcPP22e:FMT_SMR.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.9.2 NDcPP22e:FMT_SMR.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.9.3 NDcPP22e:FMT_SMR.2.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.7 of the ST states that the TOE provides an administrator role. User accounts that are associated with the administrator role are considered Security Administrators. Users in this role can perform security management functions locally and remotely. Security Administrators can manage and configure all TSF data including audit data, cryptographic data, authentication data, configuration data, user and administrator security attributes, session timeouts, and updates. The TOE ensures that only security administrators can login to manage TSF data and configure TOE services. The TOE requires that the administrator perform all configuration through the SMC (which then communicates with the Engines under its control)-the Engines provide no direct interface for administrators in their evaluated configuration.

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

The **Admin Guide** contains instructions for managing all required aspects of the TOE both locally and remotely as documented throughout this AAR. The Client GUI is the primary administrative interface, while the CLI provides a very limited set of administrative functions. Instructions are provided specifically to manage access to these

available management interfaces. The Client GUI can be accessed from a web browser or can be installed and accessed locally from an administrator workstation. All remote administration is protected using TLS. The CLI can be accessed from the virtual machine console in the VMware Host client.

The "FIPS Mode Restrictions" and "Enable FIPS mode on the SMC Appliance" sections of the **Admin Guide** provide instructions for configuring the Virtual SMC Appliance, NGFW Engine and Management Client into FIPS mode. Once FIPS mode is enabled, the list of algorithms, ciphersuites and protocols available for use is restricted to those specified in the Security Target. For remote administration, there are only two TLS ciphersuites available to the client once FIPS mode is enabled and no further configuration is needed. In FIPS mode, the NGFW Engine local console, command line interface and SSH access are not available.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table states that the Management client (GUI) should be used to manage users and passwords in the SMC. The local console user accounts are synchronized with the user accounts used in the SMC. The local console accounts and passwords are managed from the SMC. Only SMC user accounts with unrestricted permissions are available on the SMC appliance local console. This section also refers the reader to the chapter about Administrator accounts in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide

- Chapter 21 (Administrator Accounts) provides instructions for creating administrative users under the "Add administrator accounts" subsection. This includes configuring administrator authentication to use a password stored in the internal database of the SMC and configuring an administrator with unrestricted permissions which allows logon via the local console. To log on to the SMC Appliance command line, administrators must have superuser administrator permissions and can execute root-level commands using the sudo tool.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the TLS ciphersuite set. The reader is referred to the *Forcepoint Next Generation Firewall Product Guide* for instructions to enable SMC Web Access.

Forcepoint Next Generation Firewall Product Guide

- Chapter 25 (Using the Management Client in a web browser) provides the detailed steps for enabling SMC Web Access on the Management Server in order to run the Management Client in a web browser.

The "Install the Management Client" section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the "Supporting documentation" section.

The “Install the Management Client” section of the **Admin Guide** also references the *How to install Forcepoint NGFW in FIPS mode* guide which is accessed via the link found in the “Supporting documentation” section.

How to install Forcepoint NGFW in FIPS mode:

- The “Install the Management Client using a file” subsection describes how to access, download and install the management client on a PC in the operating environment and then ensure that it is operating in FIPS mode by selecting the ‘Restricted Cryptographic Algorithms Compatible with FIPS 140-2’ operating mode.

In the “Configure settings for an evaluated configuration” section in the **Admin Guide,** the Password Guidelines entry in the table refers the reader to further information on the topic about enabling and defining password policy settings in the Forcepoint *Next Generation Firewall Product Guide* which is accessed via the link found in the “Supporting documentation” section.

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

All administrative interfaces were used throughout the course of testing including through the virtual machine console, and through the Management Client GUI interface using TLS.

2.8 Packet Filtering (FPF)

2.8.1 Packet Filtering Rules - per TD0683 (VPNGW12:FPF_RUL_EXT.1)

2.8.1.1 VPNGW12:FPF_RUL_EXT.1.1

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

The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Section 6.5 states that during the TOE’s boot process, there is a lag between the time when the network interface

is operational, and the time that the Stateful Traffic Filtering functionality is fully functioning. During this time, Boot Security is enforced:

- Traffic flow through the appliance is disabled; and
- Traffic to and from the appliance is controlled by a Default Filter that drops all external traffic to the appliance

Section 6.5 states that if any interface is overwhelmed with traffic, the TOE will drop the packets.

Guidance Assurance Activities: The operational guidance associated with this requirement is assessed in the subsequent test EAs.

See subsequent test assurance activities.

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

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

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

Test 1 and Test 2: Because the TOE acts as a router, packets are sent to a TOE interface and forwarded. Thus, traffic used during STFFW14e:FFW_RUL_EXT.1, test 1 testing is directed at TOE interfaces to be forwarded as appropriate to the ultimate destination. Therefore, the tests for Packet Filtering from the VPN Gateway EP are a subset of those identified by the Firewall PP and are covered by test described under STFFW14e:FFW_RUL_EXT.1. Refer to STFFW14e:FFW_RUL_EXT.1 for results.

2.8.1.2 VPNGW12:FPF_RUL_EXT.1.2

TSS Assurance Activities: There are no EAs specified for this element. Definition of packet piltering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

There are no Evaluation Activities specified for this element.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.8.1.3 VPNGW12:FPF_RUL_EXT.1.3

TSS Assurance Activities: There are no EAs specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

There are no Evaluation Activities specified for this element.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.8.1.4 VPNGW12:FPF_RUL_EXT.1.4

TSS Assurance Activities: The evaluator shall verify that the TSS describes a packet filtering policy that can use the following fields for each identified protocol, and that the RFCs identified for each protocol are supported:

- IPv4 (RFC 791)

o source address

o destination address

o Protocol

- IPv6 (RFC 8200)

o source address

o destination address

o next header (protocol)

- TCP (RFC 793)

o source port

o destination port

- UDP (RFC 768)

o source port

o destination port

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

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

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

Section 6.5 of the ST states that Firewall rules can be set to filter on protocol, source address, destination address, source port, destination port, ICMP type or ICMP code. All protocols including icmpv4, icmpv6, ipv4, ipv6, tcp, and udp may be used in firewall rules. The firewall rules implement the SPD rules (Allow, Discard, Refuse). Each rule can be configured to log status of packets pertaining to the rule. Rules can also be assigned to each network interface.

Guidance Assurance Activities: The evaluators shall verify that the operational guidance identifies the following protocols as being supported and the following attributes as being configurable within packet filtering rules for the associated protocols:

- IPv4 (RFC 791)

o source address

o destination address

o Protocol

- IPv6 (RFC 8200)

o source address

o destination address

o next header (protocol)

- TCP (RFC 793)

o source port

o destination port

- UDP (RFC 768)

o source port

o destination port

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

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

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

The "How firewalls process traffic" section of the **Admin Guide** states that Firewall rules can be associated with one or more network interfaces of the TOE. They can also incorporate the following attributes to specify traffic that can match a given rule:

ICMPv4

- Type
- Code

ICMPv6

- Type
- Code

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

The administrator is able to define packet filtering rules within the Firewall Policy. The administrator can specify which services to allow (in this case, FTP) as well as a port range to block certain ports from being used. The policy can also be logged, and the log can be viewed under the Logs.

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

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

- IPv4

o Source address

o Destination Address

o Protocol

- IPv6

o Source Address

o Destination Address

o Next Header (Protocol)

- TCP

o Source Port

o Destination Port

- UDP

o Source Port

o Destination Port

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

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

The testing associated with this requirement are addressed in the subsequent testing of VPNGW11:FPF_RUL_EXT.1.6 where the firewall rules used during that testing were configured manually by the evaluator.

2.8.1.5 VPNGW12:FPF_RUL_EXT.1.5

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

Section 6.5 of the ST states the TOE provides an information flow control mechanism using a rule base that comprises a set of security policy rules, i.e., the firewall security policy. The NGFW Engine enforces the firewall security policy on all traffic that passes through the engine, via its internal or external network Ethernet interfaces. The traffic is TCP, UDP, ICMPv4, ICMPv6, connections over IPv4 and IPv6. The NGFW Engine inspects and filters these protocols based upon their header fields as defined by their corresponding RFC (See Table 13).

The NGFW Engine only permits traffic to pass through that has been explicitly allowed by the firewall security policy, and implements packet defragmentation to enforce the policy on entire IP packets. Administrators using the Management Server define the firewall security policy rules.

Any network traffic passed by the NGFW Engine must be explicitly allowed by a firewall rule or be part of an established session allowed by a rule, or it is dropped. This is true even in the case of attempts to flood a TOE interface (in which case some packets may be dropped, but no packets are ever passed that would violate policy).

All received network packets are processed by the NGFW Engine software module before transmission. The NGFW software module does stateful filtering of the received network packets according to the configured traffic filtering rules. Protocol Agents are used for advanced processing of traffic that require special handling such as permitting an FTP data connection dynamically. The NGFW Engine software denies the traffic if the Protocol Agent cannot process the traffic. Incoming packets are dropped if a network packet cannot be processed due to insufficient memory. All incoming network packets are also discarded before the NGFW Engine software module has been loaded, and the NGFW Engine software module denies all traffic until the module has been configured. Network interfaces and routing are configured after the NGFW Engine software module has been loaded. If the configured firewall rules cannot be applied during startup, only the management network interface will be available and traffic through the firewall will be denied.

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

The "Stateful Packet Filtering" section of the **Admin Guide** states that the Firewall tab of SmartConsole displays the firewall rules in the order that the Gateways apply them during packet processing. An administrator can reorder the rules with SmartConsole by simply dragging them up or down within the Security Policies tab.

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

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

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

The VPNGW11:FPF_RUL_EXT.1 testing is a subset of testing required for the STFFW14e:FFW_RUL_EXT.1 testing. Refer to the STFFW14e:FFW_RUL_EXT.1.9 tests.

2.8.1.6 VPNGW12:FPF_RUL_EXT.1.6

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

Section 6.5 of the ST states any network traffic passed by the NGFW Engine must be explicitly allowed by a firewall rule or be part of an established session allowed by a rule, or it is dropped.

Section 6.5 of the ST also states the TOE supports all IPv4 and IPv6 protocols except IPv6 protocol 1, which the TOE blocks by default as IP protocol 1 is ICMP for IPv4 (while protocol 58 is IPv6 ICMP) and is not a valid protocol for IPv6.

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

The "Stateful Packet Filtering" section of the **Admin Guide** states that the Check Point Security Gateways possess a default firewall policy (a policy is a set of rules, and the default set is named "Standard") that includes an "allow all" rule at the end, and thus if the administrator intends to utilize this rule, they must reconfigure the existing

allow all rule to drop all packets (or create a new drop all rule) and place this rule at the end of the policy list. The administrator must also enable logging for this policy in order to capture logs for the default drop rule. Note that all newly defined firewall policies come with a final (default) drop all rule.

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

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

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

Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

Test 4: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination

addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

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

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

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

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

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

The following table identifies the RFC defined values for the protocol fields for IPv4 and IPv6 to be used in configuring and otherwise testing Packet Filtering rule definition and enforcement:

Test 1: The evaluator performed the following steps and confirmed that packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv4 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of PERMIT & LOG rules:
    a. a specific source address and specific destination address,
    b. wildcard source address and specific destination address,
    c. specific source address and wildcard destination address, and
    d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is permitted (determine traffic is permitted by capturing packets transmitted by the TOE):
    a. The specific permitted source address (1a) and specific permitted destination address (1a).
    b. A specific address within the wildcard source address from (1b) and the specific destination address from (1b).
    c. The specific permitted source address in (1c) and a specific address within the wildcard destination address from (1c).
    d. A specific address within the wildcard source address from (1d) and A specific address within the wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet permitted by the rule results in an accurate LOG record.

Test 2: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv4 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of DENY and LOG rules:
    a. a specific source address and specific destination address,
    b. wildcard source address and specific destination address,
    c. specific source address and wildcard destination address,
    d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is denied and logged (Determine traffic is denied by capturing packets transmitted by the TOE):
    a. The specific source address (1a) and specific destination address (1a).
    b. A specific address within the wildcard source address from (1b) and the specific destination address from (1b).
    c. The specific source address in (1c) and a specific address within the wildcard destination address from (1c).
    d. A specific address within the wildcard source address from (1d) and A specific address within the wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet Denied by the rule results in an accurate LOG record.

Test 3: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv4 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of PERMIT & LOG rules and DENY & LOG rules:
    a. a specific source address and specific destination address,
    b. wildcard source address and specific destination address,

c. specific source address and wildcard destination address,

d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is denied(determine traffic is denied by capturing packets transmitted by the TOE):

   a. An address other than the specific permitted/denied source address (1a) and an address other than the specific permitted/denied destination address (1a).

   b. A specific address that is not within the wildcard source address from (1b) and an address other than the specific permitted/denied destination address from (1b)

   c. An address other than the specific permitted/denied source address in (1c) and an address that is not within the wildcard destination address from (1c).

   d. A specific address that is not within the wildcard source address from (1d) and A specific address that is not within the wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet denied by the default cleanup rule results in an accurate LOG record.

Test 4: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv6 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of PERMIT & LOG rules:

   a. a specific source address and specific destination address,

   b. wildcard source address and specific destination address,

   c. specific source address and wildcard destination address,

   d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is permitted (determine traffic is permitted by capturing packets transmitted by the TOE):

a. The specific permitted source address (1a) and specific permitted destination address (1a).

b. A specific address within the wildcard source address from (1b) and the specific destination address from (1b),

c. The specific permitted source address in (1c) and a specific address within the wildcard destination address from (1c).

d. A specific address within the wildcard source address from (1d) and A specific address within the wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet permitted by the rule results in an accurate LOG record.

Test 5: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv6 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of DENY and LOG rules:

   a. a specific source address and specific destination address,

   b. wildcard source address and specific destination address,

   c. specific source address and wildcard destination address,

   d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is denied and logged (Determine traffic is denied by capturing packets transmitted by the TOE):

   a. The specific source address (1a) and specific destination address (1a).

   b. A specific address within the wildcard source address from (1b) and the specific destination address from (1b),

   c. The specific source address in (1c) and a specific address within the wildcard destination address from (1c).

   d. A specific address within the wildcard source address from (1d) and A specific address within the

wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet Denied by the rule results in an accurate LOG record.

Test 6: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset for the IPv6 transport protocols from table 3 of the VPNGW11 supporting document. For each protocol specify the following types of PERMIT & LOG rules and DENY & LOG rules:
    a. a specific source address and specific destination address,
    b. wildcard source address and specific destination address,
    c. specific source address and wildcard destination address,
    d. wildcard source address and wildcard destination address

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is permitted (determine traffic is permitted by capturing packets transmitted by the TOE):
    a. An address other than the specific permitted/denied source address (1a) and an address other than the specific permitted/denied destination address (1a).
    b. A specific address that is not within the wildcard source address from (1b) and an address other than the specific permitted/denied destination address from (1b)
    c. An address other than the specific permitted/denied source address in (1c) and an address that is not within the wildcard destination address from (1c).
    d. A specific address that is not within the wildcard source address from (1d) and A specific address that is not within the wildcard destination address from (1d).

Note: the addresses used in 2a through 2d correspond with the addresses configured in the rules specified by 1a through 1d.

3. Ensure every packet denied by the default cleanup rule results in an accurate LOG record.

Test 7: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset using TCP to specify the following types of PERMIT & LOG rules:
    a. specific source port and destination port range,
    b. source port range and specific destination port,
    c. specific source port and specific destination port

    Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is permitted (determine traffic is permitted by capturing packets transmitted by the TOE):
    a. The specific permitted source port in (1a) and a specific port within the destination port range (1a).
    b. A specific port within the source port range (1b) and the specific destination port (1b).
    c. The specific permitted source port (1c) and specific permitted destination port (1c).

    Note: the addresses used in 2a through 2c correspond with the addresses configured in the rules specified by 1a through 1c.

3. Ensure every packet permitted by the rule results in an accurate LOG record.

Test 8: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset using TCP to specify the following types of DENY & LOG rules:
    a. specific source port and destination port range,
    b. source port range and specific destination port,
    c. specific source port and specific destination port

    Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is denied (determine traffic is denied by capturing no packets transmitted by the TOE):

a. The specific permitted source port in (1a) and a specific port within the destination port range (1a).

b. A specific port within the source port range (1b) and the specific destination port (1b).

c. The specific permitted source port (1c) and specific permitted destination port (1c).

Note: the addresses used in 2a through 2c correspond with the addresses configured in the rules specified by 1a through 1c.

3. Ensure every packet denied by the rule results in an accurate LOG record.

Test 9: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset using UDP to specify the following types of PERMIT & LOG rules:

   a. specific source port and destination port range,

   b. source port range and specific destination port,

   c. specific source port and specific destination port

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is permitted (determine traffic is permitted by capturing packets transmitted by the TOE):

   a. The specific permitted source port in (1a) and a specific port within the destination port range (1a).

   b. A specific port within the source port range (1b) and the specific destination port (1b).

   c. The specific permitted source port (1c) and specific permitted destination port (1c).

Note: the addresses used in 2a through 2c correspond with the addresses configured in the rules specified by 1a through 1c.

3. Ensure every packet permitted by the rule results in an accurate LOG record.

Test 10: The evaluator performed the following steps and confirmed packets are filtered properly and the rules are enforced.

1. Configure a ruleset using UDP to specify the following types of DENY & LOG rules:
    a. specific source port and destination port range,
    b. source port range and specific destination port,
    c. specific source port and specific destination port

Note: the addresses used in a through d must either be tested sequentially, or must be non-overlapping.

2. Generate traffic and pass it through the TOE ensuring that the following traffic is denied (determine traffic is denied by capturing no packets transmitted by the TOE):
    a. The specific permitted source port in (1a) and a specific port within the destination port range (1a).
    b. A specific port within the source port range (1b) and the specific destination port (1b).
    c. The specific permitted source port (1c) and specific permitted destination port (1c).

Note: the addresses used in 2a through 2c correspond with the addresses configured in the rules specified by 1a through 1c.

3. Ensure every packet denied by the rule results in an accurate LOG record.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9 Protection of the TSF (FPT)

2.9.1 Protection of Administrator Passwords (NDcPP22e:FPT_APW_EXT.1)

2.9.1.1 NDcPP22e:FPT_APW_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.1.2 NDcPP22e:FPT_APW_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.9 of the ST states that the Management Server stores passwords with other configuration data in a database and synchronizes this database with the Linux password database (i.e., /etc/shadow....). Synchronization takes the form of the contents of the database overwriting the contents of the Linux password database. There is no administrative interface to view or manipulate the raw configuration database. The only interface to the database is through administrative actions which modify the contents of the database in a controlled manner. Passwords are salted and hashed using SHA-512 when stored.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.2 Failure with Preservation of Secure State (Self-Test Failures) (VPNGW12:FPT_FLS.1/SelfTest)

2.9.2.1 VPNGW12:FPT_FLS.1.1/SelfTest

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall ensure the TSS describes how the TOE ensures a shutdown upon a self-test failure, a failed integrity check of the TSF executable image, or a failed health test of the noise source. If there are instances when a shut-down does not occur, (e.g., a failure is deemed non- security

relevant), the evaluator shall ensure that those cases are identified and a rationale is provided that supports the classification and justifies why the TOE's ability to enforce its security policies is not affected in any such instance.

Section 6.9 of the ST states that the Virtual SMC Appliance will halt its boot as a result of a KAT error or any error in its integrity verification (the SMC verifies the ECDSA P-521 with SHA2-384 signature on a catalog file of SHA-256 digests of the SMC binaries) upon system startup. The NGFW Engine (using its Forcepoint NGFW FIPS Library 1.1.1 based upon OpenSSL 1.1.1 (which utilizes the Forcepoint NGFW FIPS Cryptographic Module 1.2.1) uses a preinstalled public key to verify the ECDSA P-521 with SHA-512 signature of the whole partition containing TOE binaries and if it finds an error, it will reboot.

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

The "Verify the SMC Appliance self-tests" section in the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode

- Section "Check the SMC Appliance self-tests" provides instructions for checking the self-test results in the console. This includes a description of all the possible errors that may result from the self-tests as well as the actions that an administrator should take in response.

The "Verify the NGFW Engine self-tests section in the **Admin Guide** also references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode

- Section "Check the NGFW Engine self-tests" provides instructions for checking the self-test results in the console. This includes a description of all the possible error messages that may be shown via the console as well as the actions that an administrator should take in response.

These descriptions of possible self-test errors correspond to those described in the TSS.

Component Testing Assurance Activities: None Defined

2.9.3 Basic internal TSF data transfer protection - per TD0639 (NDcPP22e:FPT_ITT.1)

2.9.3.1 NDcPP22e:FPT_ITT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the TOE is not a distributed TOE, then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall examine the TSS to determine that, for all communications between components of a distributed TOE, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator shall also confirm that all protocols listed in the TSS for these inter-component communications are specified and included in the requirements in the ST.

Section 6.9 of the ST states that the TOE utilizes TLS protected communications from Engines to their SMC (log forwarding) and from the SMC to its Engines (management). The TOE has no other communications between components. This is consistent with the FPT_ITT.1 requirement.

Component Guidance Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall confirm that the guidance documentation contains instructions for establishing the relevant allowed communication channels and protocols between each pair of authorized TOE components, and that it contains recovery instructions should a connection be unintentionally broken.

See FCO_CPC_EXT.1 where this activity has already been performed.

Component Testing Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall ensure that communications using each protocol between each pair of authorized TOE components is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

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

c) Test3: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route between distributed components.

The evaluator shall ensure that, for each different pair of non-equivalent component types, the connection is physically interrupted for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration that is shorter than the application layer timeout but is of sufficient length to interrupt the network link layer.

The evaluator shall ensure that when physical connectivity is restored, either communications are appropriately protected, or the secure channel is terminated and the registration process (as described in the FTP_TRP.1/Join) re-initiated, with the TOE generating adequate warnings to alert the Security Administrator.

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

Further assurance activities are associated with the specific protocols.

Test 1: The successful joining of the distributed TOE devices was demonstrated in NDcPP20E:FCO_CPC_EXT.1. The evaluator further tested the ongoing TLS protocol between the components during tests for FCS_TLSC_EXT.1/2 and FCS_TLSS_EXT.1/2 where the SMC and Engine were tested as both client and server.

Test 2: This test was performed in NDcPP22e_FPT_ITT.1_t3 where packet captures confirmed that the channel data is TLS encrypted and not sent in plaintext.

Test 3: The evaluator physically disrupted the connection between the Engine and the SMC first for about 25 seconds and a second time for around 20 minutes. In both cases, the evaluator analyzed the traffic and found no instance of sensitive data being sent outside the protected TLS channel.

2.9.4 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys) (NDcPP22e:FPT_SKP_EXT.1)

2.9.4.1 NDcPP22e:FPT_SKP_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that it details how any pre-shared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through an

interface designed specifically for that purpose, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Section 6.9 of the ST states that none of the TOE components utilize pre-shared keys or long-lived symmetric keys. The only keys retained by the components of the TOE are associated with certificates used for TLS. These keys are stored in a password protected Java keystore (on the Virtual SMC Appliance) and on a Read/Write partition on the NGFW Engine. Since the NGFW Engine does not support an interface for local administration, this data is not accessible once stored in the partition.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.5 Reliable Time Stamps - per TD0632 (NDcPP22e:FPT_STM_EXT.1)

2.9.5.1 NDcPP22e:FPT_STM_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.5.2 NDcPP22e:FPT_STM_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.9 of the ST indicates that the NGFW engine and the Virtual SMC appliance both include their own real-time hardware-based clock. This clock is used for timestamps used in audit data, verifying certificate and

certificate revocation validity, and measuring session timeouts. The TOE time can be set by administrator action through console administration of the SMC Management Server or by enabling NTP time synchronization via the SMC GUI. The Management server is responsible for accepting and propagating clock updates initiated by an administrator. Time on the NGFW Engine is updated by the SMC Management Server or alternatively from an administrator configured NTP server.

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

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide** under the Time Management table entry, the command for setting the date and time manually on the SMC Appliance is provided. This section also provides instructions for enabling and configuring NTP time synchronization and establishing the communication path between the TOE and the NTP server using a SHA-1 authentication key.

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

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

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

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

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

Test 1: The evaluator set the clock from the local console and observed the time change via the local console interface.

Test 2: This test has been performed as part of the tests in NDcPP22e: FCS_NTP_EXT.1, test 2.

2.9.6 TSF TESTING (NDCPP22E:FPT_TST_EXT.1)

2.9.6.1 NDCPP22E:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.9 of the ST states that each component of the TOE includes a set of hardware validation tests which include Known Answer Tests (KAT) for the cryptographic features provided by the Forcepoint NGFW FIPS Library 1.1.1 based upon OpenSSL 1.1.1 (which utilizes the Forcepoint NGFW FIPS Cryptographic Module 1.2.1), Virtual SMC Appliance SMC FIPS Library 1.1.1 based on OpenSSL, Virtual SMC Appliance SMC FIPS Java API 1.0.2.3, and SMC FIPS Cryptographic Module for NTP 3.79 cryptographic libraries. These KAT tests cover operation of AES, RSA, ECDSA, DRBG, SHA and HMAC-SHA. For each KAT test, the TOE uses known data as inputs into each cryptographic function, computes a cryptographic result (e.g., the AES ciphertext or SHA-512 hash), and compares the calculated result to the expected/known value. If the two do not match, the NGFW Engine will reboot as a result of the error, while the Virtual SMC Appliance will halt its boot as a result of a KAT error or any error in its integrity verification (the SMC verifies the ECDSA P-521 with SHA2-384 signature on a catalog file of SHA-256 digests of the SMC binaries) upon system startup. The NGFW Engine (using its Forcepoint NGFW FIPS Library 1.1.1 based upon OpenSSL 1.1.1 (which utilizes the Forcepoint NGFW FIPS Cryptographic Module 1.2.1) uses a preinstalled public key to verify the ECDSA P-521 with SHA-512 signature of the whole partition containing TOE binaries and if it finds an error, it will reboot. These integrity verification keys are included in the Forcepoint software and the TOE provides administrators no method to access them. These tests are sufficient because any TOE modifications or failed cryptographic operations will be immediately noted upon boot.

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

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

The "Verify the SMC Appliance self-tests" section in the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode

- Section "Check the SMC Appliance self-tests" provides instructions for checking the self-test results on the console. This includes a description of all the possible errors that may result from the self-tests as well as the actions that an administrator should take in response. There are cases where the SMC does not restart but does not permit remote connections which is the purpose of the component. The administrator is instructed to factory reset the device in this case.

The "Verify the NGFW Engine self-tests section in the **Admin Guide** also references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section.

How to install Forcepoint NGFW in FIPS mode

- Section "Check the NGFW Engine self-tests" provides instructions for checking the self-test results on the console. This includes a description of all the possible error messages that may be shown via the console as well as the actions that an administrator should take in response.

These descriptions of possible self-test errors correspond to those described in the TSS.

Component Testing Assurance Activities: It is expected that at least the following tests are performed:

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

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

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

a) FIPS 140-2, chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software. Note that the testing is not restricted to the cryptographic functions of the TOE.

b) FIPS 140-2, chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

The evaluator shall either verify that the self tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

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

The evaluator initiated a reboot from each device and confirmed that the status messages show that the FIPS self-tests were executed successfully.

2.9.7 TSF Testing (VPNGW12:FPT_TST_EXT.1)

2.9.7.1 VPNGW12:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module requires a particular self-test to be performed, but this self-test is still evaluated using the same methods specified in the Supporting Document.

Refer to NDcPP22e:FPT_TST_EXT.1

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.8 Self-Test with Defined Methods (VPNGW12:FPT_TST_EXT.3)

2.9.8.1 VPNGW12:FPT_TST_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.8.2 VPNGW12:FPT_TST_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.9 of the ST states that the Virtual SMC Appliance will halt its boot as a result of a KAT error or any error in its integrity verification (the SMC verifies the ECDSA P-521 with SHA2-384 signature on a catalog file of SHA-256 digests of the SMC binaries) upon system startup. The NGFW Engine (using its Forcepoint NGFW FIPS Library 1.1.1 based upon OpenSSL 1.1.1 (which utilizes the Forcepoint NGFW FIPS Cryptographic Module 1.2.1) uses a preinstalled public key to verify the ECDSA P-521 with SHA-512 signature of the whole partition containing TOE binaries and if it finds an error, it will reboot.

The TOE components verify noise source operation, cryptographic algorithms, and checksums of TOE binaries upon startup as described above.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.9 Trusted update (NDcPP22e:FPT_TUD_EXT.1)

2.9.9.1 NDcPP22e:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.9.2 NDcPP22e:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.9.3 NDcPP22e:FPT_TUD_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

If the options 'support automatic checking for updates' or 'support automatic updates' are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

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

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

Section 6.9 of the ST states that the TOE performs trusted updates for both of its components: the Virtual SMC management appliance and NGFW engines. The administrator can query the current software versions for the SMC software on the SMC component and for the NGFW Engine software on the SMC Management Server. To update the TOE software of the NGFW engine, an administrator can obtain an update from Forcepoint and then

upload the update to the SMC. After the SMC has the update, the SMC will verify the Forcepoint ECDSA P-521 with SHA-512 signature on the update package and only if the signature verifies correctly, the SMC will import that package, making it available to update administrator-specified NGFW engines with the new software. Once the administrator selects to upgrade a specific NGFW Engine with a patch, the SMC will transfer that update to the NGFW Engine and the Engine will also verify the signature on the update (even though the SMC has already verified the update). The Engine will use that update package (which is a full filesystem image) to write to an internal, alternate software/system partition, and then, after verifying the checksum of the newly written system partition to check for write corruptions, the Engine will reboot into that new partition.

To update the SMC itself, the administrator obtains an SMC patch from Forcepoint. The evaluator can make the patch available to the SMC two different ways, either by saving the patch to an administrator provided USB thumb-drive which is then mounted to the SMC or by uploading it to the SMC through the GUI. Then using the local console Command Line Interface (CLI), the administrator executes the ambr_load function to verify a Forcepoint ECDSA P-521 with SHA-512 signature on the patch file. If the signature verifies, the administrator can issue the ambr_install command to install the patch, and then follow the installation process (which can require a reboot for upgrades or major new features).

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

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

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

For distributed TOEs the evaluator shall verify that the guidance documentation describes how the versions of individual TOE components are determined for FPT_TUD_EXT.1, how all TOE components are updated, and the error conditions that may arise from checking or applying the update (e.g. failure of signature verification, or exceeding available storage space) along with appropriate recovery actions. The guidance documentation only has to describe the procedures relevant for the Security Administrator; it does not need to give information about the internal communication that takes place when applying updates.

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

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

The "Verify the installed version of the SMC, NGFW Engine, and SMC Appliance" section of the **Admin Guide** provides instructions for verifying the installed version of the NGFW Engine and the SMC Appliance.

The "Secure the update process" section of the **Admin Guide** states that SMC Appliance and NGFW Engine updates are verified using ECDSA P-521 with SHA-512 digital signatures and a pre-installed public key. The commands used to update the SMC Appliance verify the digital signature and reject any update that is not valid. For NGFW updates, the SMC verifies the NGFW Engine update signature when the update is imported to the SMC. Only valid updates can be imported and installed on the NGFW Engine.

The "Secure the update process" section of the **Admin Guide** provides the steps and commands used to update the SMC Appliance. It also instructs the reader to review and follow the guidance in the chapters about managing SMC Appliance patches and upgrading NGFW Engines in the *Forcepoint Next Generation Firewall Product Guide* to ensure that the update is secure.

The "Secure the update process" section of the **Admin Guide** provides a set of steps to perform if there is a failure. This discussion addresses signature failure as well as storage space failure This is discussed for all components.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 87 (SMC Appliance maintenance) - the "Patch or upgrade the SMC Appliance in the Management Client" section provides the steps for using the Management client to install SMC appliance patches. The "Patching or Upgrading the SMC Appliance" section states that it is important to upgrade the SMC Appliance before upgrading the engines. An old SMC version might not be able to recognize the new version engines and can generate an invalid configuration for them. The Management Server can control several older versions of engines. The "Patch or upgrade the SMC Appliance on the command line" section provides the steps for upgrading or patching the SMC appliance on the command line.

- Chapter 85 (Upgrading NGFW Engines) - the "How engine upgrades work" section states that the upgrade package is imported to the Management Server manually. Before the upgrade is installed on the engines, the Management Server verifies the digital signature of the upgrade package. Also, the engines verify the digital signature of the upgrade package before the upgrade is installed. The engines have two alternative partitions for the software. The "What do I need to know before I begin" section states that the SMC must be up to date before you upgrade the engines. An old SMC version might not be able to recognize the new version engines and can generate an invalid configuration for them. The "Upgrading the NGFW engines configuration overview" section indicates that the Engines can be updated using the Management Client. The "Obtain and import NGFW engine upgrade files" provides the steps for downloading the installation files. The "Upgrade NGFW Engines remotely" section provides the steps from upgrading the Engine remotely from the Management Server.

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

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

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below, and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

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

2) An image that has not been signed

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

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

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

verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

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

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

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

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

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

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

Test 1 - The evaluator displayed the version of the TOE and then installed an update. The signature verified and the update was successfully installed. The evaluator displayed the version again and confirmed that the new version

was displayed. This test was performed for both the SMC Appliance and NGFW Engine software updates.

Test 2 (1, 2, 3, 4) - The attempted to load and install the corrupted and unsigned updates one by one. This included an image modified by a hex editor, an unsigned image and an image with an invalid signature. All three images failed to load. The evaluator further confirmed this for the SMC update attempt by observing that the availability of the update in the Web GUI did not change and there were output error messages sent to the GUI. The Engine is updated via the SMC through the Client GUI. The SMC rejects the updates at the time of loading; thus they are never installed onto the Engine. This test was performed for both the SMC Appliance and NGFW Engine software updates.

Test 3: The TOE does not verify the integrity of updates using published hash.

2.9.10 Trusted Update (VPNGW12:FPT_TUD_EXT.1)

2.9.10.1 VPNGW12:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.10.2 VPNGW12:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.10.3 VPNGW12:FPT_TUD_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There is no change to the Evaluation Activities specified for this SFR in the NDcPP Supporting Document. The PP-Module modifies this SFR to mandate that a particular selection be chosen, but this selection is part of the original definition of the SFR so no new behavior is defined by the PP-Module.

Refer to NDcPP22e:FPT_TUD_EXT.1

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.10 TOE ACCESS (FTA)

2.10.1 TSF-INITIATED TERMINATION (NDCPP22E:FTA_SSL.3)

2.10.1.1 NDCPP22E:FTA_SSL.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.10 in the ST states that the SMC Management Server supports timeouts caused by inactivity through the GUI, as well as voluntary termination of a session (i.e. logout). The TOE will terminate remote interactive sessions that have been inactive for the defined interval. The administrator can configure the duration of the inactivity timeout mechanism.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table references the *Forcepoint Next Generation Firewall Product Guide* where the instructions for setting timeouts via the GUI can be found. This document is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) provides instructions for enabling and defining the password policy which includes defining session limits and idle timeouts.

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

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

Test 1 - The evaluator configured the SMC Appliance to lock the remote administrator Management Client session out after an idle period of first, 2 minutes, and then after a period of 5 minutes. The evaluator observed that the session is terminated after the configured time periods.

2.10.2 TSF-Initiated Termination (VPN Headend) - per TD0656 (VPNGW12:FTA_SSL.3/VPN)

2.10.2.1 VPNGW12:FTA_SSL.3.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to verify that it describes the ability of the TSF to terminate an inactive VPN client session.

Section 6.10 of the ST states that the TOE can also terminate VPN client sessions that have been inactive for the administrator defined time interval.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to verify that it provides instructions to the administrator on how to configure the time limit for termination of an active VPN client session.

The “Settings for VPN clients” section of the **Admin Guide** provides instructions for configuring the user timeout period by navigating to the TOE’s settings, selecting Add-ons then selecting User Authentication and entering the Idle Time-out in seconds.

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

Test 1: The evaluator shall follow the steps provided in the operational guidance to set the inactivity timer for five minutes. The evaluator shall then connect a VPN client to the TOE, let it sit idle for four minutes and fifty seconds, and observe that the VPN client is still connected at this time by performing an action that would require VPN access. The evaluator shall then disconnect the client, reconnect it, wait five minutes and ten seconds, attempt the same action, and observe that it does not succeed. The evaluator shall then verify using audit log data that the VPN client session lasted for exactly five minutes.

Test 2: The evaluator shall configure the inactivity timer to ten minutes and repeat Test 1, adjusting the waiting periods and expected audit log data accordingly.

Test 1: The evaluator configured the inactivity timer for five minutes, then connected a VPN client to the TOE and

waited for four minutes and fifty seconds and confirmed that the TOE did not close the session. The evaluator then established a connection and let the idle timeout expire and verified that the TOE terminated the session as expected. The VPN client log showed that the connection was terminated after 5 minutes of inactivity. The evaluator verified using audit log data that the VPN client session lasted five minutes.

Test 2: The evaluator repeated Test 1 with a time period of 10 minutes.

2.10.3 User-initiated Termination (NDcPP22e:FTA_SSL.4)

2.10.3.1 NDcPP22e:FTA_SSL.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.10 in the ST states that administrators using the GUI or local console (ie. CLI) can terminate their own session using the logoff commands provided by these interfaces.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins table entry provides the command 'logout' for logging out of the local console and the steps 'Menu > File > Exit' to log out of the GUI.

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

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

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

Test 1 - The evaluator established a local session and manually logged out per the command identified in the Guidance. The session was terminated and the console displayed only the logon banner and the prompt for a

username.

Test 2 - The evaluator established a remote session and manually logged out per the command identified in the Guidance. The session was terminated and the application closed requiring the user to reauthenticate to gain access.

2.10.4 TSF-initiated Session Locking (NDcPP22e:FTA_SSL_EXT.1)

2.10.4.1 NDcPP22e:FTA_SSL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.10 in the ST states that when an administrator uses the local console's CLI, the CLI enforces an inactivity timeout value that terminates the session after the administrator-specified time period.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins table entry provides the commands for specifying the timeout in seconds to terminate an inactive local administrative session.

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

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

Test 1 - The evaluator set the console time period to 2 minutes and observed a console timeout. The evaluator then set the console timeout to 5 minutes and observed a console timeout.

2.10.5 Default TOE Access Banners (NDcPP22e:FTA_TAB.1)

2.10.5.1 NDcPP22e:FTA_TAB.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.7 of the ST states that the TOE provides an administrator role. User accounts associated with the administrator role are considered Security Administrators. Security Administrators can manage and configure warning banner configuration. The Virtual SMC Appliance offers two administrative interfaces - command line and GUI. The NGFW Engine provides no administrator access at all.

The CLI is a text based interface which can be accessed from the virtual machine console in the VMware Host client. These command line functions can be used to query the current TOE version and update the Virtual SMC Appliance's firmware (an administrator must use the GUI to query and update NGFW Engine software), to manually set the Virtual SMC Appliance's time, and to configure the SMC CLI session time out.

The Virtual SMC Appliance also offers a non-CLI, remote interface for management. This remote interface offers access through the GUI client using TLS v1.2, and provides all management functionality except the commands available only through the CLI for manually setting the time and configuring the CLI session time out.

Section 6.10 of the ST states that the GUI offered by the Virtual SMC Appliance has a configurable banner that is displayed before a user's login. The banner contents are defined by the administrator through the GUI interface. This same banner is also displayed on the Virtual SMC Appliance local console CLI prior to a user's login.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Logon Banner table entry refers to the *Forcepoint Next Generation Firewall Product Guide* for instructions on configuring the logon banner.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 7 (Using the Management Client) The “Create logon banners for administrators” subsection provides instructions for creating a banner text showing all administrators information about the selected Management Server. The text from the banner also appears in the logon window of the local console.

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

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

Test 1 - The evaluator set a login banner for each method of access and then tested that the banner is displayed.

2.10.6 TOE Session Establishment - per TD0656 (VPNGW12:FTA_TSE.1)

2.10.6.1 VPNGW12:FTA_TSE.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to verify that it describes the methods by which the TSF can deny the establishment of an otherwise valid remote VPN client session (e.g., client credential is valid, not expired, not revoked, etc.), including day, time, and IP address at a minimum.

Section 6.10 of the ST states The TOE can deny VPN client sessions based upon location (as determined by IP address), time and day.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance to determine that it provides instructions for how to enable an access restriction that will deny VPN client session establishment for each attribute described in the TSS.

The “Modify firewall policy template” section of the **Admin Guide** describes how to restrict VPN client access based on IP address or time range. To restrict access based on IP address, time or day create a new rule validity time and fill in with the desired information.

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

Test 1: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it, noting the IP address from which the client connected. The evaluator shall follow the steps described in the operational guidance to prohibit that IP address from connecting, attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 2: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it. The evaluator shall follow the steps described in the operational guidance to prohibit the VPN client from connecting on a certain day (whether this is a day of the week or specific calendar date), attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 3: The evaluator shall successfully connect a remote VPN client to the TOE and then disconnect it. The evaluator shall follow the steps described in the operational guidance to prohibit the VPN client during a range of times that includes the time period during which the test occurs, attempt to reconnect using the same VPN client, and observe that it is not successful.

Test 4: [conditional] If any other attributes are identified in FTA_TSE.1, the evaluator shall conduct a test similar to tests 1 through 3 to demonstrate the enforcement of each of these attributes. The evaluator shall demonstrate a successful remote client VPN connection, configure the TSF to deny that connection based on the attribute, and demonstrate that a subsequent connection attempt is unsuccessful.

Test 1: The evaluator connected a remote VPN client to the TOE and recorded the IP address from which the client connected, then disconnected the client. The evaluator configured the TOE to prevent that IP address from connecting, then attempted to reconnect using the same VPN client, and observed that it was not successful.

Test 2: The evaluator connected a remote VPN client to the TOE and recorded the IP address from which the client connected, then disconnected the client. The evaluator configured the TOE to prevent that IP address from connecting on a certain day. The evaluator then attempted to reconnect using the same VPN client on the configured day, and observed that it was not successful.

Test 3: The evaluator connected a remote VPN client to the TOE and recorded the IP address from which the client connected, then disconnected the client. The evaluator configured the TOE to prevent that IP address from connecting during a range of times that includes the current time. The evaluator then attempted to reconnect using the same VPN client, and observed that it is not successful.

Test 4: Not applicable. No other attributes are claimed by the TOE.

2.10.7 VPN Client Management - per TD0656 (VPNGW12:FTA_VCM_EXT.1)

2.10.7.1 VPNGW12:FTA_VCM_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall check the TSS to verify that it asserts the ability of the TSF to assign a private IP address to a connected VPN client.

Section 6.10 of the ST states that an administrator can configure the TOE to assign private IP addresses to VPN clients.

Component Guidance Assurance Activities: There are no operational guidance EAs for this component.

There are no operational guidance Evaluation Activities for this SFR.

Component Testing Assurance Activities: The evaluator shall connect a remote VPN client to the TOE and record its IP address as well as the internal IP address of the TOE. The evaluator shall verify that the two IP addresses belong to the same network. The evaluator shall disconnect the remote VPN client and verify that the IP address of its underlying platform is no longer part of the private network identified in the previous step.

The evaluator connected a remote VPN client to the TOE and recorded its IP address with the internal IP address of the TOE. The evaluator verified that the two IP addresses belong to the same network. The evaluator disconnected the remote VPN client and verified that the IP address of its underlying platform after the disconnection was not part of the private network identified in the previous step.

2.11 Trusted path/channels (FTP)

2.11.1 Inter-TSF trusted channel - per TD0639 (NDcPP22e:FTP_ITC.1)

2.11.1.1 NDcPP22e:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.1.2 NDcPP22e:FTP_ITC.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.1.3 NDcPP22e:FTP_ITC.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.11 of the ST indicates that the TOE has trusted external IT entity communications with an external syslog server (protected by TLS) and with VPN peers (protected by IPsec).

For the syslog communication channel, the TOE acts as the TLS client during the negotiation of the TLS connection. The TOE supports the use of a client certificate (which an administrator can obtain from the internal CA, from an external CA, or can import), as the mechanism to authenticate the TOE to the syslog server. The administrator can also load trusted CA certificates to which the syslog server's certificate must chain.

For communications with VPN Clients, the TOE acts as an IKE responder, while for communications with VPN peers, the TOE can act as an IKE initiator or a responder. In all cases, the TOE and authenticates IKE peers and Clients through certificate exchange.

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

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table refers the reader to specific chapters in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 27 (Configuring the Log Server) provides instructions for configuring the TOE to forward audit data from the Log Server to an external syslog server as well as how to enable TLS protection for the forwarding of this data.
- Chapter 29 (Reconfiguring the SMC and engines) provides instructions for configuring the TOE to forward audit data from the Management Server to an external syslog server as well as how to enable TLS protection for the forwarding of this data.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Audit Server Configuration entry in the table also provides specific options that should be selected and defined when following the instructions in the online guide to set the audit data forwarding in the properties of the Management Server or Log Server. This includes specifying which options should be selected for the TLS certificate to use, defining a TLS profile element to include TLSv1.2 and the approved set of TLS ciphersuites, and defining the TLS server identity.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide** under the Audit Server configuration entry in the table, there are steps provided for what to do in the event that the connection to the audit server is broken. The following steps are provided:

- In the properties of the Management Server, verify the settings on the Audit Forwarding tab.
- In the properties of the Log Server, verify the settings on the Log Forwarding tab.
- Restart the Management Server on the local console. Use the command:

  sudo daemon-ctl restart sgMgtServer
- Restart the Log Server on the local console. Use the command:

  sudo daemon-ctl restart sgLogServer

In the "VPN Configuration" section of the **Admin Guide**, the methods necessary for an administrator to configure a IKE/IPsec with a peer gateway. The "Define endpoints for VPN Gateway elements", "Defining site elements for VPN gateways manually", and "Disable automated certificate management" describe the specifics.

In the "VPN Recovery Instructions" section of the **Admin Guide** instructions are provided for recovering broken VPN connections.

Component Testing Assurance Activities: The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

The evaluator shall perform the following tests:

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

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

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

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

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

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

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

Further assurance activities are associated with the specific protocols.

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

The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public- facing document or report.

Test 1 - Refer to NDcPP22E:FCS_TLSC_EXT.1/2 where the syslog TLS channel was fully tested and to NDcPP22E:FCS_IPSEC_EXT.1 where the IPsec channel was fully tested.

Test 2 - The TOE is able to initiate communication via the trusted channel for transmitting audit records to a secure syslog server and for transmitting IPsec traffic to a VPN peer. The evaluator configured the TOE's audit forwarding feature to forward the audits generated on the TOE's internal log server over a TLS protected connection to a secure syslog server. The evaluator used this configuration to capture many of the audit messages for the TOE and tested that this process was done over a trusted channel (TLS) in FCS_TLSC_EXT.1/2. In all of these test cases, the TOE initiates a connection with the test server. Likewise, for IPsec, the evaluator configured the TOE to initiate

IPsec connections to a VPN test peer and observed the TOE send an IKE_SA_Init message and negotiate the IPsec connection.

Test 3 - Packet captures generated during testing of each communications path to confirmed the TOE encrypted the channel data.

Test 4 - The evaluator physically disrupted the TOE's syslog-TLS connection (for 30 seconds) and IPsec connection (for 20 seconds) and then reconnected. The evaluator observed that both the TLS and IPsec sessions remained secure without requiring a new TLS or IKE/IPsec handshake and no TSF data was sent in plaintext. A second test with a 20 minute (for TLS) and 10 minute (for IPsec) timeout was also performed. Upon physically reconnecting, the evaluator observed the TOE negotiate a new TLS/IKE session/channel.

In both tests, the evaluator noted that at no time was any TSF data sent in plaintext. The evaluator analyzed the traffic in both cases and found no instance of sensitive data being sent outside the protected TLS or IPsec channel.

2.11.2 Inter-TSF Trusted Channel (VPN Communications) (VPNGW12:FTP_ITC.1/VPN)

2.11.2.1 VPNGW12:FTP_ITC.1.1/VPN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Refer to NDcPP22e:FTP_ITC.1 where these activities have been performed and applied to IPsec VPN communications.

Component Guidance Assurance Activities: The EAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Refer to NDcPP22e:FTP_ITC.1 where these activities have been performed and applied to IPsec VPN communications.

Component Testing Assurance Activities: TheEAs specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications. Additional evaluation testing for IPsec is covered in FCS_IPSEC_EXT.1.

Refer to NDcPP22e:FTP_ITC.1 where these activities have been performed and applied to IPsec VPN communications.

2.11.3 Trusted Path - per TD0639 (NDcPP22e:FTP_TRP.1/Admin)

2.11.3.1 NDcPP22e:FTP_TRP.1.1/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.3.2 NDcPP22e:FTP_TRP.1.2/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.3.3 NDcPP22e:FTP_TRP.1.3/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.7 of the ST states that the Virtual SMC appliance offers a remote HTTPS interface through the GUI client using TLSv1.2. The GUI client provides all management functionality except for the limited set of commands

available through the CLI.

Section 6.11 of the ST states that the administrator's Client GUI runs within an Internet browser running on the administrator's workstation and provides a graphical user interface only. All decisions on whether the operation is allowed occur in the Management Server with which the Client GUI communicates. The Management Server only accepts TLSv1.2 connections for the Client GUI and provides an HTTPS/TLS interface for using the Client GUI in a web browser. An administrator can also configure the TOE to provide VPN Clients with access to the SMC's Management Server, allowing a remote administrator to securely access the SMC through a browser tunneled within IPsec.

These protocols are consistent with those specified in the requirement.

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

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC Appliance TLS client and server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The "Enable FIPS mode on the SMC Appliance" section of the **Admin Guide** references the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the "Supporting documentation" section. This document describes how to enable FIPS mode and 256-bit encryption as the security strength during TOE installation, in order to restrict the cryptographic algorithms available in the TOE to those that are compatible with FIPS 140-2.

How to install Forcepoint NGFW in FIPS mode:

- The "Installing the SMC" section describes how to enable FIPS restrictions on the Management Server, the Log Server, and the Management client during installation. This includes selecting FIPS 140-2 mode and 256-bit encryption in the security selections when installing the SMC Management Server and Log Server.

- The "Install the Management Client using a file" subsection describes how to access, download and install the management client on a PC in the operating environment and then ensure that it is operating in FIPS

mode by selecting the 'Restricted Cryptographic Algorithms Compatible with FIPS 140-2' operating mode.

- The "Installing the NGFW Engine in FIPS mode" describes how to install and configure the NGFW engine and to select 'FIPS-Compatible Operating Mode' during the process of defining the NGFW engine properties.

The "Install the Management Client" section of the **Admin Guide** references the *Forcepoint Next Generation Firewall Installation Guide* for instructions regarding how to log on the Management Client and verify the fingerprint of the Management Server certificate. The *Forcepoint Next Generation Firewall Installation Guide* is accessed via the link found in the "Supporting documentation" section.

Section "Configure settings for an evaluated configuration" in the **Admin Guide** provides instructions for configuring the SMC Web Access feature to use the management client in a web browser for HTTPS connections. The instructions include generating an ECDSA certificate request, importing a signed certificate and configuring the TLS ciphersuite set. The reader is referred to the *Forcepoint Next Generation Firewall Product Guide* for instructions to enable SMC Web Access.

Forcepoint Next Generation Firewall Product Guide

- Chapter 25 (Using the Management Client in a web browser) provides the detailed steps for enabling SMC Web Access on the Management Server in order to run the Management Client in a web browser.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table states that the Management client (GUI) should be used to manage users and passwords in the SMC. The local console user accounts are synchronized with the user accounts used in the SMC and are managed from the SMC. This section also refers the reader to the chapter about Administrator accounts in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide

- Chapter 21 (Administrator Accounts) provides instructions for creating administrative users under the "Adding administrator accounts" subsection. This includes configuring administrator authentication to use a password stored in the internal database of the SMC and configuring an administrator with unrestricted permissions which allows logon via the local console.

In the "Configure settings for an evaluated configuration" the Password Guidelines entry in the table refers the reader to further information on the topic about enabling and defining password policy settings in the Forcepoint *Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) - subsection "Enable and define password policy settings" provides

the steps for configuring and/or changing password policy settings. After accessing the Global System Properties Menu and clicking on the Password Policy tab, the user is instructed to click on *Enforce Password Settings for All the Administrators and Web Portal Users* and then select the password policy settings. At this point, the product guide refers back to the **Admin Guide** for the options that must be selected in the evaluated configuration. As noted above, the **Admin Guide** recommends that the "Minimum Number of Required Characters" setting should be used to configure a minimum password length of 15.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Logon Banner entry in the table refers to the chapter about using the Management Client in the *Forcepoint Next Generation Firewall Product Guide* which is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 7 (Using the Management Client) - The "Create logon banners for administrators" subsection provides instructions for creating a banner text showing all administrators information about the selected Management Server. The text from the banner also appears in the logon window of the local console.

In the "Configure settings for an evaluated configuration" section of the **Admin Guide**, the Administrative Logins entry in the table provides a reference to the *Forcepoint Next Generation Firewall Product Guide* where the instructions for setting session timeouts via the GUI can be found. This document is accessed via the link found in the "Supporting documentation" section.

Forcepoint Next Generation Firewall Product Guide:

- Chapter 21 (Administrator Accounts) provides instructions for enabling and defining the password policy which includes defining session limits and idle timeouts.

In the "Settings for VPN clients" section of the **Admin Guide**, the instructions detail how to configure the TOE to accept VPN Client connections and to configure VPN access rules and rules to encrypt traffic from the local site to the remote site, which an administrator can use to remotely access the SMC's Web UI.

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

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

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

Further assurance activities are associated with the specific protocols.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.

Test 1 -2: The successful testing of the remote administration channel and the demonstration of its encryption can be found in FCS_TLSS_EXT.1 for HTTPS on the SMC device. The evaluator verified that the results from the FCS_TLSS_EXT.1 tests were using the correct protocol and that there was no channel data being sent in plaintext.

The evaluator performed the above tests for IPsec to demonstrate that through a VPN Client running on the (evaluator's) administrator's workstation, the administrator could remotely administer the SMC within a secure IPsec tunnel. The evaluator captured traffic from the session and inspected it, confirming that the channel data was encrypted with IPsec/ESP.

2.11.4 Trusted Path - per TD0639 (NDcPP22e:FTP_TRP.1/Join)

2.11.4.1 NDcPP22e:FTP_TRP.1.1/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.4.2 NDcPP22e:FTP_TRP.1.2/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.11.4.3 NDcPP22e:FTP_TRP.1.3/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that the methods of joining components to the TOE are identified, along with how those communications are protected, including identification of whether the environment is required to provide confidentiality of the communications or whether the registration data exchanged does not require confidentiality. If the TSS asserts that registration data does not require confidentiality protection then the evaluator shall examine the justification provided to confirm that.

The evaluator shall also check that all protocols listed in the TSS in support of this process are included in the SFRs in the ST, and that if the ST uses FTP_TRP.1/Join for the registration channel then this channel cannot be reused as the normal inter-component communication channel (the latter channel must meet FTP_ITC.1 or FPT_ITT.1).

The evaluator shall examine the TSS to confirm that sufficient information is provided to determine the TOE actions in the case that the initial component joining attempt fails.

Section 6.11 of the ST states that the SMC Management Server only accepts join requests from Engines for which the Administrator has already created an object and provided the Engine with the SMC generated password. Furthermore, the SMC protects the registration channel using TLSv1.2 and requires that the registering Engine prove knowledge of the SMC generated password by exchanging a SHA-512 hash. The SMC reserves port 3021 for registration and after registration, the components communicate using mutually-authenticated TLS on different ports (8903, 8907, 8906, 8916, 8917, 3020, 3023), thus preventing reuse of the registration channel. Should a registration attempt fail, the administrator can attempt again, or can generate a new password on the SMC (and input that password into the Engine), and then attempt registration again.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to confirm that it contains instructions for establishing and using the enablement and registration channel. The evaluator shall confirm that the guidance documentation makes clear which component initiates the communication. The evaluator shall confirm that the guidance documentation contains recovery instructions should a connection be unintentionally broken during the registration process.

In the case of a distributed TOE that relies on the operational environment to provide security for some aspects of the registration channel security then there are particular requirements on the Preparative Procedures as listed below. (Reliance on the operational environment in this way is indicated in an ST by a reference to operational guidance in the assignment in FTP_TRP.1.3/Join.) In this case the evaluator shall examine the Preparative Procedures to confirm that they:

a) clearly state the strength of the authentication and encryption provided by the registration channel itself and the specific requirements on the environment used for joining components to the TOE (e.g. where the environment is relied upon to prevent interception of sensitive messages, IP spoofing attempts, man-in-the-middle attacks, or race conditions)

b) identify what confidential values are transmitted over the enablement channel (e.g. any keys, their lengths, and their purposes), use of any non-confidential keys (e.g. where a developer uses the same key for more than one device or across all devices of a type or family), and use of any unauthenticated identification data (e.g. IP addresses, self-signed certificates)

c) highlight any situation in which a secret value/key may be transmitted over a channel that uses a key of lower comparable strength than the transmitted value/key. Comparable strength is defined as the amount of work required to compromise the algorithm or key and is typically expressed as 'bits' of security. The ST author and evaluator should consult NIST 800-57 Table 2 for further guidance on comparable algorithm strength.

Section “Establishing a security configuration” in the **Admin Guide** provides the steps for configuring the SMC appliance and NGFW Engine into the evaluated configuration.

The “Enable FIPS mode on the SMC Appliance” section of the **Admin Guide** states that in order to comply with Common Criteria evaluation standards, FIPS mode and 256-bit encryption as the security strength must be enabled when the SMC Appliance is installed. When 256-bit encryption is enabled, the SMC TLS Client and Server settings are automatically configured to use:

- TLS 1.2 as the protocol
- ECDSA P-521 certificates with SHA-512 in digital signatures
- P-521, P-384, and P-256 NIST curves in TLS key establishment

The Management Server and Log Server accept the following TLS cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384

The user is instructed to use the main network interface for management for the connection to the NGFW Engine. For specific instructions, this section refers the reader to the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the “Supporting documentation” section.

Section “Create an element for the NGFW Engine” in the **Admin Guide** provides the steps to use the Management Client to create an NGFW Engine element.

Section “Enabling communication between the SMC and NGFW Engine” in the **Admin Guide** states that after the initial contact is made, subsequent TLS connections between components are mutually authenticated. The reference identifiers in the SAN DNS field for subsequent TLS client and server authentication are configured automatically during the registration.

The “Save the initial configuration in the Management Client” section in the **Admin Guide** describes how to save the initial NGFW Engine configuration in the Management Client and then select the Firewall Policy that will be automatically installed on the NGFW Engine after initial contact is made. It also instructs the user to make a note of the one-time generated password, the Management server address, and the SHA-512 certificate fingerprint as this information is needed when installing the NGFW Engine.

Section “Install NGFW Engine in FIPS mode” in the **Admin Guide** indicates that FIPS mode and 256 bit encryption must be enabled when the NGFW engine is installed. 256-bit encryption with the TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 TLS cipher suite is used for the connection between the NGFW Engine and the Management Server. If the initial contact fails, the user is instructed to restart the appliance, start the NGFW Configuration Wizard again, and verify that the following are correct: the one time password, Management server IP address, certificate fingerprint and that 256 bit encryption is used for the connection to the Management server. Further instructions are also provided in the *How to install Forcepoint NGFW in FIPS mode* document which is accessed via the link found in the “Supporting documentation” section.

How to install Forcepoint NGFW in FIPS mode

- The "Installing the SMC Appliance in FIPS mode" section provides instructions for selecting FIPS 140-2 mode during installation.
- The "Install the SMC components" section provides instructions for enabling FIPS restrictions on the Management Server, Log Server and Management Client during the installation.
- The "Create an element for the NGFW Engine" section provides the steps for creating an NGFW Engine object in the Management Client. A note here indicates there is a one-time password created that allows establishing trust with the Management Server.
- The "Install the NGFW Engine in FIPS mode" provides the instructions for upgrading the software and setting the kernel in FIPS mode after reboot. Configuration steps after reboot include selecting FIPS compatible operating mode.

Section "Disabling communication between the SMC and NGFW Engine" in the **Admin Guide** provides instructions for disabling communication from the SMC by deleting the NGFW Engine element in the Management Client, and resetting the NGFW Engine to factory settings.

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

a) Test 1: The evaluator shall ensure that the communications path for joining components to the TSF is tested for each distinct (non-equivalent) component type [The intention here is to cover all different software sections involved. For example, a single software image may be installed on different TOE components, but with different sections of the image executed according to the hardware platform or communications stack. In such as case tests should be carried out for each different software section.], setting up the connections as described in the guidance documentation and ensuring that communication is successful. In particular the evaluator shall confirm that requirements on environment protection for the registration process are consistent with observations made on the test configuration (for example, a requirement to isolate the components from the Internet during registration might be inconsistent with the need for a component to contact a license server). If no requirements on the registration environment are identified as necessary to protect confidentiality, then the evaluator shall confirm that the key used for registration can be configured (following the instructions in the guidance documentation) to be at least the same length as the key used for the internal TSF channel that is being enabled. The evaluator shall confirm that the key used for the channel is unique to the pair of components (this is done by identifying the relevant key during the registration test: it is not necessary to examine the key value).

b) Test 2: The evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be enabled by a Security Administrator for all the TOE components identified in the guidance documentation as capable of initiation.

c) Test 3: The evaluator shall ensure that if the guidance documentation states that the channel data is encrypted then the data observed on the channel is not plaintext.

Further assurance activities are associated with the specific protocols.

Test 1: The joining of the SMC and Engine during the registration process is demonstrated in NDcPP22e:FCO_CPC_EXT.1. A one-time generated password is used for registration. According to operation guidance, to comply with Common Criteria evaluation standards, you must enable FIPS mode and enable 256-bit encryption as the security strength when you install the NGFW Engine. These steps were performed during the registration process.

Test 2: The joining of the SMC and Engine during the registration process is demonstrated in NDcPP22e:FCO_CPC_EXT.1.

Test 3: The evaluator ran a packet capture during the initiation of the registration process by the NGFW N60 Engine to the SMC that was demonstrated in NDcPP22e:FCO_CPC_EXT.1. The evaluator viewed that traffic during this joining process was encrypted.

3. Protection Profile SAR Assurance Activities

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

3.1 Development (ADV)

3.1.1 Basic Functional Specification (ADV_FSP.1)

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

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

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

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

The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. The intent is that these interfaces will be adequately tested, and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied.

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

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

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

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

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

However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

For this PP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional 'functional specification' documentation is necessary to satisfy the Evaluation Activities specified in the SD.

3.2 Guidance documents (AGD)

3.2.1 Operational User Guidance (AGD_OPE.1)

Assurance Activities: The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps (per TD0536):

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR.

In addition, the evaluator performs the EAs specified below.

The evaluator shall ensure the Operational guidance documentation is distributed to administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

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

The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.

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

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

b) The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

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

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

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

As identified throughout this AAR, the **Admin Guide** provides instructions for configuring the TOE's cryptographic security functions. The **Admin Guide** provides instructions for configuring FIPS mode for each TOE component such that only the cryptographic algorithms and parameters used for the evaluated configuration are available and that it is clear that no other cryptographic engines have been evaluated or tested. There are warnings and notes throughout the **Admin Guide** regarding use of functions that are and are not allowed in the evaluated configuration. There are also specific settings identified that must be enabled or disabled in order to remain CC compliant. The process for updating the TOE is described above in NDcPP22e:FPT_TUD_EXT.1.

3.2.2 Preparative Procedures (AGD_PRE.1)

Assurance Activities: As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

It is noted that specific requirements for Preparative Procedures are defined in [SD] for distributed TOEs as part of the Evaluation Activities for FCO_CPC_EXT.1 and FTP_TRP.1(2)/Join.

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

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

In addition, the evaluator performs the EAs specified below.

The evaluator shall examine the Preparative procedures to ensure they include a description of how the administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).

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

The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

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

The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.

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

The preparative procedures must

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

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

The evaluation team had the following documents to use when configuring the TOE:

- Forcepoint Next Generation Firewall 6.10 Common Criteria Evaluated Configuration Guide, Revision C (**Admin Guide**)

In some instances, the document referenced general Forcepoint manuals which the evaluator could access via web links provided in the Supporting Documentation section. The completeness of the documentation is addressed by their use in the AA's carried out in the evaluation.

3.3 Life-cycle support (ALC)

3.3.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activities: This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a 'hard label' (e.g., stamped into the metal, paper label) or a 'soft label' (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

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

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

3.3.2 TOE CM Coverage (ALC_CMS.1)

Assurance Activities: Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

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

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

3.4 Tests (ATE)

3.4.1 Independent Testing - Conformance (ATE_IND.1)

Assurance Activities: Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes 'evaluated configuration' instructions). The focus of the testing is to confirm that the requirements specified in Section 5.1.7 are being met. The Evaluation Activities in [SD] identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

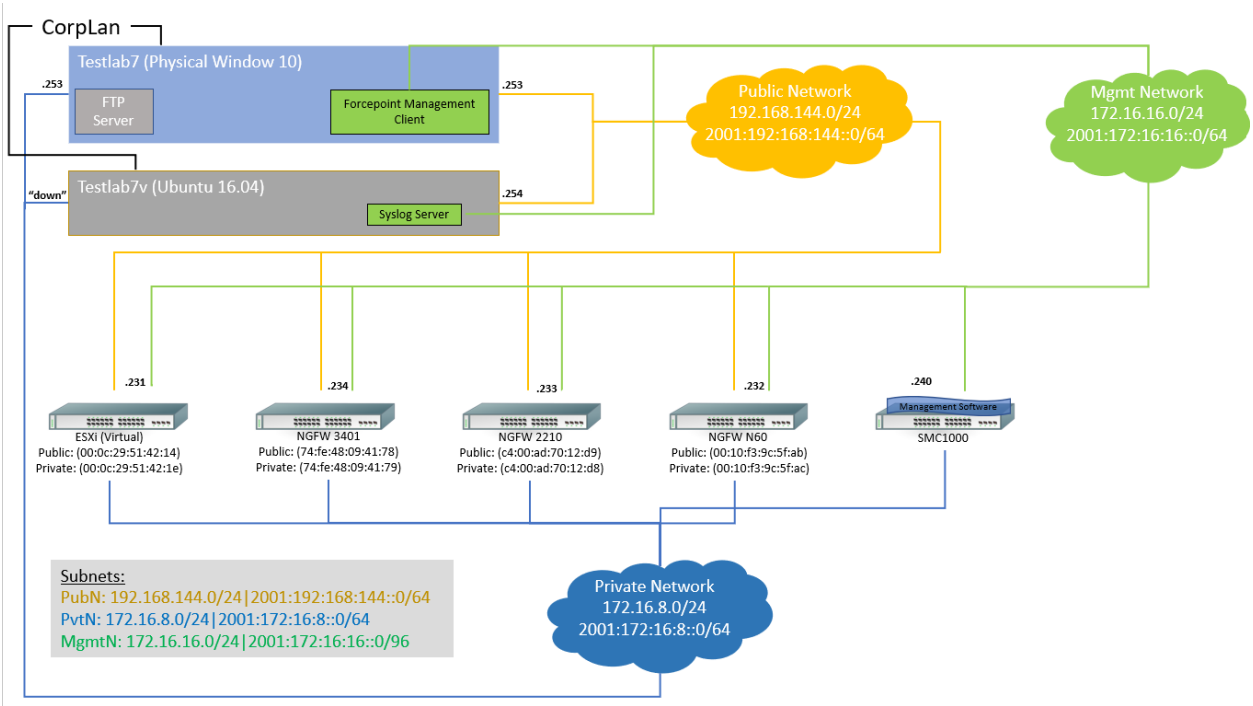
The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

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

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

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

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



TOE Platforms:

- Virtual SMC Appliance running software version 6.10.9
- N60 running software version 6.10.9
- 2210 running software version 6.10.9
- 3401 running software version 6.10.9
- Virtual NGFW Engine Appliance running software version 6.10.9

Supporting Software:

- SSH Client – Putty version 6.2
- SSH Client – SecureCRT version 5.1.2
- Big Packet Putty version 6.2
- Wireshark version 1.10.0 (3 instances identified above)
- Nmap version 6.25
- Windows 10

The Gossamer Test servers with an Ubuntu environment acted as platforms to initiate testing. The test servers also acted as a syslog server and an ntp server.

- Openssl version 1.0.2g
- Rsyslog version 8.16.0
- ntpd 4.2.8p4
- Tcpdump version 4.9.3
- Libpcap version 1.7.4
- Nmap version 7.01
- Stunnel 5.30

3.5 Vulnerability assessment (AVA)

3.5.1 Vulnerability Survey (AVA_VAN.1)

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

the approach defines the minimum level of analysis and the scope of that analysis, and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities.

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

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

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

The evaluator shall examine the documentation outlined below provided by the developer to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.

The developer shall provide documentation identifying the list of software and hardware components7 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 the TOE) such as a web server and protocol or cryptographic libraries. 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 hypotheses during their analysis.

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

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

b) a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, 6.3.3]

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.5.1.2 and 3.6.1.2.

The evaluator formulates hypotheses in accordance with process defined in Appendix A. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2. The results of the analysis shall be documented in the report according to Appendix A.3.

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

The evaluation team performed a public search for vulnerabilities in order to ensure there are no publicly known and exploitable vulnerabilities in the TOE from the following sources:

- National Vulnerability Database (<https://web.nvd.nist.gov/vuln/search>)
- Vulnerability Notes Database (<http://www.kb.cert.org/vuls/>)
- Rapid7 Vulnerability Database (<https://www.rapid7.com/db/vulnerabilities>)
- Tipping Point Zero Day Initiative (<http://www.zerodayinitiative.com/advisories>)
- Exploit / Vulnerability Search Engine (<http://www.exploitsearch.net>)
- SecurITeam Exploit Search (<http://www.securiteam.com>)
- Tenable Network Security (<http://nessus.org/plugins/index.php?view=search>)
- Offensive Security Exploit Database (<https://www.exploit-db.com/>)

The search was performed on 04/04/2023 with the following search terms: "Forcepoint", "SMC", "Openssl", "NGFW 6.10", "ESXi 7.0", "Bouncy Castle", "Intel Atom", "Dell PowerEdge R440", "Intel Xeon Silver ", "2U Xeon 4210 (Cascade Lake)", "1U Xeon D-2177NT (Skylake)", "Desktop Atom C3338 (Denverton)", "SMC FIPS Java API", "Xeon Gold", "SMC FIPS Cryptographic Module for NTP", "SMC FIPS Library", "Forcepoint NGFW FIPS Cryptographic Module".

3.5.2 Additional Flaw Hypotheses (AVA_VLA.1)

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

Not applicable. The TOE is not vulnerable to Bleichenbacher attacks because it does not use RSA for server-side connections.