Assurance Activities Report

for

Fortinet FortiMail Version 7.4

Version 1.0

24 July 2024

Prepared by:

Leidos Inc.
https://www.leidos.com/CC-FIPS140
Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive
Columbia, MD 21046

Prepared for:

Fortinet, Inc.
899 Kifer Road
Sunnyvale, CA 94086 USA

The Developer of the TOE:

Fortinet, Inc.
899 Kifer Road
Sunnyvale, CA 94086 USA

The TOE Evaluation was Sponsored by:

Fortinet, Inc.
899 Kifer Road
Sunnyvale, CA 94086 USA

Evaluation Personnel:

Tony Apted
Justin Fisher
Kofi Owusu
Pascal Patin

Contents

1 Introduction

This document presents results from performing evaluation activities associated with the Fortinet FortiMail Version 7.4 evaluation. This report contains sections documenting the performance of evaluation activities associated with each of the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) as specified in Evaluation Activities for Network Device cPP, Version 2.2, December 2019 [ND SD].

Note that, in accordance with NIAP Policy Letter #5, all cryptography in the TOE for which NIST provides validation testing of FIPS-approved and NIST-recommended cryptographic algorithms and their individual components must be NIST validated. The CCTL will verify that the claimed NIST validation complies with the NIAP-approved cPP requirements the TOE claims to satisfy. The CCTL verification of the NIST validation constitutes performance of the associated evaluation activity. As such, test activities associated with functional requirements within the scope of Policy Letter #5 are performed by verification of the relevant CAVP certification and not through performance of any testing as specified in the cPP or its supporting document.

1.1 Applicable Technical Decisions

The NIAP Technical Decisions (TDs) that apply to [NDcPP] as well as rationale for those TDs that do not apply to this evaluation are identified in the ETR as specified by Labgram #105/Valgram #125.

1.2 Evidence

The following evidence was used to complete the evaluation activities.

[ST]	Fortinet FortiMail Version 7.4 Security Target, Version 1.0, April 19, 2024
[CCECG]	FIPS 140-3 and Common Criteria Technote FortiMail 7.4, Version 0.2, July 8, 2024
[ADMIN]	Administration Guide, FortiMail 7.4.1, December 5, 2023
[CLI]	CLI Reference, FortiMail 7.4.1, September 28, 2023
[TEST]	Fortinet FortiMail Version 7.4 Common Criteria Test Report and Procedures for Network Device collaborative PP, Version 2.2e, Version 1.0, May 29, 2024
[VA]	Fortinet FortiMail Version 7.4 Vulnerability Assessment, Version 1.1, July 8, 2024
[ETR]	Evaluation Technical Report for Fortinet FortiMail Version 7.4, Version 1.0, July 24, 2024

1.3 Conformance Claims

Common Criteria Versions

- Common Criteria for Information Technology Security Evaluation Part 1: Introduction, Version 3.1, Revision 5, dated: April 2017.
- Common Criteria for Information Technology Security Evaluation Part 2: Security Functional Components, Revision 5, dated: April 2017.
- Common Criteria for Information Technology Security Evaluation Part 3: Security Assurance Components, Revision 5, dated: April 2017.

Common Evaluation Methodology Versions

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

Protection Profiles

- [NDcPP] collaborative Protection Profile for Network Devices, Version 2.2e, March 23, 2020
- [ND-SD] Evaluation Activities for Network Device cPP, Version 2.2, December 2019

2 Security Functional Requirement Evaluation Activities

This section describes the evaluation activities associated with the SFRs defined in the ST and the results of those activities as performed by the evaluation team. The evaluation activities are derived from [ND-SD] and modified by applicable NIAP TDs. Evaluation activities for SFRs not claimed by the TOE have been omitted.

2.1 Security Audit (FAU)

2.1.1 Audit Data Generation (FAU_GEN.1)

2.1.1.1 TSS 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.

[ST] Section 6.1 identifies the certificate label as being associated with key data in the case of X.509, and the admin username and IP address as being associated with key data in the case of SSH keys.

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.

The TOE is not distributed and therefore this is not applicable.

2.1.1.2 Guidance 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 “NDcPP Common Criteria Logging Addendum” section of [CCECG] lists the auditable events. This includes the events required by FAU_GEN.1.1 and the specific additional auditable events for the claimed SFRs.

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 evaluator examined the supplied guidance documentation, identifying all mechanisms available to the administrator for configuring and managing the capabilities of the TOE. Those mechanisms related to the SFRs specified in the ST were identified and mapped to the applicable SFRs. In addition, the evaluator sought to confirm that all SFRs that would be expected to have a management capability related to them had appropriate management capabilities identified in the guidance documentation.

Based on the claims made for the management functionality of the TSF, the following administrative actions were considered to necessitate auditable events:

- Ability to configure the access banner
- Ability to configure the session inactivity time before session termination
- Ability to update the TOE
- Ability to configure authentication failure parameters for FIA_AFL.1
- Ability to modify the behavior of the transmission of audit data to an external IT entity (specifically the act of configuring a connection to a remote log server)
- Ability to manage the cryptographic keys (specifically the act of generating a key pair as part of CSR generation)
- Ability to manage the trusted public keys database (specifically the act of associating public keys with SSH administrators)
- Ability to set the system clock
- Ability to add/remove CA certificates to/from the TOE's trust store
- Ability to change the minimum password length

The evaluator verified that the "NDcPP Common Criteria Logging Addendum" contained sample audit records for each of these events. The evaluator verified that for all cases except for those explicitly noted in section 6.4 of [ST] as being doable on the CLI only, the sample events were recorded for both CLI and GUI usage.

2.1.1.3 Test 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.

The evaluator collected audit records throughout testing. An appropriate audit record was generated for every event listed in FAU_GEN.1.1 and in Table 4.

The evaluator examined the audit records generated as part of this test activity. It was found that each record contained the date and time of the relevant event, along with an ID, event type, even description and an associated user where appropriate. This is consistent with the audit record format described in the NDcPP Common Criteria Logging Addendum.

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.

N/A – The TOE is not distributed.

2.1.2 User Identity Association (FAU_GEN.2)

2.1.2.1 TSS & Guidance 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.

2.1.2.2 Test 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.

N/A – The TOE is not distributed.

2.1.3 Protected Audit Event Storage (FAU_STG_EXT.1)

2.1.3.1 TSS 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.

[ST] Section 6.1 states that remote logging happens concurrently with local logging.

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.

[ST] Section 6.1 states that 80% of the TOE storage is reserved for the audit partition (with an example that the largest appliance drive is 20 TB which would mean 16 TB for audit records in this case). When the partition is 'full' (based on whether 1.5 GB is left or 5% capacity is left, whichever amount is smaller), the oldest log records begin to get overwritten with new data. This section also states that authorized administrators can view the records but that no ability to modify or delete the records is provided.

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

[ST] Section 6.1 states that the TOE is a standalone device that generates audit records that reside in its local storage.

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

The SFR has been completed with "overwrite previous audit records according to the following rule: the oldest audit records are overwritten". [ST] section 6.1 states that logs are overwritten, oldest first, upon exhaustion of log storage. This section goes on to state that exhaustion is defined as less than 5% or 1.5 GB remaining on the audit storage partition (which is 80% of the total storage volume), whichever value is lower.

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.

[ST] Section 6.1 states that the TSF is configured to transmit log data to a remote FAZ (Fortinet audit server) over TLS in real-time.

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

The TOE is not distributed and therefore this is not applicable.

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.

The TOE is not distributed and therefore this is not applicable.

2.1.3.2 Guidance 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.

[CCECG] section “Log Specific Settings” clearly identifies that the intended audit server is a FortiAnalyzer and that a connection to this is made to one or more audit servers over TLS. This section includes how to configure the server’s X.509 certificate, how to configure the trust store on the TOE to recognize the validity of that certificate, and how to configure the TOE to connect to the 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.

“Log Specific Settings” in [CCECG] describes the log transmission as happening in real-time over syslog with no buffering.

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.

The TOE does not offer any configuration options for FAU_STG_EXT.1.3 and therefore this is implicitly satisfied.

2.1.3.3 Test 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 tests for this requirement:

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.

The evaluator established a secured session between TOE and audit server, confirming that data was not transmitted in the clear and was successfully received by the server.

For this test the evaluator had the TOE export audit records to a Linux test server running OpenSSL 1.1.1. This allowed the test server to receive audit records over an encrypted TLS channel as demonstrated by the test results. Once the TOE was configured with the test server's information audit records were exported automatically and without administrator information. Additionally, for other tests (FAU_GEN.1) the TOE was configured to send audit records to a server running rsyslogd 8.32.0.

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 behavior 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).

The evaluator generated audit data and verified the data is stored locally. The evaluator was also able to verify that existing audit data was overwritten when file size limit was reached.

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

N/A – The TOE does not claim FAU_STG_EXT.2/LocSpace.

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.

N/A – The TOE does not claim FAU_STG_EXT.2/LocSpace.

2.2 Cryptographic Support (FCS)

2.2.1 Cryptographic Key Generation (FCS_CKM.1)

2.2.1.1 TSS 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.

[ST] Section 6.2 states the TOE generates asymmetric keys for SSH, TLS, and X.509 certificate signing requests. The TOE generates ECC keys using P-256, P-384, and P-521 and FFC keys using 2048-bit and 4096-bit MODP safe primes per RFC 3526. FFC keys are used for both SSH and TLS while ECC keys are used exclusively for TLS. The TOE also generates 2048-bit and 3072-bit RSA keys as part of generating certificate signing requests per FIA_X509_EXT.3.

2.2.1.2 Guidance 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 TOE supports key generation functions in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

2.2.1.3 Test 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.

Section 6.2 of [ST] (“Cryptographic Support”), Table 6 identifies the CAVP certifications verifying asymmetric key generation, as follows.

Functions	Standards	Certificates
RSA Schemes (2048, 3072-bit)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3	A5164 A5175

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 filed prime p shall both be probabl 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.

Section 6.2 of [ST] ("Cryptographic Support"), Table 6 identifies the CAVP certifications verifying asymmetric key generation, as follows.

Functions	Standards	Certificates
FFC Schemes (2048-bit DSA)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1	A5164 A5175

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.

Section 6.2 of [ST] ("Cryptographic Support"), Table 6 identifies the CAVP certifications verifying asymmetric key generation, as follows.

Functions	Standards	Certificates
ECC Schemes (ECDSA P-256, P-384, P-521 curves)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4	A5164 A5175

Modified per TD0580.

FFC Schemes using "safe-prime" groups

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

2.2.2 Cryptographic Key Establishment (FCS_CKM.2)

2.2.2.1 TSS Activities

Modified per TD0580.

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.

[ST] section 5.2.2.2 identifies the following key establishment methods supported by the TOE:

- Elliptic curve-based key establishment schemes
- FFC Schemes using “safe-prime” groups (modp2048 per RFC 3526).

These key establishment methods correspond to the key generation schemes specified in FCS_CKM.1.1 except that FCS_CKM.1.1 also claims RSA, which is exclusively for certificate key pair generation and not for key establishment. Section 6.2 states that the TOE generates ECC and FFC keys in support of TLS key establishment.

The TOE uses both ECDH and DH key establishment for TLS, depending on the negotiated ciphersuite. ECDH key establishment uses P-256, P-384, and P-521 as the supported curves, and DH key establishment uses modp2048 (DH group 14 per RFC 3526). SSH key establishment uses DH group 14.

The evaluator created the following table corresponding to the descriptions and determined that the description in the ST was complete.

Scheme	SFR	Service
Elliptic curve-based	FCS_TLSC_EXT.1 FCS_TLSC_EXT.2	Outbound syslog connectivity (TLS)
	FCS_TLSS_EXT.1	Remote administration (TLS/HTTPS)
FFC Schemes using “safe-prime” groups (groups listed in RFC 3526)	FCS_SSHS_EXT.1 FCS_TLSC_EXT.1 FCS_TLSC_EXT.2 FCS_TLSS_EXT.1	Outbound syslog connectivity (TLS) Remote administration (TLS/HTTPS) Remote administration (SSH)

2.2.2.2 Guidance 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).

The TOE supports key establishment functions in support of SSH and TLS communications. The “Administration Specific Changes” section of [CCECG] states that “The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled.” This is stated in the context of remote access. Under “Log Specific Settings,”

[CCECG] says that “cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable.” Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

2.2.2.3 Test 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.

Performed in accordance with NIAP Policy Letter #5.

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.

Section 6.2 of [ST] ("Cryptographic Support"), Table 6 ("Validated Algorithm Implementations") identifies the CAVP certifications verifying SP 800-56A Revision 3 key establishment schemes, as follows.

Functions	Standards	Certificates
Elliptic curve-based scheme (ECDSA) P-256, P-384, P-521	NIST Special Publication 800-56A Revision 3	A5164 A5175

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.

Not applicable. The TOE does not use RSA key establishment schemes.

Modified per TD0580 (section removed).

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.

Functions	Standards	Certificates
Finite field-based scheme: FFC Schemes (FB/FC)	NIST Special Publication 800-56A Revision 3, RFC 3526	A5164 A5175
FFC Schemes using 'safe-prime' groups (2048-bit MODP, 4096 bit MODP)	NIST Special Publication 800-56A Revision 3, RFC 3526	A5164 A5175

2.2.3 Cryptographic Key Destruction (FCS_CKM.4)

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

[ST] Section 6.2 states that the TOE includes both plaintext static keys that reside in nonvolatile memory and plaintext ephemeral keys that reside in volatile memory. For each key, the origin, storage location, and method of destruction (if applicable) is included. The evaluator reviewed the stored keys and identified that the keys listed are consistent with the expected usage of the TOE based on its use of trusted communications, reliance on local administration, and need for integrity of itself (for self-tests) and software updates.

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).

[ST] Section 6.2 indicates that the static keys maintained by the TOE in persistent storage are the firmware update key, SSH server key, firmware integrity key, and TLS host keys. These can be manually destroyed through direct overwrite via the execute erase-filesystem command. The SSH host key can also be destroyed via exec ssh-regen-keys command, which overwrites the existing key with a new value of the same key.

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.

[ST] section 5.2.2.3 selects 'single overwrite' (for volatile memory) so this evaluation activity does not apply to volatile memory.

For nonvolatile memory, the ST does select 'invocation of an interface'. [ST] section 6.2 describes the method of overwriting keys in nonvolatile storage as being either a command to regenerate SSH keys or to erase the file system. Per [ST], these methods are direct system calls to overwrite stored filesystem data with zeroes without the use of an intermediary API.

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 TSS does not identify any keys stored in non-plaintext form and so this is not applicable.

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.

[ST] does not identify any exceptions to the claimed behavior.

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.

The ST does not claim the use of "a value that does not contain any CSP" and therefore this activity is not applicable.

2.2.3.2 Guidance 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).

[CCECG] section "Key Zeriozation" describes how to manually erase static secret/private keys. This is done through erasing the filesystem or through deleting individual local certificates. Ephemeral keys that are destroyed as part of normal operation of the TOE are not discussed here because it is outside the scope of the TOE's operation since the administrator has no ability to control that function. The guidance does not identify any exceptions to where the described process is used to erase keys in non-volatile storage, consistent with the ST.

2.2.3.3 Test Activities

None.

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

2.2.4.1 TSS 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.

[ST] Section 6.2 states that the TOE supports AES-CBC and AES-GCM with both 128-bit and 256-bit sizes. This is claimed for both SSH and TLS.

2.2.4.2 Guidance 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 TOE supports symmetric encryption functions in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

2.2.4.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support", Table 1: Validated Algorithm Implementations) identifies the CAVP certifications verifying AES encryption and decryption, as follows.

Functions	Standards	Certificates
AES in CBC mode (128, 256 bits)	AES as specified in ISO 18033-3 CBC as specified in ISO 10116	A5164 A5175
AES in GCM mode (128, 256 bits)	AES as specified in ISO 18033-3 GCM as specified in ISO 19772	A5164 A5175

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

2.2.5.1 TSS 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.

[ST] Section 6.2 states that SHA-1, SHA-256, SHA-384, and SHA-512 are claimed. SHA-1, SHA-256, and SHA-384 are used in support of TLS. SHA-1, SHA-256, and SHA-512 are used in support of SSH. SHA-256 is used

for administrator password hashing, integrity checks for self-testing, and integrity checks for software updates.

2.2.5.2 Guidance Activities

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

The TOE supports hash functions in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

Hash functions used for integrity testing and validation of updates are not configurable.

2.2.5.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support", Table 2: Validated Algorithm Implementations) identifies the CAVP certifications verifying cryptographic hashing, as follows.

Functions	Standards	Certificates
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A5164 A5175

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

2.2.6.1 TSS 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.

[ST] Section 6.2 states that HMAC-SHA-1 (160 bit key/block size and output MAC), HMAC-SHA-256 (256 bit key/block size and output MAC), and HMAC-SHA-384 (384-bit key/block size and output MAC) are used by the TOE's HMAC function. HMAC-SHA-1/256 are used for SSH. HMAC-SHA-1/256/384 are used for TLS. HMAC-SHA-256 is used for update integrity and self-tests.

2.2.6.2 Guidance 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 TOE supports keyed hash functions in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

Keyed hash functions used for integrity testing are not configurable.

2.2.6.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support", Table 3: Validated Algorithm Implementations) identifies the CAVP certifications verifying keyed hashing, as follows.

Functions	Standards	Certificates
HMAC-SHA-1 (key/digest sizes 160 bits) HMAC-SHA-256 (key/digest sizes 256 bits) HMAC-SHA-384 (key/digest sizes 384 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2	A5164 A5175

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

2.2.7.1 TSS 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.

[ST] Section 6.2 states that RSA 2048-bit and 3072-bit are used for SSH and TLS as well as ECDSA with P-256, P-384, and P-521. This section also states that 2048-bit RSA is used for verification of software updates and firmware integrity self-testing.

2.2.7.2 Guidance 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 TOE supports digital signature functions in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e.

FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

Digital signature functions used for validation of updates are not configurable.

2.2.7.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support", Table 4: Validated Algorithm Implementations) identifies the CAVP certifications verifying digital signature services, as follows.

Functions	Standards	Certificates
RSA Digital Signature Algorithm (rDSA) (2048, 3072-bit modulus)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A5164 A5175
ECDSA (P-256, P-384, P-521)	ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"	A5164 A5175

2.2.8 HTTPS Protocol (FCS_HTTPS_EXT.1)

2.2.8.1 TSS Activities

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

[ST] Section 6.2 states that the TOE implements HTTPS for inbound connectivity from remote administrators. All HTTPS implementation is compliant with RFC 2818 and uses the TLS functionality as defined in FCS_TLSS_EXT.1. In all cases, an invalid certificate will cause the connection to be aborted.

2.2.8.2 Guidance 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.

The "Remote access requirements" section of [CCECG] states that enabling FIPS-CC mode is the sole configuration step needed for the HTTPS server (which is the only implementation of HTTPS in the TSF). The "Enabling FIPS-CC mode" section earlier in the document has the instructions for doing this.

2.2.8.3 Test 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.

2.2.9 Cryptographic Operation (Random Bit Generation) (FCS_RBG_EXT.1)

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

2.2.9.1 TSS Activities

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.

[ST] Section 6.2 identifies the ISO 18031 AES-CTR_DRBG with a 256-bit key. This section states that JitterEntropy is used as a hardware entropy source and is used to supply the DRBG with a 256 bit seed.

2.2.9.2 Guidance Activities

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

The TOE supports deterministic random bit generation in support of SSH and TLS communications. The "Administration Specific Changes" section of [CCECG] states that "The administrator does not need to take any specific actions to ensure compliance when using HTTPS or SSH as long as the FIPS-CC mode of operation has been enabled." This is stated in the context of remote access. Under "Log Specific Settings," [CCECG] says that "cryptographic operations used in securing connection to remote audit server, i.e. FortiAnalyzer, are not configurable." Based on this it is evident that no cryptographic configuration is required for any external interface (aside from enabling FIPS-CC mode as part of initial setup) in order for the TSF to be operating in its evaluated configuration with regards to cryptographic functions.

2.2.9.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support", Table 5: Validated Algorithm Implementations) identifies the CAVP certifications verifying random bit generation, as follows.

Functions	Standards	Certificates
CTR-DRBG (AES) – 256 bits entropy	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions	A5164 A5175

2.2.10 SSH Server Protocol (FCS_SSHS_EXT.1)

2.2.10.1 TSS Activities

Modified by TD0631

FCS_SSHS_EXT.1.2

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

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

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

[ST] section 6.2 identifies the RSA public key algorithm as being used for SSH client authentication.

FCS_SSHS_EXT.1.3

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

[ST] section 6.2 states that packets larger than 256KB are dropped.

FCS_SSHS_EXT.1.4

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

[ST] section 6.2 claims implementation of the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 4256, 5647, and 6668, with no optional characteristics. The supported encryption algorithms are aes128-cbc, aes256-cbc, aes128-gcm@openssh.com, and aes256-gcm@openssh.com, consistent with the SFR claims.

Modified by TD0631

FCS_SSHS_EXT.1.5

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

[ST] section 6.2 identifies rsa-sha2-512 and ecdsa-nistp-384 as the supported public key algorithms, consistent with the SFR claim made.

FCS_SSHS_EXT.1.5

The evaluator shall confirm that the TSS includes the description of how the TOE establishes a user identity when an SSH client presents a public key or X.509v3 certificate. For example, the TOE could

verify that the SSH client's presented public key matches one that is stored within the SSH server's authorized_keys file.

[ST] section 6.2 states that the TOE establishes a user identity by associating the presented public key and username with the associated values in its authorized_keys file.

FCS_SSHS_EXT.1.6
The evaluator shall check the TSS to ensure that it lists the supported data integrity algorithms, and that the list corresponds to the list in this component.

[ST] section 6.2 identifies the supported data integrity algorithms as HMAC-SHA-1 and HMAC-SHA-256, with GCM implicitly used as the MAC when a GCM encryption algorithm is negotiated, consistent with the claims made in the SFR.

FCS_SSHS_EXT.1.7
The evaluator shall check the TSS to ensure that it lists the supported key exchange algorithms, and that the list corresponds to the list in this component.

[ST] section 6.2 identifies the supported key exchange algorithm as diffie-hellman-group14-sha1, consistent with the claim made in the SFR.

FCS_SSHS_EXT.1.8
The evaluator shall check that the TSS specifies the following:

a) Both thresholds are checked by the TOE.
b) Rekeying is performed upon reaching the threshold that is hit first.

[ST] section 6.2 specifies that both time and data based thresholds are used for SSH rekeying, and that triggering either of them prompts a rekey to be performed.

2.2.10.2 Guidance Activities

FCS_SSHS_EXT.1.4
The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

The "Remote access requirements" section of [CCECG] states that SSH is configured solely through enabling FIPS-CC mode. The "Enabling FIPS-CC mode" section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_SSHS_EXT.1.5
The evaluator shall also check the guidance documentation to ensure that it contains instructions on configuring the TOE so that SSH conforms to the description in the TSS (for instance, the set of algorithms advertised by the TOE may have to be restricted to meet the requirements).

The "Remote access requirements" section of [CCECG] states that SSH is configured solely through enabling FIPS-CC mode. The "Enabling FIPS-CC mode" section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_SSHS_EXT.1.6

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

The "Remote access requirements" section of [CCECG] states that SSH is configured solely through enabling FIPS-CC mode. The "Enabling FIPS-CC mode" section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_SSHS_EXT.1.7

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

The "Remote access requirements" section of [CCECG] states that SSH is configured solely through enabling FIPS-CC mode. The "Enabling FIPS-CC mode" section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_SSHS_EXT.1.8

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

The SSH rekey thresholds are not directly configurable by a Security Administrator so this activity is not applicable.

2.2.10.3 Test Activities

FCS_SSHS_EXT.1.2

Modified by TD0631

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

The evaluator verified that all of the public-key authentication algorithms claimed in the ST were supported by the TOE.

FCS_SSHS_EXT.1.2

Modified by TD0631

Test 2: The evaluator shall choose one client public key authentication algorithm supported by the TOE. The evaluator shall generate a new client key pair for that supported algorithm without configuring the

TOE to recognize the associated public key for authentication. The evaluator shall use an SSH client to attempt to connect to the TOE with the new key pair and demonstrate that authentication fails.

The evaluator verified that the TOE rejected a connection attempt with an unrecognized public key.

FCS_SSHS_EXT.1.2

Modified by TD0631

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

The evaluator verified that it was possible to login to the TOE with a valid password.

FCS_SSHS_EXT.1.2

Modified by TD0631

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

The evaluator verified that the TOE would reject a connection attempt with an invalid password.

FCS_SSHS_EXT.1.3

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

The evaluator verified that the TOE drops packets that are larger than the value specified in [ST].

FCS_SSHS_EXT.1.4

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

The evaluator verified that a remote SSH client can successfully connect to the TOE using all symmetric encryption algorithms claimed in [ST] and that all attempts to use algorithms not claimed in [ST] resulted in unsuccessful attempts.

FCS_SSHS_EXT.1.5

Modified by TD0631

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

Has effectively been moved to FCS_SSHS_EXT.1.2.

As stated in TD0631 this test is covered by FCS_SSHS_EXT.1.2.

FCS_SSHS_EXT.1.5

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

The evaluator verified that the TOE rejects a connection with a non-supported host public key algorithm.

FCS_SSHS_EXT.1.6

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

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

The evaluator demonstrated the TOE’s ability to use both of the HMAC algorithms listed in the [ST].

FCS_SSHS_EXT.1.6

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

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

The evaluator demonstrated that the TOE would not accept a connection attempt with an HMAC-MD5 algorithm.

FCS_SSHS_EXT.1.7

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

The evaluator demonstrated that the TOE would reject a diffiehellman-group1-sha1 key exchange attempt.

FCS_SSHS_EXT.1.7

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

The evaluator demonstrated that the TOE could use the key exchange methods listed in the [ST].

FCS_SSHS_EXT.1.8

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

N/A

FCS_SSHS_EXT.1.8

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

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

The evaluator demonstrated that the TOE would rekey an SSH connection after one hour.

FCS_SSHS_EXT.1.8

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

The evaluator demonstrated that the TOE would rekey an SSH connection before 1 GB of data was transmitted.

FCS_SSHS_EXT.1.8

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

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

The evaluator demonstrated that the TOE would rekey an SSH connection before 1 GB of data was transmitted.

FCS_SSHS_EXT.1.8

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

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

N/A, rekey thresholds are not configurable.

FCS_SSHS_EXT.1.8

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

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

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

N/A

2.2.11 TLS Client Protocol without Mutual Authentication (FCS_TLSC_EXT.1)

2.2.11.1 TSS Activities

FCS_TLSC_EXT.1.1

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 listethd for this component.

[ST] Section 6.2 identifies the TLS client ciphersuites as being consistent with what is claimed in FCS_TLSC_EXT.1.1.

FCS_TLSC_EXT.1.2

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 (RFC 5952 for IPv6, RFC 3986 for IPv4) is enforced.

[ST] Section 6.2 states that the TOE validates reference identifiers in TLS server certificates in the manner specified by RFC 6125 section 6. Specifically, the client uses the FQDN (host name) in the CN or SAN as the reference identifier for the TLS server certificate. IP addresses and wildcards are supported. There is no override to accept an invalid certificate.

FCS_TLSC_EXT.1.4
The evaluator shall verify that TSS describes the Supported Elliptic Curves/Supported Groups Extension and whether the required behaviour is performed by default or may be configured.

[ST] Section 6.2 identifies secp256r1, secp384r1, and secp521r1 as the supported elliptic curves. This is the default behavior when the TOE is in its evaluated FIPS-CC mode configuration and is not configurable.

2.2.11.2 Guidance Activities

FCS_TLSC_EXT.1.1
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 "Remote access requirements" section of [CCECG] states that TLS for remote administration is configured solely through enabling FIPS-CC mode. This is also stated for the outbound TLS client interface used for audit log transfer under "Log Specific Settings." The "Enabling FIPS-CC mode" section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_TLSC_EXT.1.2
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 "Log Specific Settings" section of [CCECG] states that DNS name and IPv4/IPv6 address can be used to identify the remote audit server (which is the only remote entity to which the TLS client connects). It goes on to state that if an IP address is used, it must be present in the SAN extension, and that wildcards are not supported.

The TOE is not distributed so this portion of the evaluation activity is not applicable.

FCS_TLSC_EXT.1.4
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.

The ST does not indicate that the Supported Elliptic Curves/Supported Groups extension must be configured so this evaluation activity is not applicable to the TOE.

2.2.11.3 Test Activities

Removed by TD0670: ~~For all tests in this chapter the TLS server used for testing of the TOE shall be configured not to require mutual authentication~~.

FCS_TLSC_EXT.1.1

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).

The evaluator established a TLS connection with the TOE using all supported ciphersuites.

FCS_TLSC_EXT.1.1

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

The evaluator configured the server to present a certificate that did not have the Server Authentication purpose in the extendedKeyUsage field and verified the connection failed.

FCS_TLSC_EXT.1.1

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.

The evaluator attempted a connection using a EC certificate while using the expected ECDSA RSA ciphersuite and verified the connection failed.

FCS_TLSC_EXT.1.1

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.

The evaluator attempted to connect using unsupported and NULL ciphersuites and verified the connection failed. The evaluator also presented an unsupported curve and verified the connection failed.

FCS_TLSC_EXT.1.1

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

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

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.

The evaluator used a TLS tool to modify the TLS version on the server and attempted a connection, verifying the connection failed. The evaluator used the tool to modify the signature block in the Server's Key Exchange handshake message and verified the connection failed.

FCS_TLSC_EXT.1.1

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 evaluator used a TLS tool to modify a byte in the Server Finished handshake message and verified the connection failed. The evaluator also used the tool to send a garbled message from the server after the ChangeCipherSpec message and verified the connection failed. Lastly, the evaluator used the tool to modify a byte in the server's nonce in the Server Hello handshake message and verified the connection failed.

Note: Where applicable all of the following tests were performed with FQDN, IPv4 and IPv6 reference identifiers. This is consistent with the reference identifier types claimed in section 5.2.2.11 of the [ST].

FCS_TLSC_EXT.1.2

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.

N/A

FCS_TLSC_EXT.1.2

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:

FCS_TLSC_EXT.1.2

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.

The evaluator configured the server to present a certificate whose CN does not match the reference identifier and does not contain a SAN extension. The connection failed.

FCS_TLSC_EXT.1.2

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.

The evaluator configured the server to present a certificate whose CN matched the reference identifier but its SAN does not, and verified the connection failed.

FCS_TLSC_EXT.1.2

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.

The evaluator configured the server to present a certificate whose CN matched the reference identifier and has no SAN, and verified the connection still succeeded.

FCS_TLSC_EXT.1.2

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).

The evaluator configured the server to present a certificate whose CN did not match the reference identifier but had a SAN that did, and verified the connection succeeded.

FCS_TLSC_EXT.1.2

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.)

The evaluator verified that the TOE rejected a TLS connection when it received a server certificate with a wildcard that wasn't in the left-most label of the presented identifier. The TOE would accept a wildcard that was in the left-most label.

Modified per TD0790.

FCS_TLSC_EXT.1.2

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

Test 6: [conditional] If IP **address identifiers are** supported **in the SAN or CN,** the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with **a wildcard** asterisk (*) (e.g. CN=*****.168**.0.1** when connecting to 192.168**.0.1,** CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A. The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

The TOE rejects server certificates with wildcards for both IPv4 and IPv6 identifiers.

FCS_TLSC_EXT.1.2

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.)

N/A – RFC 5280 was not selected.

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

FCS_TLSC_EXT.1.3

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.

The evaluator loaded all necessary certificates to the TOE to complete the validation chain and successfully established a trusted channel with an external entity.

FCS_TLSC_EXT.1.3

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.

The evaluator configured the TOE to cover the presented scenarios and confirmed the connection failed for each.

FCS_TLSC_EXT.1.3

Test 3 [conditional]: The purpose of this test to verify that only selected certificate validation failures could be administratively overridden. If any override mechanism is defined for failed certificate

validation, the evaluator shall configure a new presented certificate that does not contain a valid entry in one of the mandatory fields or parameters (e.g. inappropriate value in extendedKeyUsage field) but is otherwise valid and signed by a trusted CA. The evaluator shall confirm that the certificate validation fails (i.e. certificate is rejected), and there is no administrative override available to accept such certificate.

N/A – The TOE does not support administrative overrides for certificate validation failures.

FCS_TLSC_EXT.1.4

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

The evaluator established successful connections with supported curves.

2.2.12 TLS Client Support for Mutual Authentication (FCS_TLSC_EXT.2)

2.2.12.1 TSS 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.

[ST] section 6.2 states that TLS mutual authentication is supported on the client side (i.e., the TOE can provide a TLS client certificate to a server).

2.2.12.2 Guidance 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 "Log Specific Settings" section of [CCECG] states that the default client certificate used to authenticate the TOE to the remote log server (which in the evaluated configuration is a separate Fortinet product) is present automatically and cannot be overridden.

2.2.12.3 Test Activities

Modified by TD0670

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.

The evaluator verified that the TOE successfully connected to a TLS server configured for mutual authentication. The TOE was seen to send both a client Certificate and Certificate Verify messages.

2.2.13 TLS Server Protocol without Mutual Authentication (FCS_TLSS_EXT.1)

2.2.13.1 TSS Evaluation Activity

FCS_TLSS_EXT.1.1
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.

[ST] Section 6.2 identifies the supported TLS ciphersuites consistent with the claims made in the SFR.

FCS_TLSS_EXT.1.2
The evaluator shall verify that the TSS contains a description of how the TOE technically prevents the use of old SSL and TLS versions.

[ST] Section 6.2 states that enabling FIPS-CC mode prevents old SSL/TLS versions from being used.

Modified per TD0635.
FCS_TLSS_EXT.1.3
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.

[ST] Section 6.2 states that secp256r1, secp384r1, and secp521r1 are the EC curves offered by the TOE's TLS server, and that 2048-bit MODP (Diffie-Hellman group 14) is the sole finite field group that is offered.

FCS_TLSS_EXT.1.4
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.

[ST] Section 6.2 states that the TOE supports session resumption using session tickets, and that AES CBC/GCM 128/256 bits is used to encrypt the session tickets.

2.2.13.2 Guidance Activities

FCS_TLSS_EXT.1.1
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 “Remote access requirements” section of [CCECG] states that TLS for remote access (which is the TSF’s sole TLS server implementation) is configured solely through enabling FIPS-CC mode. The “Enabling FIPS-CC mode” section of [CCECG] describes the steps for enabling FIPS-CC mode.

FCS_TLSS_EXT.1.2 and FCS_TLSS_EXT.1.3

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

The “Remote access requirements” section of [CCECG] states that TLS for remote access (which is the TSF’s sole TLS server implementation) is configured solely through enabling FIPS-CC mode. The “Enabling FIPS-CC mode” section of [CCECG] describes the steps for enabling FIPS-CC mode.

2.2.13.3 Test Activities

FCS_TLSS_EXT.1.1

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).

The evaluator used OpenSSL’s s_client functionality to establish TLS connections to the TOE using each of the ciphersuites claimed in section 5.2.2.11 of the [ST]. A wire capture of the test was also examined to verify that its output was accurate.

FCS_TLSS_EXT.1.1

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.

The evaluator used OpenSSL’s s_client functionality to attempt to open a connection to the TOE using a ciphersuite not claimed in the [ST] and verified that the TOE rejected the connection.

The evaluator also used a proprietary TLS test tool to attempt to open a connection using the TLS_NULL_WITH_NULL_NULL ciphersuite. The TOE was observed to reject the connection attempt.

FCS_TLSS_EXT.1.1

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 evaluator used a TLS tool to modify a byte in the Client Finished handshake message and verified the connection failed. The evaluator also examined a successful connection and confirmed the Finished message was encrypted.

FCS_TLSS_EXT.1.2

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

The evaluator used an open source TLS test tool to attempt a connection using unsupported TLS protocol versions SSL2.0, SSL3.0, TLS1.0, and TLS1.1. The TOE rejected all connection attempts.

FCS_TLSS_EXT.1.3

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.

The evaluator used OpenSSL s_client to make a successful connection with each of the supported elliptic curves claimed in the [ST]. Configuring the s_client to use an unsupported curve resulted in the TOE rejecting the connection attempt.

FCS_TLSS_EXT.1.3

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).

The evaluator verified that the TOE would open a TLS connection using the 2048-bit DHE parameter claimed in the [ST].

FCS_TLSS_EXT.1.3

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.

N/A – The TOE does not support RSA key establishment.

FCS_TLSS_EXT.1.4

Test Objective: To demonstrate that the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).

FCS_TLSS_EXT.1.4

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.

Added per TD0569.

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

N/A, the TOE supports session resumption based on session tickets according to RFC 5077.

FCS_TLSS_EXT.1.4

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.

Added per TD0569.

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

N/A – The TOE does not support session resumption based on session IDs.

Modified by TD0556 and TD0555.

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 an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the

client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

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.

Added per TD0569.

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

The evaluated or verified that the TOE responded to valid session tickets with an abbreviated handshake. The evaluator also verified that when the TOE received a modified session ticket it did not resume a previous TLS session but instead performed a full handshake.

2.3 Identification and Authentication (FIA)

2.3.1 Authentication Failure Management (FIA_AFL.1)

2.3.1.1 TSS 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.

[ST] Section 6.3 provides a description of the SSH and HTTPS remote interfaces. This section describes the ability of the administrator to be locked out after exceeding the configured number of failed authentication attempts, and that when a lockout occurs it persists for the configured time period.

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).

[ST] Section 6.3 states that lockout does not apply to the local administrative interface. For the virtual TOE, the local interface is launched from the ESXi console.

2.3.1.2 Guidance 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.

[CCECG] states under “Miscellaneous administration changes” that the default lockout behavior is to lock out a remote administrator account (doesn’t apply to the local console interface) for one hour after three failed password attempts.

The commands to control this behavior are documented under “system global” in [CLI], under admin-lockout-duration and admin-lockout-threshold. This is also documented in [ADMIN] under “General security tuning”.

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.

[CCECG] section “Miscellaneous administration changes” identifies that the lockout only applies to password-based authentication and only for remote administrators, so access is still preserved via local administration and through configuration of SSH public key authentication.

2.3.1.3 Test 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):

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.

The evaluator configured the number of unsuccessful connections allowed before locking the account, as well as the length of lockout. The evaluator then triggered the lockout by failing authentication attempts until the username was locked out.

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.

The evaluator followed guidance to administratively unlock the locked user account. The evaluator also waited the configured time and confirmed the username was unlocked.

2.3.2 Password Management (FIA_PMG_EXT.1)

2.3.2.1 TSS Evaluation Activity

Modified per TD0792

The evaluator shall check that the TSS lists the supported special character(s) for the composition of administrator passwords.

[ST] Section 6.3 identifies the supported character set for locally-defined administrator passwords.

Added per TD0792

The evaluator shall check the TSS to ensure that the minimum password length parameter is configurable by a Security Administrator.

[ST] section 6.3 states that the minimum password length is configurable.

Added per TD0792

The evaluator shall check that the TSS lists the range of values supported for the minimum password length parameter. The listed range shall include the value of 15.

[ST] section 6.3 states that the minimum password length can be configured to a value between 8 and 15.

2.3.2.2 Guidance Activities

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

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

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

Section “The FIPS-CC Mode of Operation” in [CCECG] identifies the minimum password length and composition requirements that are enforced in that mode. [CLI] documents the command for setting the minimum password length in the CLI under “system password-policy.” [ADMIN] documents configuring this behavior in the GUI under “Configuring system options.”

2.3.2.3 Test 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.

The evaluator generated a number of good users and passwords and confirmed they were accepted by the TOE.

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.

The evaluator attempted to create a number of bad users with poor passwords that did not pass complexity requirements, and verified those were rejected.

2.3.3 Protected Authentication Feedback (FIA_UAU.7)

2.3.3.1 TSS Evaluation Activity

None.

2.3.3.2 Guidance 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.

Via test observation the evaluator verified that authentication data is automatically obfuscated during local login and that no setting exists to modify this behavior; therefore, there is no expectation that this is discussed in the operational guidance.

2.3.3.3 Test Activities

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

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.

The evaluator logged in locally and verified authentication information was obscured.

2.3.4 Password-based Authentication Mechanism (FIA_UAU_EXT.2)

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

2.3.5 User Identification and Authentication (FIA_UIA_EXT.1)

2.3.5.1 TSS Evaluation Activity

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”.

[ST] section 6.3 describes the use of a local password-based authentication mechanism for all TOE interfaces (local CLI, SSH CLI, web GUI), and the use of a public key authentication mechanism for the SSH CLI. Successful logins are defined as supplying a valid credential for the claimed identity.

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.

[ST] Section 6.3 states that no actions are allowed prior to user authentication besides viewing of the warning banner.

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.

The TOE is not distributed and therefore this is not applicable.

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.

The TOE is not distributed and therefore this is not applicable.

2.3.5.2 Guidance 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 “Connecting to the web UI or CLI” section of [ADMIN] provides instructions for how to authenticate to the TOE using each management interface (remote GUI, remote CLI, local CLI).

The configuration of administrator passwords in the GUI is discussed under “Configuring administrator accounts” in [ADMIN]. The configuration of administrator passwords and SSH public keys in the CLI is referenced under “system admin” in [CLI].

2.3.5.3 Test 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:

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.

The evaluator attempted to log in using good credentials and successfully connected. The evaluator attempted to log in using bad credentials and was denied access. This test was performed with both username/password and SSH public key authentication.

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.

The evaluator examined the TOE for any services available to a user prior to remotely authenticating and found none.

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.

The evaluator examined the TOE for any services available to a user prior to locally authenticating and found the services were consistent.

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.

N/A – The TOE is not distributed.

2.3.6 X.509 Certificate Validation (FIA_X509_EXT.1/Rev)

2.3.6.1 TSS 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).

[ST] Section 6.3 states that all certificates that are presented to the TOE are validated. This includes certificates presented to the TOE's TLS client and certificates returned by a CA in response to a certificate signing request.

The use of certificates for trusted updates, self-testing, or any other function is not claimed by the TOE.

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.

[ST] Section 6.3 states that the TOE supports the use of either CRL or OCSP for revocation status checking of TLS certificates. In all cases, if the revocation status of a certificate cannot be determined, the TSF treats it is invalid. Both the leaf certificate and any intermediate CA certificates are checked for TLS.

2.3.6.2 Guidance 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.

CRL and OCSP checking in the context of remote log server connections (which is the only interface that validates X.509 certificates) is discussed under “Log Specific Settings” in [CCECG]. This section also describes how to configure the trust store to upload CA and intermediate certificates for validating the log server certificate. This same section identifies the extendedKeyUsage fields that are validated for certificates based on the intended use of the certificate.

The “Managing Certificates” section of [ADMIN] describes how to configure the trust store in the GUI. The “system certificate local” and “system certificate remote” sections of [CLI] describe how to configure the trust store in the CLI. The “system global” section of [CLI] describes how to specify which certificate the TOE uses by default when multiple certificates are available for its use.

2.3.6.3 Test 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/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:

Note: As stated in section 5.2.3.7 of the [ST] the TOE only uses x.509 certificates to support authentication for TLS/HTTPS connections. Because of this all of the following tests were run to demonstrated the TOE’s use of certificates for those connection types.

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).

The evaluator uploaded all necessary CA files to complete the validation chain and successfully connected.

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.

The evaluator removed the intermediate CA, thus breaking the validation chain. Connections using that validation chain failed.

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

The evaluator attempted to connect using an expired certificate and verified the connection failed.

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.

The evaluator revoked the server cert and attempted to connect. The connection failed due to the cert being revoked. The evaluator repeated the test using a revoked intermediate certificate, and this failed as well. These revocation tests were performed with both CRLs and OCSP.

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.

The evaluator uploaded a CRL signed by a CA that did not have CRLsign key usage and attempted to connect using this CRL. The connection failed.

The evaluator then used an OCSP responder whose certificate lacked the OCSP signing purpose. The TOE again rejected the connection attempt after receiving the responder's certificate.

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.)

The evaluator used a TLS tool to modify a byte in the first eight bytes of the certificate and started a connection. The connection failed.

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

The evaluator used a TLS tool to modify a byte in the signatureValue field of the certificate and started a connection. The connection failed.

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.)

The evaluator used a TLS tool to modify a byte in the certificate's public key and started a connection. The connection failed.

Test added in accordance with TD0527.

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 evaluation 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.

N/A – The TOE has certificates loaded into its certificate store, not presented in certificate messages.

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.

N/A – The TOE has certificates loaded into its certificate store, not presented in certificate messages.

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.

The evaluator verified that the TOE will not import an intermediate CA certificate that has explicitly defined EC parameters.

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).

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).

The evaluator verified that the TOE will not install a certificate without the basicConstraints extension into its trust store.

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 verified that the TOE will not install a certificate with the basicConstraints extension set to FALSE into its trust store.

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).

X.509 certificates are not used for trusted updates and executable code integrity verification, therefore there are no other distinct uses of certificates.

2.3.7 X.509 Certificate Authentication (FIA_X509_EXT.2)

2.3.7.1 TSS 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.

[ST] Section 6.3 states that all certificates that are presented to the TOE are validated. The TOE has separate TLS client and server certificates and chooses which to use based on its configuration. In all other situations, the TOE validates the certificate that is presented to it.

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.

[ST] Section 6.3 states that if the revocation status of a certificate cannot be determined, the TSF treats it is invalid. No claim is made for the default action being administratively configurable.

2.3.7.2 Guidance 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 “Log Specific Settings” section of [CCECG] describes the configuration for X.509 certificates, including how to configure the operational environment so that the TOE can connect to a log server with a valid certificate (the log server interface is the only interface where X.509 certificate validation is used).

FIA_X509_EXT.2.2 in [ST] claims that certificates are automatically rejected if their validity status cannot be determined, so there is no expectation that the guidance document would have steps to follow for this behavior.

2.3.7.3 Test 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.

The evaluator shut down the CRL distribution server and started a connection with the TOE. The connection failed because the TOE was unable to communicate with the distribution server, and thus unable to verify any connections.

The evaluator then attempted a connection with an OCSP responder that did not have an entry for the certificate the TOE was trying to verify. The TOE rejected the connection attempt.

2.3.8 X.509 Certificate Requests (FIA_X509_EXT.3)

2.3.8.1 TSS 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.

This activity is not applicable since the ST author does not select "device-specific information".

2.3.8.2 Guidance Activities

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

The ”Generating a certificate signing request” section of [ADMIN] describes how to generate a certificate signing request in the GUI and it includes instructions for setting the Common Name, Organization, Organization Unit, and Country fields. The “certificate” section of [CLI] describes how to generate a certificate signing request in the CLI.

2.3.8.3 Test Activities

The evaluator shall perform the following tests:

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.

The evaluator generated a CSR following user guidance, captured the generated request and ensured that it conforms to the format specified and provides the public key and other required information.

Test 2: The evaluator shall demonstrate that validating a response message to a Certification Request without a valid certification path results in the function failing. The evaluator shall then load a certificate or certificates as trusted CAs needed to validate the certificate response message and demonstrate that the function succeeds.

The evaluator attempted to upload a signed cert from the previous CSR, but without a full valid certification path. The attempted failed. The evaluator uploaded the necessary CAs to complete the certification path and retried uploaded, which succeeded.

2.4 Security Management (FMT)

2.4.1 General requirements for distributed TOEs

2.4.1.1 TSS 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.

The TOE is not distributed and therefore this is not applicable.

2.4.1.2 Guidance Activities

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

The TOE is not distributed and therefore this is not applicable.

2.4.1.3 Tests Activities

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

The TOE is not distributed and therefore this is not applicable.

2.4.2 Management of Security Functions Behavior (FMT_MOF.1/Functions)

2.4.2.1 TSS Activities

For distributed TOEs see chapter 2.4.1.1.

The TOE is not distributed and therefore this is not applicable.

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).

[ST] Section 6.4 states that the Security Administrator is able to determine and modify the behavior of the TOE's connectivity to an external IT entity for transmission of audit data. This can be done on both local and remote interfaces.

2.4.2.2 Guidance 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 "Log Specific Settings" section of [CCECG] describes how to configure the transmission of audit data to an external IT entity. The selections for handling of audit data and audit functionality when local audit storage space is full are not chosen so these parts of the evaluation activity are not applicable.

2.4.2.3 Test 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.

Without prior authentication as Security Administrator, evaluator was unable to make configuration changes regarding transmission of audit data to an external IT entity.

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

With prior authentication as Security Administrator, the evaluator was able to make configuration changes.

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 SFR s FAU_STG_EXT.1.2, FAU_STG_EXT.1.3 and FAU_STG_EXT.2/LocSpace.

N/A – This option was not selected.

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.

N/A – This option was not selected.

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.

N/A – This option was not selected.

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.

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

[N/A – This option was not selected.

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

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.

Without authentication as Security Administrator, the evaluator was unable to perform any tasks to determine the behavior of selected functions.

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.

With authentication as Security Administrator, the evaluator was able to perform tasks such as configure syslog to an appointed syslog server.

2.4.3 Management of Security Functions Behavior (FMT_MOF.1/ManualUpdate)

2.4.3.1 TSS Activities

For distributed TOEs see section 2.4.1.1. There are no specific requirements for non-distributed TOEs.

The TOE is not distributed.

2.4.3.2 Guidance 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).

The sections “Downloading the FIPS-CC certified firmware and MD5 check sums,” “Verifying the integrity of the firmware build,” and “Installing the FIPS-CC firmware build” of [CCECG] provide the steps to download and install product updates. The instructions are specifically for initial setup but are generally applicable to subsequent updates as well. [CLI] describes how to apply an update using the CLI under “restore image” and [ADMIN] describes how to apply an update using the GUI under “Installing firmware.” The section states that the digital signature of the image is validated and provides an example GUI screenshot and event log message when the signature is missing or invalid.

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 TOE is not distributed and therefore this is not applicable.

2.4.3.3 Test 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 shall fail.

Without administrator privileges, the evaluator was unable to perform any form of updates. The Update option does not exist in the interface for non-administrative users.

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.

This test case was covered by the tests for FPT_TUD_EXT.1.

2.4.4 Management of TSF Data (FMT_MTD.1/CoreData)

2.4.4.1 TSS 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.

[ST] Section 6.4 identifies the administrative functions and whether the function is accessed from the local and/or the remote administrative interface(s). Both interfaces and therefore all of the administrative functions require administrator authentication prior to access. The guidance documentation does not identify any admin functions as available prior to authentication.

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.

[ST] Section 6.4 states that the management of the TOE's trust store is performed by a Security Administrator, which for the purposes of the TOE is the 'Admin' user on the CLI or GUI.

2.4.4.2 Guidance 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.

The evaluator reviewed the guidance documentation and determined that each of the TSF-data-manipulating functions implemented in response to the requirements of the cPP is identified.

- Ability to configure the access banner;
  - CLI: "system global" in [CLI] (set pre-login-banner)
  - GUI: "configuring system options" in [ADMIN] (Login Disclaimer Setting)
- Ability to configure the session inactivity time before session termination or locking;
  - CLI: "system global" in [CLI] (set admin-idle-timeout)
  - GUI: "configuring system options" in [ADMIN] (Idle timeout)
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
  - CLI: "execute restore image" in [CLI]
  - GUI: [ADMIN]
- [CLI] describes how to apply an update using the CLI under "restore image" and [ADMIN] describes how to apply an update using the GUI under "Installing firmware."
- Ability to configure the authentication failure parameters for FIA_AFL.1;

- CLI: “system global” in [CLI] (set admin-lockout-duration and set-admin-lockout-threshold)
- GUI: N/A, see [ST] section 6.4.

- Ability to modify the behaviour of the transmission of audit data to an external IT entity;
  - CLI: “log setting remote” in [CLI] (various settings, see “Log Specific Settings” in [CCECG] for those specifically relevant to the evaluated configuration)
  - GUI: “Installing firmware” in [ADMIN]
- Ability to manage the cryptographic keys;
  - CLI: “system certificate local” in [CLI] (import/delete the TOE’s own certificate), “execute certificate local” in [CLI] (generate CSR), “system global” in [CLI] (set default-certificate to specify which certificate to use if multiple are available)
  - GUI: “Managing certificates” in [ADMIN]
- Ability to manage the trusted public keys database
  - CLI: “system admin” in [CLI] (set sshkey)
  - GUI: N/A, see [ST] section 6.4.
- Ability to manage the TOE’s trust store and designate X.509v3 certificates as trust anchors;
  - CLI: “system certificate ca” in [CLI]
  - GUI: “Managing certificates” in [ADMIN]
- Ability to import X.509v3 certificates to the TOE’s trust store;
  - CLI: “system certificate ca” in [CLI]
  - GUI: “Managing certificates” in [ADMIN]
- Ability to set the time which is used for time-stamps.
  - CLI: “system time manual” in [CLI]
  - GUI: “Configuring system time, options, and other system options” in [ADMIN]

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 as a trust anchor.

The ”Managing certificates” section of [ADMIN] provides instructions for the administrator on how to configure the trust store in the GUI, including loading CA and intermediate certificates. The “system certificate ca” command guidance in [CLI] also provides this information for the CLI. [CCECG] provides additional guidance under “Log Specific Settings” to recognize the log server’s certificate as valid.

2.4.4.3 Test 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.

2.4.5 Management of TSF Data (FMT_MTD.1/CryptoKeys)

2.4.5.1 TSS Activities

For distributed TOEs see chapter 2.4.1.1.

The TOE is not distributed and therefore this is not applicable.

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 those operations are performed.

[ST] Section 6.4 states that the Security Administrator has the ability to manage cryptographic keys by manipulating the contents of the trust store for certificates, generating certificate signing requests, and managing the SSH trusted key database. Management of the SSH key database and configuration of authentication failure parameters can only be done using the CLI while all other functions can be performed using the GUI.

2.4.5.2 Guidance 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.

The ”Generating a certificate signing request” section of [ADMIN] describes how to generate a certificate signing request in the GUI and it includes instructions for setting the Common Name, Organization, Organization Unit, and Country fields. The “certificate” section of [CLI] describes how to generate a certificate signing request in the CLI.

2.4.5.3 Test 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.

Without prior authentication as Security Administrator, the evaluator was unable to modify or manage any cryptographic keys.

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

With prior authentication as Security Administrator, the evaluator was able to modify cryptographic keys.

2.4.6 Specification of Management Functions (FMT_SMF.1)

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

2.4.6.1 TSS Activities (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).

[ST] Section 6.4 identifies the administrative functions provided by the TOE and whether the function is accessed from the local and/or the remote administrative interface(s). For those functions that are available remotely, this section further notes whether they can be performed at both the CLI and GUI, or only at the CLI.

The evaluator examined the TSS, Guidance Documentation and the TOE as observed during all other testing and confirmed that the management functions specified in FMT_SMF.1 are provided by the TOE.

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.

[ST] Section 6.4 identifies the local management interface as a direct console interface to the CLI, either via physical port or through the ESXi console for the virtual TOE.

[CCECG] describes configuration and use of the local interface under “Miscellaneous administration related changes,” the use of a local interface is inherent via connection to the TOE hardware or access through VM hypervisor based on whether the TOE is a physical or virtual appliance.

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 behavior observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

The TOE is not distributed and therefore this is not applicable.

2.4.6.2 Guidance Activities

See section 2.4.6.1 in this AAR.

2.4.6.3 Test Activities

The evaluator tests management functions as part of testing the SFRs identified in section 2.4.6. 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.

The evaluator was able to exercise all management functions.

2.4.7 Restrictions on Security Roles (FMT_SMR.2)

2.4.7.1 TSS 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.

[ST] Section 6.4 states that the TOE supports a single fully-privileged 'Admin' role that functions as the Security Administrator for the TOE.

2.4.7.2 Guidance 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 "Installing the CC Certified Firmware" section of [CCECG] includes initial steps for secure acceptance of the TOE prior to any use. [ADMIN] includes a section called "Setting up FortiMail system" that includes initial steps for setting up the web UI, initial steps for setting up the local CLI, and enabling remote access to the CLI via SSH.

2.4.7.3 Test 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.

In the course of performing the testing activities for the evaluation, the evaluator used both supported management interfaces: local console and remote Web GUI.

2.5 Protection of the TSF (FPT)

2.5.1 Protection of Administrator Passwords (FPT_APW_EXT.1)

2.5.1.1 TSS 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.

[ST] Section 6.5 states that the TSF stores password data in SHA-256 hashes and that no interface exists to view the hashed password data.

2.5.1.2 Guidance Activities

None defined.

2.5.1.3 Test Activities

None defined.

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

2.5.2.1 TSS 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.

[ST] Section 6.2 indicates that SSH private key and certificate private keys are stored in plaintext but in places that no administrator interface can access.

2.5.2.2 Guidance Activities

None defined.

2.5.2.3 Test Activities

None defined.

2.5.3 Reliable Time Stamps (FPT_STM_EXT.1)

2.5.3.1 TSS 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.

[ST] Section 6.5 states that the TSF uses time data for audit log timestamps, timed administrator lockouts, idle session timeouts, and validation of X.509 certificates. The TOE has an internal real-time clock that is administratively configurable.

Added per TD0632.
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.

[ST] does not select "obtain time from the underlying virtualization system."

2.5.3.2 Guidance 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.

[ADMIN] describes how to configure the time using the web GUI under "Configuring system time, options, and other system options." [CLI] identifies the "system time manual" as the CLI command used to change the system clock. The TOE does not use an NTP server.

Added per TD0632.
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.

The TSF does not obtain time from an underlying virtualization system so this is not applicable.

2.5.3.3 Test Activities

The evaluator shall perform the following tests:

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.

The evaluator was able to manually set the time and observed that the time was set correctly.

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.

N/A – The TOE is unable to support NTP servers.

Added per TD0632.

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.

N/A. Time is set on the TOE and it does not receive time from an underlying VS.

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.

N/A. The audit component of the TOE does not consist of several parts with independent time information, and therefore this is not applicable.

2.5.4 TSF Testing (FPT_TST_EXT.1)

2.5.4.1 TSS 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.

[ST] Section 6.5 states that to verify its own integrity, the TOE performs integrity tests on its firmware and configuration file at power on. The TOE also performs power on self-testing for its cryptographic module, which includes known answer tests for its cryptographic algorithms and NIST SP800-90A RNG health tests.

This section also argues that the self-testing is sufficient because the self-tests would detect any modification to the executable TOE or its configuration, or any degradation in hardware functionality that could affect the behavior of its cryptographic functions.

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

The TOE is not distributed and therefore this is not applicable.

2.5.4.2 Guidance 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.

The execution of the self-tests is described in [CCECG] under “Self-tests.” The “FIPS Error Mode” section identifies what happens if a self-test fails and possible recovery steps if a failure is encountered. The summary of the self-tests identified in [CCECG] is consistent with the self-tests listed in the ST.

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.

This is not applicable since the TOE is not distributed.

2.5.4.3 Test 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 fulfil 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.

The evaluator rebooted the TOE and examined system logs after startup. These logs showed that the TOE had run and passed the self-tests claimed in the [ST]. The evaluator also verified the firmware version and confirmed the TOE's FIPS-CC status was enabled.

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.

N/A – The TOE is not distributed.

2.5.5 Trusted Update (FPT_TUD_EXT.1)

2.5.5.1 TSS 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.

[ST] Section 6.5 states that the TOE supports software/firmware updates, and that only one version of the TOE software/firmware is loaded at any one time. The version can be checked by an administrator via the CLI and GUI per this same section.

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.

[ST] Section 6.5 indicates that the TOE has a manual update mechanism. Updates are acquired from the vendor's site and copied to the TOE from the administrator's workstation. When an update is uploaded to the TOE, the digital signature is automatically validated against a hardcoded public key stored in the TOE's firmware. An attempted update using a package with a missing or mismatched public key is automatically aborted.

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.

The options 'support automatic checking for updates' and 'support automatic updates' are not chosen from the selection in FPT_TUD_EXT.1.2 and therefore this activity is not applicable.

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.

The TOE is not distributed and therefore this is not applicable.

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.

A published hash is not used to protect the integrity of updates so this evaluation activity is not applicable.

2.5.5.2 Guidance 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 "Installing the FIPS-CC firmware build" section of [CCECG] identifies the "get system status" CLI command as being used to query the active version. The TOE can only have one software version on disk at any given time so there is no case for installation with a delayed activation.

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.

[CCECG] section “Verifying the integrity of the firmware build” Section 5.4 describes how verification of the authenticity of the update is performed using a digital signature. No separate signature validation is done; it is performed automatically as part of the update process. The same section of [CCECG] provides example user feedback and log data for a failed verification. Sample logs for successful and failed updates are included in “NDcPP Common Criteria Logging Addendum.”

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.

The TOE does not claim a published hash for update integrity verification and therefore this test is not applicable.

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.

The TOE is not distributed and therefore this is not applicable.

If this was information 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.

The TOE is not distributed and therefore this is not applicable.

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 TOE does not use a certificate-based mechanism. Therefore this activity is not applicable.

2.5.5.3 Test Activities

The evaluator shall perform the following tests:

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.

The evaluator verified the current version of the product; confirmed ability to install an update of UAG components; and verified the version correctly corresponds to that of the update and to the current version of the product.

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.

The evaluator verified that the TOE will not accept an update with a missing or invalid digital signature.

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.

N/A, the TOE does not perform verification of updates using a published hash.

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).

N/A, the TOE does not perform verification of updates using a published hash.

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

N/A – The TOE is not distributed.

2.6 TOE Access (FTA)

2.6.1 TSF-initiated Termination (FTA_SSL.3)

2.6.1.1 TSS Activities

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

[ST] Section 6.6 states that each remote administrative interface has a separately configurable max idle session timeout between 1 and 480 minutes.

2.6.1.2 Guidance Activities

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

Instructions for configuring the inactivity time period are included under “system global” in [CLI] (set admin-idle-timeout) and under “configuring system options” in [ADMIN], for the CLI and GUI respectively. There is no difference between the local and remote CLI for this.

2.6.1.3 Test Activities

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

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.

The evaluator used the guidance documentation to configure 2 and 5 minute inactivity time periods and verified that the user was logged out after exceeding those time limits.

2.6.2 User-initiated Termination (FTA_SSL.4)

2.6.2.1 TSS Activities

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

[ST] Section 6.6 states that the TOE includes mechanisms for each interface for the Security Administrator on each to voluntarily terminate their own session.

2.6.2.2 Guidance Activities

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

The “Logging out from the CLI console” section of [CLI] says that a CLI session is terminated with the ‘exit’ command regardless of whether it is a local or remote session. The “Miscellaneous administration related changes” section of [CCECG} identifies how to terminate a GUI session.

2.6.2.3 Test Activities

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

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.

The evaluator was able to log in and log off in a local session.

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.

The evaluator was able to log in and log off in a remote session.

2.6.3 TSF-initiated Session Locking (FTA_SSL_EXT.1)

2.6.3.1 TSS 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.

[ST] Section 6.6 states that local administration session termination is supported and that it is configurable to the same range of values as remote interfaces.

2.6.3.2 Guidance 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.

Instructions for configuring the inactivity time period are included under “system global” in [CLI] (set admin-idle-timeout) and under “configuring system options” in [ADMIN], for the CLI and GUI respectively. There is no difference between the local and remote CLI for this.

2.6.3.3 Test Activities

The evaluator shall perform the following test.

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

The evaluator used the guidance documentation to configure 2 and 5 minute inactivity time periods and verified that the locally logged in user was logged out after exceeding those time limits.

2.6.4 Default TOE Access Banners (FTA_TAB.1)

2.6.4.1 TSS Evaluation Activity

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).

[ST] Section 6.6 states that all local and remote administrative interfaces display a configurable pre-authentication warning banner (a single banner configurable for all interfaces). This section also describes how to configure the banner text on both the CLI and GUI.

2.6.4.2 Guidance Activities

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

Instructions for configuring the login banner are included under “system global” in [CLI] (set pre-login-banner) and under “configuring system options” in [ADMIN], for the CLI and GUI respectively. There is no difference between the local and remote CLI for this.

2.6.4.3 Test Activities

The evaluator shall also perform the following test:

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.

The evaluator was able to set notice and warning messages and verified they appeared for local and remote access.

2.7 Trusted Path/Channels (FTP)

2.7.1 Inter-TSF Trusted Channel (FTP_ITC.1)

2.7.1.1 TSS 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.

[ST] Section 6.7 identifies the only TSF trusted channel as the TLS client interface to a remote audit server.

2.7.1.2 Guidance 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.

The only external interface that FTP_ITC.1 applies to is the remote log server interface. Per the “Remote access requirements” section of [CCECG] there is no protocol configuration beyond enabling FIPS-CC mode. The “Configure the FortiAnalyzer connection on the FortiMail unit” section of [CCECG] identifies that reconnection attempts will be made automatically in the event of an unintentionally broken connection. Configuration of the interface with regards to establishing one or more log servers as a trusted endpoint and recipient of log data is addressed by the “Log Specific Settings” section of [CCECG].

2.7.1.3 Test 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:

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.

The evaluator verified connections with a remote audit server was tested and successful.

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.

Evidence for this test can be seen in FCS_TLSC_EXT.1.1 Test 1. That test activity demonstrates the ability of the TOE to initiate a secure connection to an external syslog server.

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

Evidence for this can be seen in the FCS_TLSC_EXT.1 testing. The evaluator verified connections not sent in plaintext.

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.

The evaluator physically disrupted connections with the TOE and verified restored connections were protected.

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.

N/A – The TOE is not distributed.

2.7.2 Trusted Path (FTP_TRP.1/Admin)

2.7.2.1 TSS 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.

[ST] Section 6.7 identifies both SSH (for remote CLI) and TLS/HTTPS (for remote GUI) as trusted paths for remote authentication.

2.7.2.2 Guidance Activities

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

The “Connecting to the web UI or CLI” section of [ADMIN] describes how to connect to the TOE remotely using both HTTPS (for the GUI) and SSH (for the CLI).

2.7.2.3 Test Activities

The evaluated shall perform the following tests.

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.

The evaluator verified that all communications successfully connected.

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

The evaluator verified that all communications were 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.

N/A – The TOE is not distributed.

3 Security Assurance Requirements

3.1 Class ADV: Development

3.1.1 ADV_FSP.1 Basic Functional Specification

The EAs for this assurance component focus on understanding the interfaces (e.g., application programming interfaces, command line interfaces, graphical user interfaces, network interfaces) described in the AGD documentation, and possibly identified in the TOE Summary Specification (TSS) in response to the SFRs. Specific evaluator actions to be performed against this documentation are identified (where relevant) for each SFR in Section 2, and in EAs for AGD, ATE and AVA SARs in other parts of Section 3.

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.

3.1.1.1 ADV_FSP.1 Evaluation Activity

The evaluator shall examine the interface documentation to ensure it describes the purpose and method of use for each TSFI that is identified as being security relevant.

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.

Through review of [ST] and [CCECG], the evaluation team identified that the following external interfaces are security relevant:

- TLS Web GUI
- SSH CLI
- Local CLI

- Syslog interface.

The evaluation team determined the interface documentation described the purpose and method of use for each TSFI identified as being security relevant, sufficient to enable each of the evaluation activities to be completed satisfactorily. The evaluation team's results from performing the evaluation activities are documented in Section 2 of this AAR.

3.1.1.2 ADV_FSP.1 Evaluation Activity

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 evaluation team determined the interface documentation identified and described the parameters for each TSFI identified as being security relevant, sufficient to enable each of the evaluation activities to be completed satisfactorily. The evaluation team's results from performing the evaluation activities are documented in Section 2 of this AAR.

3.1.1.3 ADV_FSP.1 Evaluation Activity

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'.

The evaluation team examined the interface documentation and was able to map interfaces to SFRs, sufficient to enable each of the evaluation activities to be completed satisfactorily. The evaluation team's results from performing the evaluation activities are documented in Section 2 of this AAR.

3.2 Class AGD: Guidance Documents

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

3.2.1 AGD_OPE.1 Operational User Guidance

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.

3.2.1.1 AGD_OPE.1 Evaluation Activity

The evaluator shall ensure the Operational guidance documentation is distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware f the existence and role of the documentation in establishing and maintaining the evaluated configuration.

The [CCECG] and [ADMIN] will be published with the Security Target at the https://www.niap-ccevs.org/ website. The distribution of the documentation shall provide a reasonable guarantee that administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

[ADMIN] is published at Fortinet's documentation web site (https://docs.fortinet.com/). The [CCECG] will also be published upon the successful completion of Common Criteria evaluation.

3.2.1.2 AGD_OPE.1 Evaluation Activity

The evaluator shall ensure that the Operational guidance is provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

The Certified Models section of [CCECG] identifies the platform claimed for the TOE as identified in [ST]; the specific conditions that are expected to be met by the operational environment and/or administrators; and the components in the operational environment.

3.2.1.3 AGD_OPE.1 Evaluation Activity

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

The section labeled Installing the CC Certified Firmware starting on page 5 of [CCECG] describes how to install the TOE's FIPS-CC firmware. This is the only configuration required in order to get the TOE to use the correct cryptographic engine for the evaluated configuration.

3.2.1.4 AGD_OPE.1 Evaluation Activity

The evaluator shall ensure the Operational guidance makes it clear to an administrator which security functionality and interfaces have been assessed and tested by the EAs.

The [CCECG] Introduction section makes it explicitly clear that only the functionality claimed in [ST] was evaluated. Any other functionality that does not relate to the security functional claims are considered to

be non-interfering with respect to security. Any product security functionality that is within the scope of the evaluation must be configured in the manner specified in this guidance.

3.2.1.5 AGD_OPE.1 Evaluation Activity

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

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

b) **Updated according to TD0536**

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:

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.

Part a) is addressed by section 3.2.1.3 above.

Part b) is addressed in section 2.5.5.2 above.

Part c) is addressed by section 3.2.1.4 above.

3.2.2 AGD_PRE.1 Preparative Procedures

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

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

In addition, the evaluator performs the EAs specified below.

3.2.2.1 AGD_PRE.1 Evaluation Activity

The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).

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).

Page 5 of [CCECG] identifies the assumptions that state the specific conditions that are expected to be met by the operational environment and/or administrators. The Installation Requirements section describes the scope of evaluation and operational environment that the system must be in to ensure a secure deployment. This section is written in an informal style and provide sufficient detail and explanation that they can be understood and used by the target audience.

3.2.2.2 AGD_PRE.1 Evaluation Activity

The evaluator shall examine the Preparative procedures to ensure they are provided for every Operational Environment that the product supports as claimed in the Security Target and shall adequately address all platforms claimed for the TOE in the Security Target.

Section 2.3 of [CCECG] identifies two TOE platforms consistent with the security target. The instructions and guidance contained in the [CCECG] are applicable to these TOE platforms.

3.2.2.3 AGD_PRE.1 Evaluation Activity

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

The evaluators reviewed [CCECG] and [ADMIN] and determined that they describe how set up the TOE's logical interfaces for initial use and to configure the external interfaces specified as the TOE's Operational Environment by [ST]. The instructions and guidance contained in the [CCECG] is applicable to the TOE platform and following them results in a successful installation.

3.2.2.4 AGD_PRE.1 Evaluation Activity

The evaluator shall examine the preparative procedures to ensure they include instructions to manage the security of the TSF as a product and as a component of the larger operational environment.

[CCECG] provides instructions for managing the security of the TOE both as a product and as a component of the larger operational environment.

3.2.2.5 AGD_PRE.1 Evaluation Activity

In addition, the evaluator shall ensure that the following requirements are also met. The preparative procedures must a) include instructions to provide a protected administrative capability; and b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

[CCECG] page 9 includes instructions to provide a protected administrative capability. This section provides information for setting an administrative password.

3.3 Class ALC: Life-Cycle Support

3.3.1 ALC_CMC.1 Labelling of the TOE

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.

These CEM work units have been completed and documented in the ETR.

3.3.2 ALC_CMS.1 TOE CM Coverage

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

These CEM work units have been completed and documented in the ETR.

3.4 Class ASE: Security Targeted Evaluation

General ASE

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

3.4.1 ASE_TSS.1 TOE Summary Specification for Distributed TOEs

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

Note that additional Evaluation Activities for the TSS in the case of a distributed TOE are defined in section A.9.1.1 in the SD.

The TOE is not a distributed TOE. Therefore, this activity is not applicable.

3.5 Class ATE: Tests

3.5.1 ATE_IND.1 Independent Testing – Conformance

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.

The evaluator should consult Appendix A [in the SD] when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

Testing of the TOE was performed at the Leidos Accredited Testing and Evaluation Lab located in Columbia, Maryland from September 2023 to May 2024.

The evaluation team established a test configuration including the following TOE instances:

- FML 900F, Version 7.4.3
- FML VM, Version 7.4.3

Each relevant Test Activity identified in [SD_ND] was given its own test case in the Test Report. Each test case consists of the test steps specified in the Supporting Documents, along with the actual test steps performed by the evaluators and any corresponding evidence.

3.5.1.1 Tests Evaluation Activity

Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section A.9.3 in the SD.

The TOE is not a distributed TOE, so these additional activities are not applicable.

3.6 Class AVA: Vulnerability Assessment

3.6.1 AVA_VAN.1 Vulnerability Survey

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. Table 2: Mapping of AVA_VAN.1 CEM Work Units to Evaluation Activities, in the SD, 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 in the SD, while an “outline” of the assurance activity is provided below.

3.6.1.1 AVA_VAN.1 Evaluation Activity (Documentation)

In addition to the activities specified by the CEM in accordance with Table 2 in the SD, 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.

Modified by TD0547:

The developer shall provide documentation identifying the list of software and hardware components[1] 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), for example a web server, protocol or cryptographic libraries (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version

[1] In this sub-section the term “components” refers to parts that make up the TOE. It is therefore distinguished from the term “distributed TOE components”, which refers to the parts of a TOE that are present in one physical part of a distributed TOE. Each distributed TOE component will therefore generally include a number of the hardware and software components that are referred to in this sub-section: for example, each distributed TOE component will generally include hardware components such as processors and software components such as an operating system and libraries.

number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

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

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

b) a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, 6.3.3]

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

The vendor provided information on the software components of the TOE. Specifically, [ST] identifies the name of the TOE, the OS (Linux 5.10.180), and the OpenSSL 1.1.1w cryptographic implementation included in the TOE. The vendor separately provided proprietary material on the specific third-party libraries used by the TOE, including version information. The hardware components of the TOE are implicit in the model name, but the evaluators also considered the processors used in these TOE models as potential threat vectors.

The TOE is not distributed and therefore the last parts of the activity are not applicable.

3.6.1.2 AVA_VAN.1 Evaluation Activity

The evaluator formulates hypotheses in accordance with process defined in Appendix A in the SD. The evaluator documents the flaw hypotheses generated for the TOE in the report in accordance with the guidelines in Appendix A.3 in the SD. The evaluator shall perform vulnerability analysis in accordance with Appendix A.2 in the SD. The results of the analysis shall be documented in the report according to Appendix A.3 in the SD.

The evaluation team performed a search of the following public vulnerability databases:

- National Vulnerability Database (https://nvd.nist.gov/)
- US-CERT (https://www.kb.cert.org/vuls/html/search)
- Fortinet's Product Security Incident Response Team (PSIRT) (https://www.fortiguard.com/psirt)

Searches were performed several times, most recently June 25, 2024, using search terms that referenced the TOE itself, the processors that the physical TOE models use, the OS kernel version, the cryptographic library, and the list of additional third-party software components provided by the vendor.

No vulnerabilities were identified for the TOE.

The evaluation team determined that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential. Results are detailed in the proprietary vulnerability assessment report.