Assurance Activities Report

for

One Identity Safeguard for Privileged Sessions 6.9

Version 1.0

28 February 2022

Prepared by:

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

Prepared for:

National Information Assurance Partnership
Common Criteria Evaluation and Validation Scheme

The Developer of the TOE:

One Identity LLC
4 Polaris Way
Aliso Viejo, CA 92656

The TOE Evaluation was Sponsored by:

One Identity LLC
4 Polaris Way
Aliso Viejo, CA 92656

Evaluation Personnel:

Greg Beaver
Justin Fisher
Pascal Patin
Furukh Siddique

Contents

1 Introduction

This document presents results from performing assurance activities associated with the One Identity Safeguard for Privileged Sessions (SPS) evaluation. This report contains sections documenting the performance of assurance 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 and including the following optional and selection-based SFRs: FCS_HTTPS_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.1, FCS_TLSS_EXT.2, FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, FIA_X509_EXT.3, and FMT_MTD.1/CryptoKeys.

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 verified that the claimed NIST validation complies with the NIAP-approved PP requirements the TOE claims to satisfy. The CCTL verification of the NIST validation constitutes performance of the associated assurance 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 PP or its supporting document.

1.1 Applicable Technical Decisions

The NIAP Technical Decisions referenced below apply to [NDcPP]. Rationale is included for those Technical Decisions that do not apply to this evaluation.

TD0527 Updates to Certificate Revocation Testing (FIA_X509_EXT.1)

This TD is applicable to the TOE.

TD0528 NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4

This TD is not applicable to the TOE. The TD applies to FCS_NTP_EXT.1, which the TSF does not claim.

TD0536 NIT Technical Decision for Update Verification Inconsistency

This TD is applicable to the TOE.

TD0537 NIT Technical Decision for Incorrect Reference to FCS_TLSC_EXT.2.3

This TD is applicable to the TOE.

TD0538 NIT Technical Decision for Outdated link to Allowed-with List

This TD is not applicable to the TOE. The TOE does not claim any of the PP-Modules that the Allowed-with List references.

TD0546 NIT Technical Decision for DTLS - clarification of Application Note 63

This TD is not applicable to the TOE. The TOE does not claim DTLS.

TD0547 NIT Technical Decision for Clarification on developer disclosure of AVA_VAN

This TD is applicable to the TOE.

TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test

This TD is applicable to the TOE.

TD0556: NIT Technical Decision for RFC 5077 question

This TD is applicable to the TOE.

TD0563: NiT Technical Decision for Clarification of audit date information

This TD is applicable to the TOE.

TD0564: NiT Technical Decision for Vulnerability Analysis Search Criteria

This TD is applicable to the TOE.

TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7

This TD is not applicable to the TOE. The TD only applies in cases where TLS session resumption is supported, which the TSF does not claim.

TD0570: NiT Technical Decision for Clarification about FIA_AFL.1

This TD is applicable to the TOE.

TD0571: NiT Technical Decision for Guidance on how to handle FIA_AFL.1

This TD is applicable to the TOE.

TD0572: NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers

This TD is applicable to the TOE.

TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e

This TD is applicable to the TOE.

TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3

This TD is not applicable to the TOE. The TD adds a selection to FCS_CKM.2 that the TSF does not claim.

TD0591: NIT Technical Decision for Virtual TOEs and Hypervisors

This TD is not applicable to the TOE. The TOE does not have a virtual component.

TD0592: NIT Technical Decision for Local Storage of Audit Records

This TD is applicable to the TOE.

1.2 Evidence

[ST] One Identity Safeguard for Privileged Sessions v6.9 Security Target, Version 1.0, January 20, 2022

[CCECG] One Identity Safeguard for Privileged Sessions v6.9.3 Common Criteria Evaluated Configuration Guidance (CCECG), Version 1.1, January 26, 2022

[Admin]	One Identity Safeguard for Privileged Sessions 6.9.3 Administration Guide, April 30, 2021
[Install]	One Identity Safeguard for Privileged Sessions 6.9.3 Installation Guide, April 30, 2021
[REST]	One Identity Safeguard for Privileged Sessions 6.9.3 REST API Reference Guide, April 30, 2021
[Packaging]	One Identity Safeguard for Privileged Sessions 6.9.3 Packaging Checklist, April 30, 2021
[Upgrade]	One Identity Safeguard for Privileged Sessions 6.9.3 Upgrade Guide, April 30, 2021
[3000]	Super SC113 Chassis Series User's Manual, Version 1.0d.
[3500]	Supermicro SuperServer 1029U-T Series User's Manual, Revision 1.0i
[IPMI]	Supermicro BMC IPMI User's Guide, Revision 1.1b, August 26, 2020
[Release]	One Identity Safeguard for Privileged Sessions 6.9.4 Release Notes, January 27, 2022
[Test]	One Identity Safeguard for Privileged Sessions v6.9 Common Criteria Test Report and Procedures, Version 1.1, February 28, 2022
[VA]	One Identity Safeguard for Privileged Sessions v6.9 Vulnerability Assessment, Version 1.1, February 22, 2022

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
- [SD-ND] Evaluation Activities for Network Device cPP, Version 2.2, December 2019

1.4 SAR Evaluation

The following Security Assurance Requirements (SARs) were evaluated during the evaluation of the TOE:

SAR	Verdict
ASE_CCL.1	Pass
ASE_ECD.1	Pass
ASE_INT.1	Pass
ASE_OBJ.1	Pass
ASE_REQ.1	Pass

ASE_TSS.1	Pass
ADV_FSP.1	Pass
AGD_OPE.1	Pass
AGD_PRE.1	Pass
ALC_CMC.1	Pass
ALC_CMS.1	Pass
ATE_IND.1	Pass
AVA_VAN.1	Pass

The evaluation work units are listed in the proprietary ETR. The evaluators note per the PP evaluation activities that many of the SARs were successfully evaluated through completion of the associated evaluation activities present in the claimed PP.

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 Technical Decisions. 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 6.1.1 states for cryptographic key operations, the TOE logs the key name or certificate reference.

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 so this evaluation activity is N/A.

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 FAU_GEN.1 section of [CCECG] lists the auditable events mandated by the cPP and provides sample audit records for each event. This is split into two tables; the first table lists all of the SFRs along with sample events for each, to match the auditable events that the cPP defines in Tables 2, 4, and 5 for the SFRs that the TOE claims, and the second table lists the administrative actions for which auditable events are required, as specified in FAU_GEN.1.1 part c and the management functions that are claimed in FMT_SMF.1.

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 determined that the administrative actions for which auditable events are required should be derived from the management function claims. The evaluator observed that [CCECG] includes a table of the claimed management functions in the FAU_GEN.1 section. This table specifies sample audit records for each management function that is not explicitly covered by the SFR auditable events already specified in the tables 2, 4, or 5 of [NDcPP], specifically:

- Configuring the access banner
- Configuring the session inactivity time before session termination or locking
- Configuring the authentication failure parameters for FIA_AFL.1
- Configuring audit behavior (change to external audit storage location)
- Managing cryptographic keys (creating/deleting)
- Configuring the cryptographic functionality (configuration of TLS/SSH connection parameters)
- Managing the TOE's trust store and designating X.509v3 certificates as trust anchors
- Importing X.509v3 certificates into the TOE's trust store
- Creating/modifying users
- Configuring password policy

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 verified that the TOE was able to generate all of the audit records required for this SFR.

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

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

The 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.

This test was performed in conjunction with FAU_GEN.1. The records in that test show the IP addresses of external connections to the TOE.

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 6.1.3 states that audit data can transferred to an external syslog server using TLS, and that this happens in real-time (i.e. the TOE does not collect logs to transfer in batches at periodic intervals).

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 6.1.3 states that configuration changes are stored indefinitely and that other audit logs are retained for seven days. This section also states that the minimum size of the partition where audit logs reside is 10TB, and that the specific amount varies by TOE model. These logs are rotated daily, with a new log file being created each day, and the oldest log file being deleted. If somehow the TOE's storage space is exhausted, the TSF prevents execution of security-relevant functions.

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

The TSS states that the TOE is a single standalone component. It is not distributed.

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.

[ST] section 6.1.3 states that the TOE will prevent the execution of TSF functions if the audit trail becomes full.

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 6.1.3 states that the remote transfer of audit data is 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 so this evaluation activity is N/A.

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 so this evaluation activity is N/A.

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 FAU_STG_EXT.1 includes a reference to the "Configuring system logging" section of [Admin], which describes the steps needed to configure the external syslog server connectivity. [Admin] identifies the "Protocol" dropdown option as where TLS is enabled through the use of the TCP+TLS setting.

This section of [CCECG] also references [Admin] for information on how to enable mutual authentication for this interface and notes that this setting is optional in the evaluated configuration.

This section also references [REST] for guidance on how the same functions can be done via the REST API.

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.

[CCECG] states that audit data is written to the external syslog server in real-time.

The section of [Admin] referenced by the CCECG also states that the TOE buffers up to 10 MB of log messages to disk in case the syslog server is inaccessible.

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.

FAU_STG_EXT.1.3 in [ST] claims that the TSF prevents execution of its functions if audit storage space is exhausted. The corresponding TSS section states that configuration changes are stored indefinitely and that all other data is stored in a daily rotating log for the most recent seven days.

The FAU_STG_EXT.1 section of [CCECG] references the "Configuring system logging" section of [Admin] which notes that local log data is retained for seven days. This section

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 verified that the TOE is capable of securely transmitting audit records over TLS to an external server.

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 verified that the TOE disconnects external clients when audit storage space reaches a preset limit.

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

The TOE does not comply with FAU_STG_EXT.2/LocSpace. Therefore, this activity is not applicable.

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

The TOE is not distributed.

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 6.2.1 states that the TOE generates ECC keys for TLS and X.509, FFC keys for TLS, and safe-prime FFC keys for SSH. In all cases, this section also specifies the key sizes used by each scheme.

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.

[ST] claims the use of ECC and FFC schemes (with and without safe primes) for key generation. FFC with safe primes are used for SSH and the other schemes are used for TLS, and X.509 certificate generation.

[CCECG] under the FCS_CKM.1 and FCS_CKM.2 section states that the key generation algorithms used for TLS is dependent on the cipher suites being used and then references the FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of the same document for guidance on how to configure the cipher suites required by the evaluated configuration. Specifically, this is configured under the "Cryptography Settings" option in "Basic Settings > Trust Stores"

[CCECG] under the FCS_CKM.1 and FCS_CKM.2 section states that the key generation algorithms used for SSH is dependent on the key exchange algorithms being used and then references the

FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of the same document for guidance on how to configure the key exchange algorithms required by the evaluated configuration. Specifically, this is configured under the "Key exchange algorithms" option in "Settings" under "SSH Control".

[CCECG] under the FCS_CKM.1 section states that certificate signing requests will always result in the generation of ECC keys. This section also references the FIA_X509_EXT.3 section of the same document for guidance on how to generate a certificate signing request.

2.2.1.3 Test Activities

Key Generation for FIPS PUB 186-4 RSA Schemes

Performed in accordance with NIAP Policy Letter #5.

Key Generation for Elliptic Curve Cryptography (ECC)

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 5 ("Cryptographic Functions") identifies the CAVP certifications verifying asymmetric key generation, as follows.

Algorithm	Tested Capabilities	Certificates
RSA Schemes	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4	CAVP #C2178 (RSA)
ECC Schemes	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1 (P-256, P-384, P-521)	CAVP #C2178 (ECDSA)
FFC Schemes	NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography	N/A

Diffie-Hellman Group 14 and FFC Schemes using "safe-prime" groups

Testing for FFC Schemes using Diffie-Hellman group 14 and/or 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

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.

If Diffie-Hellman group 14 is selected from FCS_CKM.2.1, the TSS shall claim the TOE meets RFC 3526 Section 3. The intent of this activity is to be able to identify the scheme being used by each service. This would mean, for example, one way to document scheme usage could be:

Scheme	SFR	Service
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
Diffie-Hellman (Group 14)	FCS_SSHC_EXT.1	Backup 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 6.2.2 identifies the supported key establishment schemes and their uses as follows:

- ECC schemes: audit server, authentication server (TLS client), remote administration (TLS server), and SSH proxy (SSH client and server)
- FFC schemes: audit server, authentication server (TLS client), remote administration (TLS server)
- Safe-prime FFC schemes: SSH proxy (SSH client and server)

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).

[CCECG] under the FCS_CKM.1 and FCS_CKM.2 section states that the key establishment schemes used by the TOE will implicitly be determined by configuring the cipher suites (for TLS) and key exchange algorithms (for SSH). Configuring these interfaces to be consistent with the claims made in the TLS and SSH-specific SFRs will automatically cause the selected key establishment schemes to be used.

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.

SP800-56A Key Establishment Schemes

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 6 (“Cryptographic Functions”) identifies the CAVP certifications verifying SP 800-56A key establishment schemes, as follows.

Algorithm	Tested Capabilities	Certificates
Elliptic curve-based Schemes	NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography	KAS ECC #C2178

Algorithm	Tested Capabilities	Certificates
Finite field-based Schemes	NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography	KAS FFC #C2178

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.

N/A – the TOE does not implement RSA-based key establishment schemes.

Diffie-Hellman Group 14

The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses Diffie-Hellman group 14.

N/A – the TOE does not use Diffie-Hellman Groups.

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.

N/A – the TOE does not use safe-prime groups.

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 6.2.3 identifies in Table 7 the keys and CSPs used by the TOE, their storage location, when they're zeroized, and what mechanism is responsible for the zeroization. This section also describes the destruction mechanisms in sufficient detail to justify the selections that were made in FCS_CKM.4.1 ("destruction of reference…" and "instructs a part of the TSF to destroy…").

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 6.2.3 states that non-volatile key data is destroyed through invocation of an OS kernel API.

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.

The ST selects "destruction of reference" in FCS_CKM.4.1 for volatile memory. [ST] section 6.2.3 states that plaintext keys stored in volatile memory are destroyed by the explicit_bzero(3) call. TLS key data is retained in volatile memory until no longer needed and is destroyed by invocation of the OpenSSL cryptographic module's destruction mechanism.

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.

[ST] table 7 identifies that SSH client keys are stored encrypted in a credential store database using AES-256-CBC. This AES key used to protect these client keys (KEK) can itself be password-protected, where the SHA-512 hash of the password is used as the decryption key. If the KEK is not password-protected, it resides on the file system in a built-in credential store, and can be destroyed in a manner consistent with FCS_CKM.4. If it is password-protected, the key used to decrypt the KEK is a SHA-512 hash of the input password, and is therefore derived rather than stored persistently.

This table also identifies administrator credentials being stored in a non-plaintext format, but the storage mechanism is a hash, and there is therefore no KEK to consider.

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] section 6.2.3 ("FCS_CKM.4: Cryptographic Key Destruction") states a delay in the destruction of the CSP file may occur when the TOE is writing zeros into it before it has been flushed, but the TOE mitigates this by calling file flush immediately after zeroizing the CSP file.

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 specify the use of "a value that does not contain any CSP" to overwrite keys. This activity is therefore 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).

The [CCECG] section for FCS_CKM.4 identifies the various keys that can be deleted by an administrator and the instructions necessary to delete them. This section also states that regardless of whether keys are deleted automatically or manually, the zeroization process specified in the ST is followed by default, and there are no situations where this process is delayed.

2.2.3.3 Test Activities

None defined.

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 6.2.4 identifies the AES modes and key sizes supported by the TOE. The table in section 6.2 identifies the associated NIST algorithm validations.

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.

[ST] states that the TOE supports AES in CBC, CTR, and GCM modes with both 128-bit and 256-bit key sizes. CBC and GCM are supported for TLS and CTR is supported for SSH.

The FCS_COP.1/DataEncryption section of [CCECG] references the TLS and SSH sections of the guidance for instructions on how to configure the TOE to use the required AES algorithms.

2.2.4.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 6 ("Cryptographic Functions") identifies the CAVP certifications verifying AES encryption and decryption, as follows.

Algorithm	Tested Capabilities	Certificates
AES-CBC as defined in in ISO 10116	AES-CBC: Direction: Decrypt, Encrypt Key Lengths: 128, 256 (bits)	CAVP #C2178 (AES)
AES-GCM as defined in ISO 19772	AES-GCM: Direction: Decrypt, Encrypt IV Generation: Internal Key Lengths: 128, 256 (bits)	CAVP #C2178 (AES)
AES-CTR as defined in ISO 10116	AES-CTR: Direction: Encrypt Key Size: 128, 256 (bits)	CAVP #C2178 (AES)

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

2.2.5.1 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 6.2.5 identifies the digital signature algorithms and key sizes (modulus sizes for RSA) supported by the TOE. The table in section 6.2 identifies the associated NIST algorithm validations.

2.2.5.2 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.

[ST] states that the TOE supports RSA and ECDSA digital signatures.

The FCS_COP.1/SigGen section of [CCECG] references the TLS and SSH sections of the guidance for instructions on how to configure the TOE to use the required signature algorithms. This section also states that ECDSA is automatically used for CSRs and references the FIA_X509_EXT.3 section of the same document for guidance on how to perform this function.

2.2.5.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), Table 6 ("Cryptographic Functions") identifies the CAVP certifications verifying digital signature services, as follows.

Algorithm	Tested Capabilities	Certificates
RSA schemes using cryptographic key sizes [2048 bits] that meet FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 4	RSA SigGen/SigVer (FIPS186-4) PKCS 1.5 modulo 2048 with SHA-1/256/384/512 PKCSPSS modulo 2048 with SHA-1/256/384/512	CAVP #C2178 (RSA)
ECDSA schemes using [P-256, P-384, P-521] that meet FIPS 186-4, “Digital Signature Standard”, Section 6 and Appendix D	ECDSA SigGen/SigVer (FIPS186-4) Curve P-256 with SHA2-256/SHA2-384/SHA2-512 Curve P-384 with SHA2-256/SHA2-384/SHA2-512 Curve P-512 with SHA2-256/SHA2-384/SHA2-512	CAVP #C2178 (ECSA)

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

2.2.6.1 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 6.2.6 identifies the hash algorithms supported by the TOE as well as which specific algorithms are used for which functions. The table in section 6.2 identifies the associated NIST algorithm validations.

2.2.6.2 Guidance Activities

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

[ST] states that the TOE supports SHA-1, SHA-256, SHA-384, and SHA-512 hash algorithms.

The FCS_COP.1/Hash section of [CCECG] references the TLS and SSH sections of the guidance for instructions on how to configure the TOE to use the required hash algorithms. This section also states that SHA-256 is used automatically for self-testing without any administrator configuration.

2.2.6.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 6 (“Cryptographic Functions”) identifies the CAVP certifications verifying cryptographic hashing, as follows.

Algorithm	Tested Capabilities	Certificates
SHS as defined in ISO/IEC 10118-3:2004	SHA-1 SHA-256 SHA-384 SHA-512	CAVP #C2178 (SHA-1, SHA2-256, SHA2-384, SHA2-512)

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

2.2.7.1 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 6.2.7 identifies the HMAC algorithms supported by the TOE. For each algorithm, Table 8 identifies the associated hash algorithm, key size, block size, and message digest size (output MAC length) used. The table in section 6.2 identifies the associated NIST algorithm validations.

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 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.

[ST] states that the TOE supports HMAC with SHA-1, SHA-256, SHA-384, and SHA-512.

The FCS_COP.1/KeyedHash section of [CCECG] references the TLS and SSH sections of the guidance for instructions on how to configure the TOE to use the required keyed hash algorithms. Specifically, SSH supports HMAC-SHA-256 and HMAC-SHA-512, and TLS supports HMAC-SHA-1, HMAC-SHA-256, and HMAC-SHA-384.

2.2.7.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] (“Cryptographic Support”), Table 6 (“Cryptographic Functions”) identifies the CAVP certifications verifying cryptographic keyed hashing, as follows.

Algorithm	**Tested Capabilities**	**Certificates**
HMAC that meets ISO/IEC 9797-2:2011, Section 7 “MAC Algorithm 2”.	HMAC-SHA-1 Key size < block size Key size = block size Key size > block size HMAC-SHA2-256 Key size < block size Key size = block size Key size > block size HMAC-SHA2-384 Key size < block size Key size = block size Key size > block size HMAC-SHA2-512	CAVP #C2178 (HMAC-SHA-1, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512)

Algorithm	Tested Capabilities	Certificates
	Key size < block size Key size = block size Key size > block size	

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 6.2.8 states that the TOE's HTTPS protocol implementation is used for web-based administration and asserts blanket conformance with RFC 2818.

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 FCS_HTTPS_EXT.1 section of [CCECG] references the relevant portion of [Admin] for enabling TLS for the TOE's HTTP server and the TLS section of [CCECG} for establishing appropriate TLS configuration settings for this interface.

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.

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)

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 6.2.9 specifies that the TOE uses an AES-CTR DRBG that is seeded with 256 bits of minimum entropy that comes from a hardware-based noise source (Intel RDRAND). The table in section 6.2 identifies the associated NIST algorithm validation.

2.2.9.2 Guidance Activities

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

The FCS_RBG_EXT.1 section of [CCECG] states that compliant RNG functionality is configured by default and does not require user intervention.

2.2.9.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] ("Cryptographic Support"), ("Table 6 Cryptographic Functions") identifies the CAVP certifications verifying deterministic random bit generation, as follows.

Algorithm	Tested Capabilities	Certificates
CTR_DRBG in accordance with NIST Special Publication 800-90A	Counter DRBG Mode: AES-256	CAVP #C2178 (DRBG)

2.2.10 SSH Client (FCS_SSHC_EXT.1)

2.2.10.1 TSS Activities

FCS_SSHC_EXT.1.2 The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication and that this list conforms to FCS_SSHC_EXT.1.5 and ensure that if password-based authentication methods have been selected in the ST then these are also described.

[ST] section 6.2.10 states that the TOE supports both public key and password-based authentication methods. For public key authentication, the public key algorithms claimed are consistent with FCS_SSHC_EXT.1.5.

FCS_SSHC_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 6.2.10 states that the threshold for 'large packets' is 131072 bytes such that everything larger than that is automatically discarded.

FCS_SSHC_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 6.2.10 states that 128/256-bit AES-CTR and CBC are used for the TOE's SSH transport implementation, which is consistent with the SFR claims. This section also states that no optional characteristics are supported.

FCS_SSHC_EXT.1.5

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

If x509v3-based public key authentication algorithms are claimed, the evaluator shall confirm that the TSS includes the description of how the TOE establishes the server's identity and how this identity is confirmed with the one that is presented in the provided certificate. For example, the TOE could verify that a server's configured IP address matches the one presented in the server's x.509v3 certificate.

[ST] section 6.2.10 identifies the public key algorithms that are supported, consistent with the SFR claims that are made in FCS_SSHC_EXT.1.5. This section also states that no optional characteristics are specified.

The TOE does not claim any X.509v3-based public key authentication algorithms so the second part of the evaluation activity is N/A.

FCS_SSHC_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 6.2.10 identifies the supported data integrity algorithms, consistent with the claims made in FCS_SSHC_EXT.1.6.

FCS_SSHC_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 6.2.10 identifies the supported key exchange algorithms, consistent with the claims made in FCS_SSHC_EXT.1.7.

FCS_SSHC_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 6.2.10 states that a rekeying of the SSH session key occurs after one hour or one gigabyte of data, whichever happens first.

2.2.10.2 Guidance Activities

FCS_SSHC_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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required encryption algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

FCS_SSHC_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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required public key algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

FCS_SSHC_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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required MAC algorithms for the SSH implementation along with instructions for how to configure them. However, it notes that the default algorithms are consistent with the evaluated configuration of the TOE so the configuration instructions are provided for informational purposes only.

FCS_SSHC_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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required key exchange algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

FCS_SSHC_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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] states that the rekey thresholds are not configurable.

2.2.10.3 Test Activities

FCS_SSHC_EXT.1.2

Test 1: If password-based authentication methods have been selected in the ST then using the guidance documentation, the evaluator shall configure the TOE to perform password-based authentication to an SSH server and demonstrate that a Security Administrator can be successfully authenticated by the TOE to an SSH server using a password as an authenticator.

Note: Public key authentication is tested as part of testing for FCS_SSHC_EXT.1.5

The evaluator demonstrated the ability to use password authentication to open a connection to the TOE.

FCS_SSHC_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 terminates an SSH connection after receiving an abnormally large packet.

FCS_SSHC_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 with a remote server (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 established an SSH connection from the TOE using each of the ciphers claimed in the [ST].

FCS_SSHC_EXT.1.5
Test 1: The evaluator shall establish an SSH connection using each of the public key algorithms specified by the requirement to authenticate an SSH server to the TOE. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test. Test objective: The purpose of this positive test is to check the authentication of the server by the client (when establishing the transport layer connection), and not for checking generation of the authentication message from the client (in the User Authentication Protocol). The evaluator shall therefore establish sufficient separate SSH connections (with an appropriately configured server) to cause the TOE to demonstrate use of all public key algorithms claimed in FCS_SSHC_EXT.1.5 in the ST.

The evaluator established an SSH connection from the TOE using each of the public key algorithms specified in the [ST].

FCS_SSHC_EXT.1.5
Test 2: The evaluator shall configure an SSH server to only allow a public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the TOE to the SSH server and observe that the connection is rejected.

The evaluator attempted to establish an SSH connection from the TOE to an SSH server that disallowed the public key algorithms supported by the TOE. The TOE was observed to terminate the connection attempt after receiving the server's Key Exchange Init message.

FCS_SSHC_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 established an SSH connection from the TOE to a TLS test server using teach of the HMAC algorithms claimed in the [ST]

FCS_SSHC_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 server to only allow a MAC algorithm that is not included in the ST selection. The evaluator shall attempt to connect from the TOE to the SSH server 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 attempted to establish an SSH connection from the TOE to a server that was configured to use an HMAC algorithm that the TOE did not support. The TOE was observer to terminate the connection attempt after receiving the server’s Key Exchange Init message.

FCS_SSHC_EXT.1.7

Test 1: The evaluator shall configure an SSH server to permit all allowed key exchange methods. The evaluator shall attempt to connect from the TOE to the SSH server using each allowed key exchange method and observe that each attempt succeeds.

The evaluator established an SSH connection from the TOE to an SSH server using each of the key exchange methods claimed in the [ST].

FCS_SSHC_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.

For testing of the time-based threshold, the evaluator shall use the TOE to connect to an SSH server 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 server the TOE is connected to.

For testing of the traffic-based threshold the evaluator shall use the TOE to connect to an SSH server 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_SSHC_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 server the TOE is connected to.

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).

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

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

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

The evaluator verified that the TOE rekeyed an SSH connection after exceeding limits of 1 hour and 1 GB of data.

FCS_SSHC_EXT.1.9

Test 1: The evaluator shall delete all entries in the TOE's list of recognized SSH server host keys and, if selected, all entries in the TOE's list of trusted certification authorities. The evaluator shall initiate a connection from the TOE to an SSH server. The evaluator shall ensure that the TOE either rejects the connection or displays the SSH server's public key (either the key bytes themselves or a hash of the key using any allowed hash algorithm) and prompts the Security Administrator to accept or deny the key before continuing the connection.

The evaluator verified that the TOE would not connect to an external SSH server whose host key it did not recognize. The server's host key had to be added to the TOE in order for an SSH connection to proceed.

FCS_SSHC_EXT.1.9

Test 2: The evaluator shall add an entry associating a host name with a public key into the TOE's local database. The evaluator shall replace, on the corresponding SSH server, the server's host key with a different host key. If 'password-based' is selected for the TOE in FCS_SSHC_EXT.1.2, the evaluator shall initiate a connection from the TOE to the SSH server using password-based authentication, shall ensure that the TOE rejects the connection, and shall ensure that the password was not transmitted to the SSH server (for example, by instrumenting the SSH server with a debugging capability to output received

passwords). If 'password-based' is not selected for the TOE in FCS_SSHC_EXT.1.2, the evaluator shall initiate a connection from the TOE to the SSH server using public key-based authentication and shall ensure that the TOE rejects the connection

The evaluator verified that the TOE terminated an SSH connection when it received a host key from a server that did not match what was expected.

2.2.11 SSH Server (FCS_SSHS_EXT.1)

2.2.11.1 TSS Activities

FCS_SSHS_EXT.1.2
The evaluator shall check to ensure that the TSS contains a description of the public key algorithms that are acceptable for use for authentication and that this list conforms to FCS_SSHS_EXT.1.5. and ensure that if password-based authentication methods have been selected in the ST then these are also described.

[ST] section 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

FCS_SSHS_EXT.1.5
The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics are specified, and the public key algorithms supported are specified as well. The evaluator shall check the TSS to ensure that the public key algorithms specified are identical to those listed for this component.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

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 6.2.11 states that the TOE's SSH server implementation uses identical cryptographic parameters and configuration to the client implementation and therefore references section 6.2.10 for that information.

2.2.11.2 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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required encryption algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required public key algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required MAC algorithms for the SSH implementation along with instructions for how to configure them. However, it notes that the default algorithms are consistent with the evaluated configuration of the TOE so the configuration instructions are provided for informational purposes only.

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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] identifies the required key exchange algorithms for the SSH implementation along with instructions for how to configure them. This section also notes what the default configuration is so that it's clear to the reader what the initial state of the product is when first deployed.

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 FCS_SSHC_EXT.1 and FCS_SSHS_EXT.1 section of [CCECG] states that the rekey thresholds are not configurable.

2.2.11.3 Test Activities

FCS_SSHS_EXT.1.2

Test 1: If password-based authentication methods have been selected in the ST then using the guidance documentation, 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 user.

The evaluator verified that password-based authentication could be used to establish an SSH connection to the TOE.

FCS_SSHS_EXT.1.2

Test 2: If password-based authentication methods have been selected in the ST then the evaluator shall use an SSH client, enter an incorrect password to attempt to authenticate to the TOE, and demonstrate that the authentication fails.

Note: Public key authentication is tested as part of testing for FCS_SSHS_EXT.1.5.

The evaluator verified that the TOE rejected an SSH connection when an incorrect password was entered.

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 receiving an abnormally large SSH packet caused it to terminate an SSH connection.

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 established an SSH connection to the TOE using each of the SSH ciphers claimed in the [ST].

FCS_SSHS_EXT.1.5
Test 1: The evaluator shall establish an SSH connection using each of the public key algorithms specified by the requirement to authenticate the TOE to an SSH client. It is sufficient to observe (on the wire) the successful negotiation of the algorithm to satisfy the intent of the test.

The evaluator established an SSH connection to the TOE using each of the public key algorithms claimed in the [ST].

FCS_SSHS_EXT.1.5
Test 2: The evaluator shall choose one public key algorithm supported by the TOE. The evaluator shall generate a new key pair for that algorithm without configuring the TOE to recognize the public key for authentication. The evaluator shall use an SSH client to attempt to connect to the TOE with the new key pair and demonstrate that authentication fails. Test objective: The purpose of this negative test is to verify that the server rejects authentication attempts of clients that present a public key that does not match public key(s) associated by the TOE with the identity of the client (i.e. the public keys are unknown to the server). To demonstrate correct functionality, it is sufficient to determine that an SSH connection was not established after using a valid username and an unknown key of supported type.

The evaluator attempted to open an SSH connection to the TOE using a public key that the TOE did not recognize. The TOE rejected the connection attempt.

FCS_SSHS_EXT.1.5
Test 3: The evaluator shall configure an SSH client to only allow a public key algorithm that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection from the SSH client to the TOE and observe that the connection is rejected.

The evaluator attempted to open a connection to the TOE using a public key algorithm not claimed in the [ST]. The TOE rejected the connection attempt.

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 opened an SSH connection to the TOE using each of the MAC algorithms claimed 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 attempted to open an SSH connection to the TOE using a MAC algorithm not claimed in the [ST]. The TOE rejected the connection attempt.

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 attempted to open an SSH connection to the TOE using an SSH client that only allowed diffiehellman-group1-sha1 key exchange. The TOE rejected the connection 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 opened an SSH connection to the TOE using each of the key exchange methods claimed 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.

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

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

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

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

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

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

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

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

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

The evaluator verified that the TOE rekeyed an SSH connection after 1 hour and 1 GB of data transmission.

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

2.2.12.1 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 listed for this component.

[ST] section 6.2.12 identifies the TLS ciphersuites supported by the TOE, and that these are consistent with the selections made 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.

[ST] section 6.2.12 states that the TOE uses hostname and distinguished name for its reference identifiers, and that neither IP addresses nor 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.

The TOE is not distributed so this evaluation activity is not applicable.

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.

The TOE does not support IP addresses in the CN as reference identifiers and therefore this evaluation activity is not applicable.

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 6.2.12 states that the TOE supports secp256r1 in the Supported Elliptic Curves/Supported Groups extension by default.

2.2.12.2 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 FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of [CCECG] states that TLS versions 1.1 and 1.2 are supported and includes instructions for configuring the supported TLS versions. This section also describes how to configure the supported TLS cipher suites and lists the configuration string used to set cipher suites that are consistent with [ST].

This section also describes how to configure TLS for the supported TLS client interfaces (syslog and authentication servers).

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.

The FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of [CCECG] states that DN and hostname are automatically used as the peer reference identifier; no configuration exists for this function.

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.

This activity is N/A to the TOE; FPT_ITT.1 is not claimed.

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] Section 6.2.12 indicates that the Supported Elliptic Curves Extension is enabled by default. Therefore, this activity is N/A to the TOE.

2.2.12.3 Test Activities

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 connection from the TOE to an external TLS test server using each of the algorithms claims in the [ST].

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 verified that the TOE rejected a connection attempt to a server whose certificate lacked the Server Authentication purpose in the extendedKeyUsage field.

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

The evaluator verified that the TOE would not connect to a TLS test server whose certificate did not match the server-selected ciphersuite.

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 verified that the TOE would not connect to a server that was configured to select the TLS_NULL_WITH_NULL_NULL ciphersuite.

The evaluator verified that the TOE would not connect to a TLS test server which presented a ciphersuite selection that was not in the Client Hello message.

The evaluator verified that the TOE would not connect to a TLS test server which used selected an elliptic curve not supported by the TOE.

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 verified that the TOE would not connect to a TLS test server whose TLS version was 0x0306.

The evaluator verified that the TOE terminated a TLS connection when it received a Server Key Exchange message with a modified signature block.

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 verified that the TOE would reject a TLS connection from a server that modified the Server Finished message.

The evaluator verified that the TOE would terminate a TLS connection after receiving a garbled message after the ChangeCipherSpec message.

The evaluator verified that the TOE would terminate a TLS connection after receiving a Server Hello message with a modified nonce.

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.

[Test Notes]

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.

[Test Notes]

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

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 verified that the TOE would reject a connection to a server whose certificate had an incorrect CN and did not have the SAN extension.

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 verified that the TOE would reject a connection to a TLS server that presented a certificate with a correct CN but an incorrect SAN.

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 verified that the TOE would open a TLS connection to a server whose certificate had a CN that matched the reference identifier and did not contain the SAN extension.

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 verified that the TOE would open a TLS connection to a server whose certificate had a correct value in the SAN but an incorrect value in the CN.

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, URIID):

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 would not open a TLS connection to a server whose certificate had a wildcard outside of the left-most label in the identifier.

The evaluator verified that the TOE would open a connection a to a TLS server whose certificate had a wildcard in the left-most label. If the TOE's reference identifier had two left-most labels or none then the TOE would terminate the connection attempt.

Test 6 [conditional]: If IP addresses are supported, 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 an asterisk (*) (e.g. CN=192.168.1.* when connecting to 192.168.1.20, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

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

N/A, the TOE does not claim support for IP address reference identifiers.

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, the TOE does not claim FPT_ITT.

FCS_TLSC_EXT.1.3

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

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

Refer to FCS_TLSC_EXT.1.1 Test 1. That test demonstrates the ability of the TOE to connect to a server whose certificate is issued by a trusted CA.

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.

Covered by FCS_TLSC_EXT.1.3 Test 1, FIA_X509.1.1/REV Test 1, FIA_X509_EXT.1.1/REV Test 2, and FIA_X509_EXT.1.1/REV Test 3 Testing

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 offer any administrative override for this.

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 verified that the TOE could establish a TLS connection using the curve claimed in the ST.

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

2.2.13.1 TSS Activities

FCS_TLSC_EXT.2.1

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 6.2.13 states that the TOE optionally supports TLS mutual authentication for its TLS client interfaces.

2.2.13.2 Guidance Activities

FCS_TLSC_EXT.2.1

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 FCS_TLSC_EXT.2 section of [CCECG] identifies the settings that must be enabled to use mutual authentication for the TOE's TLS client interfaces.

2.2.13.3 Test Activities

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

FCS_TLSC_EXT.2.1

(covered by FCS_TLSC_EXT.1.1 Test 1 and testing for FIA_X.509_EXT.*).

The evaluator verified the TOE could be configured for TLS mutual authentication.

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

2.2.14.1 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 6.2.14 states that the TLS server ciphersuites are identical to those claimed for the TLS client, which is consistent with the claims made in FCS_TLSS_EXT.1.1.

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 6.2.14 states that the TOE prevents the use of TLS versions other than 1.1 and 1.2 by configuration.

FCS_TLSS_EXT.1.3

If using ECDHE or DHE ciphers, the evaluator shall verify that the TSS describes the key agreement parameters of the server Key Exchange message.

[ST] section 6.2.14 states that the TOE uses 2048-bit DH parameters and secp256r1 ECDHE curves for key exchange.

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).

[ST] section 6.2.14 states that session resumption is not supported.

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.

N/A; session tickets are not supported.

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.

N/A; session tickets are not supported.

2.2.14.2 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 FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of [CCECG] references the HTTPS section for guidance on how to enable TLS for the web server and configure the appropriate cipher suites and TLS versions.

FCS_TLSS_EXT.1.2
The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

The FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of [CCECG] states that there is no configuration that is necessary to meet the requirements for reference identifier validation.

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 FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 section of [CCECG] states that there is no configuration that is necessary to meet the requirements for key establishment parameters.

2.2.14.3 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 verified that a TLS test client could open a connection to the TOE using each of the ciphersuites claimed in the [ST].

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 verified that the TOE would reject a TLS connection attempt with an unsupported ciphersuite or the TLS_NULL_WITH_NULL_NULL ciphersuite.

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 verified that the TOE would reject a TLS connection attempt where the last byte in the Client Finished message was modified.

The evaluator verified that TLS data was actually encrypted as claimed.

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 verified that the TOE rejected TLS connection attempts for TLS 1.0 and lower.

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 verified that the TOE would open a TLS connection with a client which used the p256 curve.

The evaluator verified that the TOE would not open a TLS connection with a client which used the p192 curve.

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 with a 2048 bit DHE parameter.

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.

Only DHE and ECDHE ciphersuites are claimed in FCS_TLSS_EXT.1.1.

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).

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.

The evaluator verified that the TOE responded to a Client Hello with no session ticket and with a Session ID length of 0.

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.

The TOE does not support session resumption.

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.

The TOE does not support session tickets.

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 6.3.1 states that the authentication failure handling mechanism applies to all management interfaces except for the local console, so there is no need to provide descriptions of individual interface behaviors.

This section describes the mechanism as causing a lockout after a configurable number of consecutive failures has happened (1-50 with the default being 20). When locked out, an administrator is unable to access the TOE for a configurable period of time (1-720 minutes with a default of 10) or until a local administrator manually unlocks them.

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 6.3.1 states that lockout behavior does not apply to the local console so it can be assumed that local administrator access is always available. From this interface, a remote administrator account can always be unlocked, so there is no possibility by which a deliberate or inadvertent series of failed authentication attempts can cause a denial of service of the TOE's management capability.

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.

The FIA_AFL.1 section of [CCECG] references the "Protecting against brute-force attacks" section of [Admin] for guidance on configuring the maximum number of authentication failures and lockout period. Specifically, this requires a configuration setting to be enabled (which is enabled by default), which then allows for the maximum failure threshold and the lockout period to be configured.

This section also states that a locked out user can have their access restored through one of the following mechanisms:

- The configured lockout period expires
- The server is physically rebooted
- The root user on the local console users the "Clear list of blocked users" option

This section also states that the lockout mechanism applies to all interfaces except the local console.

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

The FIA_AFL.1 section of [CCECG] states that the local console interface is not subject to the lockout function, so a mechanism for unlocking other user accounts always exists.

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 verified that after reaching a maximum number of unsuccessful number of login attempts it was no longer possible to access the TOE for that account.

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 verified that after a designated lockout period a previously locked out account was able to access the TOE again.

2.3.2 Password Management (FIA_PMG_EXT.1)

2.3.2.1 TSS Evaluation Activity

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

[ST] section 6.3.2 states that administrator passwords have a configurable minimum length between 1 and 99 characters. This section also explicitly lists the special characters that are supported for passwords.

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.

The FIA_PMG_EXT.1 section of [CCECG] states that the allowable password characters are specified by setting the "Minimal password strength" parameter to strong and identifies the characters that are

allowed when this setting is enabled. This section also states that the "Minimal password length" setting is used to set the minimum password length from anywhere between 8 and 99 characters.

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 verified that the TOE would accept passwords that met its password policy.

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 verified that the TOE would not accept passwords that did not meet its password policy.

2.3.3 Protected Authentication Feedback (FIA_UAU.7)

2.3.3.1 TSS Evaluation Activity

None defined.

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.

The guidance documentation does not identify any necessary configuration to ensure that when logging in, the TOE will not echo passwords at the login display. This function is enabled by default and not require configuration.

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 verified that passwords were obscured when they were entered into the web GUI login page.

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 6.3.3 states that the TOE supports local and remote administration over HTTPS. The logon process for this, regardless of whether it's local or remote, is to provide a username/password, which is either done locally by the TOE or by an environmental authentication server (LDAP/AD), depending on configuration. A successful logon is when the supplied credentials are consistent with the expected value for the claimed identity.

This section also identifies a local admin interface that is accessible only via a local console port and is authenticated using a local password.

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 6.3.3 states that the only pre-authentication TOE function that can be performed is the display of the login 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 so this evaluation activity 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 so this evaluation activity 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.

[ST] claims support for local password-based authentication and also states that authentication can be performed by an external LDAP/AD server, depending on configuration.

The FIA_UIA_EXT.1 section of [CCECG] describes how to set up the initial username/password for the console root admin as well as the default Web UI admin by referencing the "The Welcome Wizard and the first login" section of [Admin]. This section of [CCECG] also references "Managing One Identity Safeguard for Privileged Sessions (SPS) users from an LDAP database" in [Admin] for guidance on how to configure LDAP/AD authentication.

No configuration is needed to limit the services available prior to login; [CCECG] identifies that the only pre-login function available is viewing the pre-authentication warning banner.

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 verified that correct credentials could be used to access the web GUI while incorrect credentials resulted in access being denied.

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 verified that no unexpected servers are available on the TOE.

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.

Local access is provided by connecting a network cable directly into the TOE. See FIA_UIA_EXT.1 Test 2.

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.

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 6.3.6 states that certificate validation is performed when a TLS client or server certificate is presented to the TOE, or when a certificate for the TOE's own use is loaded onto it. This section also notes that the extendedKeyUsage fields for TLS server certificates and TLS client certificates are checked where appropriate, and that the Code Signing and OCSP Signing purpose checks are trivially satisfied because the TOE does not use X.509 certificates for either of those functions.

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 6.3.6 states that CRL checking is used for all certificate validation operations.

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.

The FIA_X509_EXT.1/Rev section of [CCECG] states that the TOE checks certificate validity when the TOE is acting as a TLS client and when a new certificate is loaded onto the TOE for its own use.

The FIA_X509_EXT.1/Rev section of [CCECG] states that the OCSP Signing, Code Signing, and Client Authentication extendedKeyUsage purpose are N/A as there are no situations where the TOE validates such certificates in the evaluated configuration.

The FIA_X509_EXT.1/Rev section of [CCECG] states that CRL is used for revocation checking. This section references "Verifying certificates with Certificate Authorities using trust stores" in [Admin] for guidance on how to configure custom trust stores, which includes how to enable revocation checking and where to specify the CRL URL. For secure usage, [CCECG] notes that the "None" setting for CRL checking should not be used so that CRL checking is enabled. This section also states that CRLs are configured by individual connections by configuring one or more trust stores (which has CRL checking enabled) to be used and then associating a connection with one of those trust stores.

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:

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

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

The evaluator created a certificate chain with a root CA, an intermediate CA and a leaf certificate. The leaf certificate was used to identify a TLS test server.

The root and intermediate CA certificates were loaded into the TOE's trust store. The TOE was then able to open a TLS connection to a server identified by the leaf certificate.

The intermediate CA was then removed from the TOE's trust store and the connection above was re-attempted. The TOE did not accept the TLS server's certificate. It should be noted that the TOE received the full certificate chain but still rejected the connection attempt because the intermediate certificate was not in the TOE's trust store. It is not sufficient to have the root CA in the trust store. This is relevant to FIA_X509_EXT.1.1 Test 8.

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

The evaluator attempted to open a TLS connection to a server that identified itself with an expired certificate. The TOE rejected the certificate and terminated the connection attempt.

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 certificate chain created for Test 1 used CRLs to check the revocation status of both the leaf certificate and the intermediate CA.

The evaluator first verified that the TOE could open a TLS connection to a TLS server when all of the certificates were valid. Next the evaluator revoked the leaf certificate and verified that the TOE would not open a TLS connection. A good leaf certificate but a revoked intermediate CA were then used and the TOE again rejected the connection attempt.

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 used a CRL signed by a CA that did not have cRLsign set to check the revocation status of a leaf certificate. When attempting to connect to a TLS server identified by that certificate the TOE rejected the connection attempt because of the bad CRL signing CA.

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 rejected a connection attempt to a TLS server whose certificate had its first eight bytes modified.

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 rejected a connection attempt to a TLS server whose certificate had its signatureValue field modified.

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 rejected a connection attempt to a TLS server whose certificate had its

Test added in accordance with TD0527.

Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall conduct the following tests:

Test 8a: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a valid chain of EC certificates (terminating in a trusted CA certificate), where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain.

Test 8b: (Conditional on TOE ability to process CA certificates presented in certificate message) The test shall be designed in a way such that only the EC root certificate is designated as a trust anchor, and by setting up the trust store in a way that the EC Intermediate CA certificate needs to be provided, together with the leaf certificate, from outside the TOE to complete the chain (e.g. by storing only the EC root CA certificate in the trust store). The evaluator shall present the TOE with a chain of EC certificates (terminating in a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

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

Tests 8a and 8b are not applicable because CA certificate must be loaded into the TOE's trust store, not presented in the certificate message.

The evaluator verified that the TOE would not load an intermediate CA certificate with explicit format elliptic curve parameters into its trust store.

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 would not validate a leaf certificate that had been issued by an intermediate CA which lacked the basicConstraints extension.

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 would not validate a leaf certificate that had been issued by an intermediate CA which had the basicConstraints extension set to FALSE.

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).

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 6.3.7 states that when the TOE is validating a TLS client or server certificate, or when a certificate is loaded onto it for its own use, it uses the certificate that is presented to it for validation. When the TOE is providing its own server certificate to a TLS client, or when mutual authentication is configured and the TOE is providing its own client certificate to a TLS server, it will only have one certificate loaded onto it for such purposes.

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 6.3.7 states that a certificate with undetermined revocation status will be rejected in all cases, with no administrative ability to override this.

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 FIA_X509_EXT.1/Rev section of [CCECG] describes how to import and use certificates for use with the TOE, and how to configure trust stores so that the TOE has a trust relationship with the external certificates that are presented to it. This section also states that one or more external PKIs that support CRLs must be configured and used for environmental connections in order for the TOE's certificate validation function to work as specified by the evaluated configuration.

There is no administrative option to configure the TOE's behavior when a certificate's revocation status cannot be determined, per [ST].

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 verified that when the TOE could not download a CRL for a certificate it would not open a TLS connection to a server which used that certificate.

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.

The ST does not select “device-specific information” in FIA_X509_EXT.3.1. Therefore, this activity is not applicable.

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.

[CCECG] section FIA_X509_EXT.3 references instructions for how to generate a CSR for the TOE’s own certificate and how to import the certificate that is subsequently issued by the signing CA for use within the TOE.

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 certificate signing request on the TOE and verified that its format was correct.

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 verified that a signed CSR would not be accepted to the TOE if the CA that signed it was not in the TOE’s trust store. Once the signing CA was added the TOE would accept the signed CSR.

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 so this evaluation activity 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 so this evaluation activity is not applicable.

2.4.1.3 Test 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 so this evaluation activity is not applicable.

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

2.4.2.1 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 so this evaluation activity is not applicable.

2.4.2.2 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 FMT_MOF.1/ManualUpdate section of [CCECG] identifies the vendor website where software/firmware updates are located and makes it clear to the reader that updates must be obtained outside of the TOE. Information about the actual update process is discussed under FPT_TUD_EXT.1 below.

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 so this evaluation activity is not applicable.

2.4.2.3 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.

The evaluator verified that the interface that is required to perform an updated to the TOE is not available to a non-administrative user.

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.

See FPT_TUD_EXT.1.

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

2.4.3.1 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 6.4.2 states that no TSF administrative functions are accessible prior to administrator authentication.

[ST] section 6.4.2 identifies how the TOE's management interfaces are rendered inaccessible to non-administrative users, as follows:

- For the web GUI, an administrator must have valid credentials to authenticate to the TOE (either locally defined or define in an environmental AD/LDAP directory) and their specific administrative privileges are derived from role-based permissions that are associated to them based on group membership. [ST] section 6.4.5 lists the roles/groups that the TOE includes by default.
- For the local console, a root user is established during initial setup as the Security Administrator for that interface. This is the only account that is authorized to authenticate to the local console.

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 6.4.2 states that the certificate trust store resides in access controlled storage and can only be managed by a Security Administrator using the web interface.

2.4.3.2 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 it identifies management instructions each claimed TSF-data-manipulating function as follows:

- Ability to configure the access banner
    - UI: references “Authentication banner” section of [Admin]
    - API: not managed from this interface
    - Console: not managed from this interface
- Ability to configure session inactivity time before session termination or locking
    - UI: references “Web interface timeout” section of [Admin]
    - API: references “Web interface” section of [REST]
    - Console: not managed from this interface
- Ability to update the TOE
    - UI: references “Upgrading One Identity Safeguard for Privileged Sessions (SPS) (single node)” in [Admin]
    - API: not managed from this interface
    - Console: not managed from this interface
- Ability to configure the authentication failure parameters for FIA_AFL.1
    - UI: references “Protecting against brute-force attacks” in [Admin]
    - API: not managed from this interface
    - Console: not managed from this interface
- Ability to configure audit behavior
    - UI: references “Configuring system logging” in [Admin]
    - API: references “Syslog server settings” in [REST]
    - Console: not managed from this interface
- Ability to manage cryptographic keys

- UI: FMT_MTD.1/CryptoKeys section of [CCECG] is referenced, which lists the various cryptographic keys and how the various management operations are performed against them.
- API: FMT_MTD.1/CryptoKeys section of [CCECG] is referenced, which lists the various cryptographic keys and how the various management operations are performed against them.
- Console: not managed from this interface

- Ability to configure the cryptographic functionality
  - UI: references “Verifying certificates with Certificate Authorities using Trust Stores” in [Admin] for TLS and “Creating and editing protocol-level SSH settings” in [Admin] for SSH
  - API: references “SSH settings policies” in [REST]
  - Console: not managed from this interface
- Ability to set the time which is used for timestamps
  - UI: references “Configuring date and time” in [Admin]
  - API: not managed from this interface
  - Console: not managed from this interface
- Ability to manage the TOE’s trust store and designate X.509v3 certificates as trust anchors
  - UI: references “Verifying certificates with Certificate Authorities using trust stores” in [Admin]
  - API: references “Trust Stores” in [REST]
  - Console: not managed from this interface
- Ability to import X.509v3 certificates into the TOE’s trust store
  - UI: references “Verifying certificates with Certificate Authorities using trust stores” in [Admin]
  - API: references “Trust Stores” in [REST]
  - Console: not managed from this interface
- Ability to create and modify users
  - UI: references “Creating local users in One Identity Safeguard for Privileged Sessions (SPS)” in [Admin]
  - API: not managed from this interface
  - Console: not managed from this interface
- Ability to unlock users
  - UI: not managed from this interface

- API: not managed from this interface
- Console: managed by navigating to Troubleshooting and selecting the Clear list of blocked users/IPs option

- Ability to configure password policy
  - UI: references “Setting password policies for local users” in [Admin]
  - API: not managed from this interface
  - Console: not managed from this interface

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

The TOE does support handling of X.509v3 certificates and provides a trust store. It also provides loading of CA certificates.

The FMT_MTD.1/CoreData section of [CCECG] references the FMT_SMF.1 section of the document for guidance on how to manage trust stores and designate CA certificates as trust anchors. It states that an administrator must have the “read” and “write/perform” permissions for the “Basic Settings” object to perform this function.

2.4.3.3 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.

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

2.4.4.1 TSS Activities

For distributed TOEs see chapter 2.4.1.1.

The TOE is not distributed so this evaluation activity 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 6.4.3 states that the Security Administrator can manage the private keys used for TLS and SSH over the TOE’s web interface.

2.4.4.2 Guidance Activities

For distributed TOEs see chapter 2.4.1.2.

The TOE is not distributed so this activity is not applicable.

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 FMT_MTD.1/CryptoKeys section of [CCECG] includes guidance on all of the cryptographic keys that can be directly managed by the TOE, the operations that can be performed against them, and where guidance can be found for how to perform these operations.

2.4.4.3 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.

The evaluator verified that a non-administrative user does not have access to alter the TOE's cryptographic keys. Non-administrative users are given read-only access.

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

The evaluator verified that an administrative user was able to delete a trust store.

2.4.5 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.

2.4.5.1 TSS Activities (also including activities for 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 6.4.4 lists all of the management functions supported by the TOE. It also identifies the interfaces that can be used to perform the management functions; specifically, the local console is only used to unlock a locked administrator account or to change a password, and all other management functions can be performed using the web UI or API.

The following paragraphs identify each management function specified in FMT_SMF.1 and how each is covered in the TSS. Guidance documentation references to these functions are referenced in [CCECG]; refer to FMT_MOF.1/CoreData above.

Ability to administer the TOE locally and remotely—[ST] section 6.4.4 states that this function is supported.

Ability to configure the access banner—[ST] section 6.4.4 states that this function is supported.

Ability to configure the session inactivity time before session termination or locking—[ST] section 6.4.4 states that this function is supported.

Ability to update the TOE, and to verify the updates using hash comparison capability prior to installing those updates—[ST] section 6.4.4 states that this function is supported.

Ability to configure the authentication failure parameters for FIA_AFL.1—[ST] section 6.4.4 states that this function is supported.

Ability to configure audit behaviour (i.e. changes to remote storage locations for audit)—[ST] section 6.4.4 states that this function is supported.

Ability to manage the cryptographic keys—[ST] section 6.4.4 states that this function is supported.

Ability to configure the cryptographic functionality—[ST] section 6.4.4 states that this function is supported.

Ability to set the time which is used for time-stamps—[ST] section 6.4.4 states that this function is supported.

Ability to manage the TOE's trust store and designate X.509v3 certificates as trust anchors—[ST] section 6.4.4 states that this function is supported.

Ability to import X.509v3 certificates to the TOE's trust store—[ST] section 6.4.4 states that this function is supported.

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 6.3.3 identifies two local management interfaces:

- Direct network connection (via crossover cable) to invoke the HTTPS web UI or REST API
- Physical access to console interface

[CCECG] section for FMT_SMF.1 states that local management of the TOE is possible by directly connecting an administrative computer to the console serial port, or by direct connection to the Ethernet port for Web UI or REST API access.

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 so this evaluation activity is not applicable.

2.4.5.2 Guidance Activities

See section 2.4.4.1. (2.4.6.1 in this AAR)

This activity was completed in section 2.4.6.1 above.

2.4.5.3 Test Activities

The evaluator tests management functions as part of testing the SFRs identified in section 2.5.7. 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.

2.4.6 Restrictions on Security Roles (FMT_SMR.2)

2.4.6.1 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 6.4.5 identifies the default administrative roles available on the web interface (both UI and REST API) and the default permissions associated with each role. This section also notes that additional roles can be defined. This section also states that any group with write privileges to the AAA menu are essentially system administrators as they have the ability to authorize themselves any additional privileges that they do not already have. This is a warning against potential violations of least privilege and is not a hard-coded restriction. The only restriction is that when such a user gives themselves new privileges, they must re-authenticate to the TOE before those privileges take effect.

[ST] section 6.4.5 also states that the local console interface has a single root user that functions as the Security Administrator for that interface, as opposed to the web interface, which has role separation.

2.4.6.2 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 FMT_SMR.2 section of [CCECG] includes information for how to access the console locally and how to access the UI/API both locally and remotely. This section makes it clear that while there is only a single root user for the console interface, UI/API users have role-based permissions, regardless of whether they're defined locally on the TOE or defined in LDAP/AD. This section goes on to list the default groups, the object that group membership allows access to, and the specific operations that are allowed for that group (e.g. the "auth-view" group has read permissions on AAA objects while the "auth-write" group has read and write/perform permissions on the same objects).

2.4.6.3 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.

This can be seen from the evidence in FCS_TLSS_EXT and throughout the test report where administrative actions were performed.

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 6.5.1 states that the TOE protects user passwords using a SHA-512 hash that uses a 12-byte salt, and that there is no administrative interface to view stored 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 6.5.2 states that there are no administrative interfaces to view stored key data in plaintext.

This section states that key data may be stored encrypted in the following cases:

- SSH proxy private keys may be stored encrypted in the credential store. Per [ST] section 6.2.3, a 256-bit AES key acts as a KEK for these, and the KEK may in turn be encrypted using a 256-bit key that is derived from a SHA-512 hash of an administrator-specified password.

- If the TOE's configuration data is exported, key data would be visible unless obfuscated. In the evaluated configuration, the configuration data is encrypted using a key that is derived from a password.

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 6.5.3 states that the TOE is a hardware appliance that has a hardware-based real-time clock that is manually configurable by an administrator. This section also identifies the specific uses of the clock (audit record time stamps, session activity for termination and lockout, and cryptographic operations based on time/date).

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.

The FPT_STM_EXT.1 section of [CCECG] references guidance in [Admin] for how to configure the system time. The TOE does not claim use of an NTP server, which is also noted in the guidance.

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 administrator's ability to set the time on the TOE was verified.

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 does not support use of an NTP server.

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 TOE consists of a single time source.

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 6.5.4 describes the TOE's software/firmware integrity and pseudorandom number generator (PRNG) self-tests. This section describes the integrity test as occurring at startup and validating the TOE's software/firmware against a known SHA-256 checksum. The PRNG test runs every ten seconds and checks for repeating patterns (i.e. a 'stuck' RNG). It is understood that this is sufficient to ensure the correct functionality of the TOE as these self-tests would detect any modification to security-relevant files or subsystems or any failure of an entropy source that would result in insecure 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 so this evaluation activity 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.

FPT_TST_EXT.1 section of [CCECG] identifies the errors that may result from failed self-tests and the actions that administrators should take in response to receiving such errors.

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

The TOE is not distributed so this evaluation activity is not applicable.

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 verified that the TOE carried out start-up self-tests.

The evaluator shall either verify that the self-tests described above are carried out during initial start-up or that the developer has justified any deviation from this.

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

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 6.5.5 states that the TOE can have up to five images loaded for delayed activation purposes, and includes instructions on how to view all loaded firmware images, including the active image.

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 6.5.5 states that software updates are acquired from the vendor support site. This site also includes the published SHA-1 hash for each update. Per [ST] section 6.5.5, the update is acquired outside the TOE (e.g. by an administrator downloading it to their workstation) and is verified outside of the TOE as well. The administrator uploads the update onto the TOE once the validation has been confirmed.

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.

FPT_TUD_EXT.1.2 does not include 'support automatic checking for updates' or 'support automatic updates'. Therefore, this assurance 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 so this evaluation activity 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.

[ST] section 6.5.5 states that a Security Administrator has the ability to upload an update to the TOE. The update is obtained and verified outside the TOE. Once the integrity of the update has been validated against the SHA-1 hash that is published on the site where the update was downloaded, the Security Administrator can upload the update to the TOE and initiate the update process.

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 FMT_MOF.1/ManualUpdate section of [CCECG] references [Admin] for instructions on how to view the currently executing version of the TOE software/firmware. This section also states that the TOE can have up to five software/firmware versions can be loaded at one time; all versions, whether actively running or not, can be queried in the same manner.

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.

The FMT_MOF.1/ManualUpdate section of [CCECG] describes a SHA-1 hash as being used as the upgrade integrity verification mechanism and identifies how the hash can be computed once the update has been obtained.

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 FMT_MOF.1/ManualUpdate section of [CCECG] states that the hash of an update is published on the Downloads page of the vendor support portal where the file itself can be downloaded.

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 so this evaluation activity 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 so this evaluation activity 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 for update verification so this evaluation 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 performed an updated on the TOE and verified that this was done successfully.

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.

N/A. TOE updates are verified with a published hash, not a 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.

N/A the TOE does not perform verification of the published hash.

If the verification of the hash value over the update file(s) against the published hash is not performed by the TOE, Test 3 shall be skipped.

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

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

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 6.6.1 states that remote administrative sessions are terminated after a configurable amount of time between 5 minutes and 720 minutes (12 hours), with a default of 10 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.

The FTA_SSL_EXT.1 and FTA_SSL.3 section of [CCECG] which references the timeout parameter as the variable used to configure idle session timeout for all administrative interfaces.

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 demonstrated that different inactivity timeout periods could be configured on the TOE and that those timeout periods were enforced.

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 6.6.2 states that a remote session over the Web UI is terminated with a logout option, and a remote session over the REST API has a delete operation for the active 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 FTA_SSL.4 section of [CCECG] identifies the mechanisms used to log out of the local console, Web UI, and REST API.

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 TOE uses the Ethernet management port with no networking infrastructure between the TOE hardware for local access. Hence, this testing is covered by FTA_SSL.4 Test 2.

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 demonstrated the ability to log on and log off from the TOE's web GUI.

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 6.6.3 states that local sessions on the web interface (both Web UI and REST API) and local console are terminated after a configurable period between 5 minutes and 720 minutes (12 hours), with a default of 10 minutes. The same configuration setting applies to both 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.

The FTA_SSL_EXT.1 and FTA_SSL.3 section of [CCECG] which references the timeout parameter as the variable used to configure idle session timeout for all administrative interfaces.

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 TOE uses the Ethernet management port and SSH with no networking infrastructure between the TOE hardware and SSH client for local access. Hence, this testing is covered by FTA_SSL.3.

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] 6.6.4 states that a configurable warning banner is displayed at both interactive administrative interfaces (web interface and local console) prior to login. The same banner text is used for both interfaces.

2.6.4.2 Guidance Activities

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

The FTA_TAB.1 section of [CCECG] describes how to configure the login banner text for both the web UI and local console. The same setting is used for both interfaces.

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 verified the ability to configure an access banner to the TOE.

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 6.7.1 identifies the following remote interfaces:

- Syslog server using TLS (TOE acts as client, non-TSF endpoint identified by X.509 certificate)
- LDAP/AD authentication server using TLS (TOE acts as client, non-TSF endpoint identified by X.509 certificate)
- SSH session proxying using SSH (TOE acts as both client and server, non-TSF endpoint identified by hostname/public key when acting as a client and claimed subject identity when acting as a server)

For the TLS client interfaces, this section also specifies that the use of TLS mutual authentication is optional.

This is consistent with the ST's claims of FCS_TLSC_EXT.1, FCS_TLSC_EXT.2 FCS_SSHC_EXT.1, and FCS_SSHS_EXT.1.

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.

[CCECG] section FTP_ITC_EXT.1 discusses establishment of all claimed trusted channels (TLS client to syslog server, TLS client to LDAP/AD server, SSH client/server for proxy connections). It also references a troubleshooting section in [Admin] for cases where connectivity issues are being experienced.

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 configured the TOE to communicate with a syslog server and verified that the connection was done over TLS. Evidence for this can be seen in FCS_TLSC_EXT.1.1 Test 1. In that test protected communications are established with a syslog server in order to verify the TOE's TLS functionality.

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.

The evaluator configured the TOE to communicate with a syslog server and verified that the connection was done over TLS. Evidence for this can be seen in FCS_TLSC_EXT.1.1 Test 1. In that test protected communications are established with a syslog server in order to verify the TOE's TLS functionality.

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

The evaluator configured the TOE to communicate with a syslog server and verified that the connection was done over TLS. Evidence for this can be seen in FCS_TLSC_EXT.1.1 Test 1. In that test protected communications are established with a syslog server in order to verify the TOE's TLS functionality. The wire captures obtained for that test show encrypted Application Data packets being sent once the TLS handshake is complete.

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 interrupted the physical connection to the TOE twice, once long enough to timeout the application layer and once shorter than the application layer timeout. In both instances the TOE maintained a protected communications channel.

Further assurance activities are associated with the specific protocols.

[Test Notes]

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.

[Test Notes]

The developer shall provide to the evaluator application layer configuration settings for all secure communication mechanisms specified by the FTP_ITC.1 requirement. This information should be sufficiently detailed to allow the evaluator to determine the application layer timeout settings for each cryptographic protocol. There is no expectation that this information must be recorded in any public-facing document or report.

[Test Notes]

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 6.7.2 states that remote administration uses HTTPS with TLS 1.1 or 1.2. This is consistent with the ST's claim of FCS_TLSS_EXT.1.

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 FTP_TRP.1/Admin section of [CCECG] provides instructions on how to access the UI and REST API, as well as guidance on ensuring that TLS is enabled and correctly configured for these interfaces.

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.

Evidence for this can be seen from the FCS_TLSS_EXT.1.1 Test 1. That test required the evaluator to establish a secure connection to the TOE's remote administrative interface.

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

Evidence for this can be seen from the FCS_TLSS_EXT.1.1 Test 1. That test required the evaluator to establish a secure connection to the TOE's remote administrative interface. The wire captures generated as part of that test show encrypted Application Data packets being sent after the TLS handshake.

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.

3 Security Assurance Requirements

3.1 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).

3.1.1 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.

The evaluator shall examine the TSS to determine that it is clear which TOE components contribute to each SFR or how the components combine to meet each SFR.

The TOE is not distributed so this evaluation activity is not applicable.

The evaluator shall verify the sufficiency to fulfil the related SFRs. This includes checking that the TOE as a whole fully covers all SFRs and that all functionality that is required to be audited is in fact audited regardless of the component that carries it out.

The TOE is not distributed so this evaluation activity is not applicable.

Additional Evaluation Activities for the TSS in the case of a distributed TOE as defined in section A.9.1.1 in the SD.

The evaluator shall examine the TSS to identify any extra instances of TOE components allowed in the ST and shall examine the description of how the additional components maintain the SFRs to confirm that it is consistent with the role that the component plays in the evaluated configuration. For example: the secure channels used by the extra component for intra-TOE communications (FPT_ITT) and external communications (FTP_ITC) must be consistent, the audit information generated by the extra component must be maintained, and the management of the extra component must be consistent with that used for the original instance of the component in the minimum configuration.

The TOE is not distributed so this evaluation activity is not applicable.

3.2 Class ADV: Development

3.2.1 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.

3.2.1.1 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:

- Web UI
- REST API
- Local console
- Syslog interface
- LDAP/AD interface
- SSH proxy client and server interfaces

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.

3.2.1.2 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.

Through completion of the evaluation activities for FAU_STG_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, and FCS_TLSS_EXT.1, the evaluator verified that security-relevant configuration instructions are present for all TOE external interfaces. In particular, the security-relevant parameters are those that are used to ensure that the data in transit is protected in a manner that is consistent with the claimed SFRs.

3.2.1.3 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 evaluator reviewed [ST] and [CCECG] and determined that the following mappings were evident from the documentation:

- Web UI:
  - FAU_GEN.1
  - FAU_GEN.2
  - FCS_HTTPS_EXT.1
  - FCS_TLSS_EXT.1
  - FIA_AFL.1
  - FIA_UAU_EXT.2
  - FIA_UIA_EXT.1
  - FMT_MOF.1/ManualUpdate
  - FMT_MTD.1/CoreData
  - FMT_MTD.1/CryptoKeys
  - FMT_SMF.1
  - FMT_SMR.2
  - FPT_STM_EXT.1
  - FPT_TUD_EXT.1

    - FTA_SSL.3
    - FTA_SSL.4
    - FTA_TAB.1
    - FPT_TRP.1/Admin
- REST API:
    - FAU_GEN.1
    - FAU_GEN.2
    - FCS_HTTPS_EXT.1
    - FCS_TLSS_EXT.1
    - FIA_AFL.1
    - FIA_UAU_EXT.2
    - FIA_UIA_EXT.1
    - FMT_MOF.1/ManualUpdate
    - FMT_MTD.1/CoreData
    - FMT_MTD.1/CryptoKeys
    - FMT_SMF.1
    - FMT_SMR.2
    - FPT_STM_EXT.1
    - FTP_TRP.1/Admin
- Local console:
    - FAU_GEN.1
    - FAU_GEN.2
    - FIA_AFL.1
    - FIA_UAU_EXT.2
    - FIA_UIA_EXT.1
    - FMT_MTD.1/CoreData
    - FMT_SMF.1
    - FMT_SMR.2
    - FTA_SSL_EXT.1
    - FTA_SSL.4
    - FTA_TAB.1
    - FTP_TRP.1/Admin
- Syslog interface:
    - FAU_STG_EXT.1
    - FCS_TLSC_EXT.1
    - FCS_TLSC_EXT.2
    - FTP_ITC.1
- LDAP/AD interface:
    - FCS_TLSC_EXT.1
    - FCS_TLSC_EXT.2
    - FIA_UAU_EXT.1
    - FTP_ITC.1
- SSH proxy client and server interfaces:
    - FCS_SSHC_EXT.1
    - FCS_SSHS_EXT.1
    - FTP_ITC.1

In each case, the security relevance of the external interfaces are clear, and there are no cases where SFR-

related functionality is missing a relevant external interface

3.3 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.

Additional Evaluation Activities for the guidance documentation in the case of a distributed TOE as defined in section A.9.2 in the SD.

The evaluator shall examine the description of the extra instances of TOE components in the guidance documentation to confirm that they are consistent with those identified as allowed in the ST. This includes confirmation that the result of applying the guidance documentation to configure the extra component will leave the TOE in a state such that the claims for SFR support in each component are as described in the ST and therefore that all SFRs continue to be met when the extra components are present.

The TOE is not distributed so this evaluation activity is N/A.

Additional Evaluation Activities for the guidance documentation in the case of a distributed TOE as defined in section A.9.2 in the SD.

The evaluator shall examine the secure communications described for the extra components to confirm that they are the same as described for the components in the minimum configuration (additional connections between allowed extra components and the components in the minimum configuration are allowed of course).

The TOE is not distributed so this evaluation activity is N/A.

3.3.1 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.

3.3.1.1 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 of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

The [CCECG] 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. [CCECG] additionally references existing vendor documentation ([Install], [REST], [Admin], [Packaging], [IPMI], [3000], and [3500]) to make it clear to the reader what already-existent vendor documentation exists and which parts of it must be considered when placing the TOE into its evaluated configuration.

The vendor documentation is available at https://support.oneidentity.com/one-identity-safeguard-for-privileged-sessions/. This is referenced prominently in the Product Support section of [CCECG].

3.3.1.2 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 ST claims a single Operational Environment, comprising the two supported models of the standalone TOE appliance included in the scope of evaluation (3000 and 3500). The guidance in [CCECG] and referenced documents adequately addresses the single TOE appliance claimed in the ST. Specifically, [Packaging] references [3000] and [3500] for documentation about physical setup for the TOE appliance, which is the Supermicro SC113 Chassis for the 3000 model and the Supermicro SuperServer 1029U-T for the 3500 model. The Initial Configuration section of [CCECG] references this documentation as well.

[Install] references a number of physical and virtual models; [CCECG] makes it clear to the reader that only the 3000 and 3500 appliance in standalone configuration (i.e. not HA) are within the evaluation scope in the Scope of the Evaluation section.

The vendor guidance for version 6.9.3 of the TOE is published on the vendor site; as of the creation of this report, version 6.9.4 is available as a release candidate, and will be general availability by the publication of this report. The evaluator reviewed [Release] and observed the following differences between the two versions:

- Custom trust stores are allowed for LDAP – this is already covered in the existing vendor guidance when configuring LDAP trust stores, the only difference is that the administrator will be presented with a larger list of trust stores to select if any custom ones have been configured.
- Syslog-ng client supports full certificate revocation – this corrects a product flaw that was found during the evaluation when testing version 6.9.3 of the product. Since it corrects an existing implementation there is no effect on the product guidance.
- Elliptic curve private keys can be generated using the REST API – this functionality is not present in [REST] but is documented in the FIA_X509_EXT.3 section of [CCECG].
- Several features are deprecated; none of these apply to the TSF.

3.3.1.3 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 Initial Configuration section of [CCECG] notes that the only cryptographic engine used by the product is OpenSSL 1.1.1 and that no special configuration is needed to ensure that this is enabled. This section also notes, however, that the algorithms used by this engine must be limited to those that are in the evaluated configuration.

3.3.1.4 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.

Section “About this Guide” of [CCECG] states the guidance provided in [CCECG] allows the administrator to deploy the TOE in an environment consistent with the TOE’s evaluated configuration and provides the administrator with instructions for exercising the security functions that were claimed as part of the CC evaluation. Section “Scope of the Evaluation” identifies product features and protocols that were not evaluated and that must be disabled in the evaluated configuration, along with rationale for doing so.

3.3.1.5 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 in section 3.3.1.3 above.

Part b) is addressed in section 2.5.5.2 above.

Part c) is addressed in section 3.3.1.4 above.

3.3.2 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.

3.3.2.1 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).

Section "Evaluation Assumptions" of [CCECG] describes assumptions about the intended operational environment and method of use of the TOE, sufficient for an administrator to verify the operational environment can fulfil its role to support the evaluated security functionality.

3.3.2.2 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.

The ST claims a single Operational Environment, comprising a standalone TOE appliance included in the scope of evaluation, either the 3000 or 3500 model. The guidance in [CCECG], [Install], and [Packaging] adequately addresses the TOE appliances claimed in the ST. Specifically, the non-CC documentation includes reference to other appliances, including virtual appliances and a high availability configuration, but [CCECG] makes it clear that only the 3000 and 3500 models are within the evaluation scope.

3.3.2.3 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 Initial Configuration section of [CCECG] includes step-by-step guidance on initial physical deployment and first-time setup of the TOE, referencing external documentation as needed.

3.3.2.4 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] includes guidance for configuration of the TOE as well as environmental components as needed to ensure all external interfaces are consistent

3.3.2.5 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] includes guidance on how to provide a protected administrative capability by ensuring protected physical access for local administration and by enabling HTTPS and configuring TLS for remote

administrative capability. This guide also notes that SSH for the console is disabled by default and this should not be re-enabled for the TOE to be in its evaluated configuration.

[CCECG] identifies the default IPMI and BIOS passwords and references [Install] for guidance on how to change them. [CCECG] also identifies the default console root user and references [Admin] for guidance on how to change it during first use. The UI is accessed for the first time using the Welcome Wizard, which does not have a password; the initial administrator password for that interface is specified on first use.

3.4 Class ALC: Life-Cycle Support

3.4.1 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.

3.4.2 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.

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.

The test activities were performed in accordance with [Test] and in all cases demonstrated that the SFR claims specified in [ST] were implemented as described.

Two TOE models were claimed in section 1.1 of the ST: a model 3000 and a model 3500 device. Both devices have identical software/firmware including hardware interface drivers. The only differences between the models are as follows:

- Processor
  - 3000: Intel Xeon E3-1275 v6 (Kaby Lake)
  - 3500: 2x Intel Xeon Silver 4110 (Skylake)
- Network interfaces
  - 3000: 2x RJ45 GbE Ethernet, 4x 1G Base-T Ethernet
  - 3500: 2x RJ45 10GbE Ethernet, 2x 1G Base-T Ethernet
- Memory
  - 3000: 32GB
  - 3500: 64GB
- Storage
  - 3000: 6TB

- 3500: 12TB

Section A.7.2 of the Supporting Document – Evaluation Activities for Network Device cPP lists the potential product differences affecting equivalency as Platform/Hardware Dependencies, Differences in TOE Software Binaries, Differences in Libraries Used to Provide TOE Functionality, TOE Management Interface Differences, and TOE Functional Differences. These equivalencies were assessed as follows:

Platform/Hardware Dependencies: The two processors used by the two TOE models are not equivalent because of microarchitecture differences, which could impact their cryptographic implementation. NIST algorithm testing was obtained for both models and no equivalency is therefore argued for those functions (FCS_CKM.1, FCS_CKM.2, FCS_COP.1/*, FCS_RBG_EXT.1). All other functional requirements are implemented purely in software and the processor architecture is considered to be equivalent because both product models use the same x86_64 kernel. While processor microarchitecture may introduce differences in how cryptographic computations are made, it would not affect other security-relevant functionality such as the content and format of audit events, the ability to authenticate administrators and reject invalid authentication attempts, or the higher-level implementation of transport/application layer protocols (TLS, SSH, HTTPS) that rely on the correct functioning of low-level cryptography.

Per the Supporting Document, re-testing is only necessary for the functionality that is dependent on platform/hardware-provided functionality. Therefore, the CCTL chose to re-test the functionality that is validated through NIST algorithm testing. All other hardware differences (number and speed of network ports, amount of memory, and amount of disk storage) are performance differences that do not directly relate to any SFR-enforcing functionality, and are therefore considered to be equivalent.

Differences in TOE Software Binaries: The product does not use dynamic driver loading in the kernel so all code needed to interface with the underlying hardware for each model is present in both models. The kernel binary loaded into each device is identical and therefore equivalent in this regard.

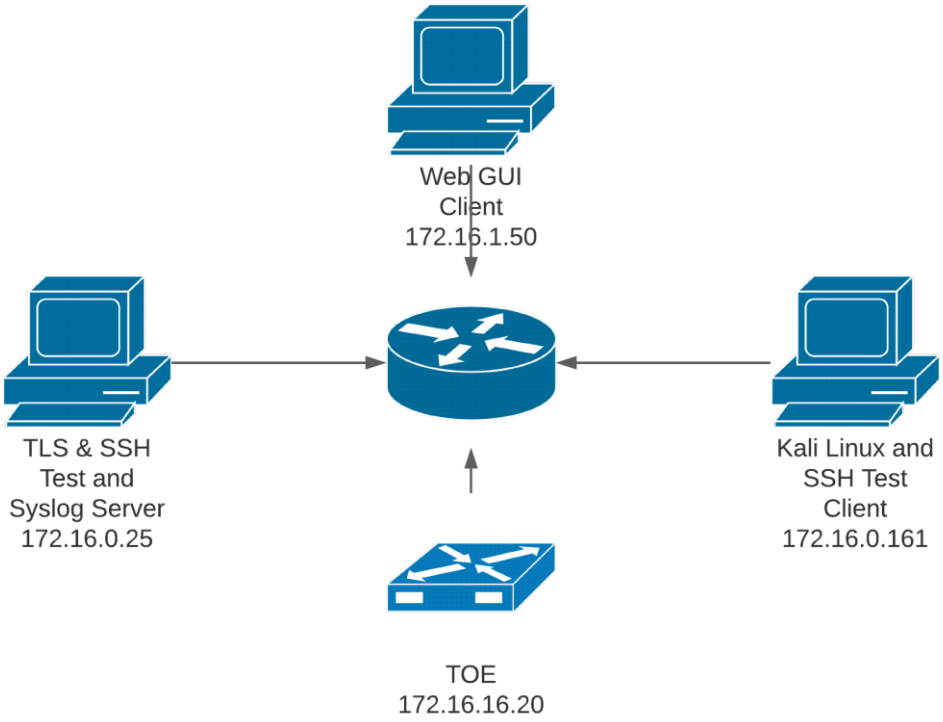
Differences in Libraries Used to Provide TOE Functionality: Because the software binaries are identical between models, the libraries included for each are identical as well. There are no cases where one model has one library that the other lacks, or where the two models use different versions of the same library. They are therefore equivalent in this regard.

TOE Management Interface Differences: Both TOE models offer the same local and remote management interfaces with no difference between them with regards to functionality differences, the method by which they are accessed, the privilege model that determines the actions available to a given administrator, or any other differences. They are therefore equivalent in this regard.

TOE Functional Differences: Both TOE models make identical security functional claims with respect to the claimed PP. The Security Target does not identify any functional behavior that is claimed on only one model, or where the same function is implemented in two different ways on the different models. They are therefore equivalent in this regard.

Based on this, the sampling approach taken by the CCTL was to perform full cryptographic algorithm testing (through NIST ACVP per NIAP Policy #5) on both TOE models, while considering it sufficient to perform the remaining tests on a single model given the equivalency between both models on all other functions.

The test environment was configured in the manner depicted below:



The non-TOE systems included the following software and tools that were relevant to execution of the testing:

Web GUI Client

Microsoft Windows 2016 Server Datacenter

Google Chrome Version 97

Putty release 0.71

TLS Test and Syslog Server

Ubuntu 18.04

Wireshark 2.6.10

SSLyze 2.1.4

OpenSSL 1.1.1

Python

Leidos Proprietary TLS tools

Kali Linux and SSH Test Client

Kali Linux Release 2020.4

Wireshark 2.6.10

XCA Certificate Authority 2.0.1

SSLyze 2.1.4

Apache Web Server 2.4.4.1 (for CRL distribution)

3.5.1.1 Evaluation Activity

Note that these 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 evaluator tests the TOE in the minimum configuration as defined in the ST (and the guidance documentation).

The TOE is not distributed so these evaluation activities are not applicable.

If the description of the use of extra components in the ST and guidance documentation identifies any difference in the SFRs allocated to a component, or the scope of the SFRs involved (e.g. if different selections apply to different instances of the component) then the evaluator tests these additional SFR cases that were not included in the minimum configuration.

In addition, the evaluator tests the following aspects for each extra component that is identified as allowed in the distributed TOE:

• Communications: the evaluator follows the guidance documentation to confirm, by testing, that any additional connections introduced with the extra component and not present in the minimum configuration are consistent with the requirements stated in the ST (e.g. with regard to protocols and ciphersuites used). An example of such an additional connection would be if a single instance of the component is present in the minimum configuration and adding a duplicate component then introduces an extra communication between the two instances. Another example might be if the use of the additional components necessitated the use of a connection to an external authentication server instead of using locally stored credentials.

• Audit: the evaluator confirms that the audit records from different instances of a component can be distinguished so that it is clear which instance generated the record.

• Management: if the extra component manages other components in the distributed TOE then the evaluator shall follow the guidance documentation to confirm that management via the extra component uses the same roles and role holders for administrators as for the component in the minimum configuration.

The TOE is not distributed so these evaluation 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. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

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

3.6.1.1 AVA_VAN.1 Evaluation Activity (Documentation)

Modified by TD0547:

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

The evaluator shall examine the documentation outlined below provided by the developer to confirm that it contains all required information. This documentation is in addition to the documentation already required to be supplied in response to the EAs listed previously.

The developer shall provide documentation identifying the list of software and hardware components[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) such as a web server and protocol or cryptographic libraries (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components, and will be used by the evaluators in formulating 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]

[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.

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.

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 applied each AVA_VAN.1 CEM work unit. The evaluation team performed a vulnerability analysis following the processes described in the claimed PP. This comprised a search of public vulnerability databases.

Searches of public vulnerability repositories were performed on 3 March 2022.

The evaluation team searched the following public vulnerability repositories.

- National Vulnerability Database (http://web.nvd.nist.gov/view/vuln/search)
- US-CERT Vulnerability Notes Database (https://www.kb.cert.org/vuls/)
- The vendor's notifications and alerts page for the product (https://support.oneidentity.com/one-identity-safeguard-for-privileged-sessions/all/alerts-notifications)
- Ubuntu Security Notices (for the Ubuntu 18.04 LTS operating system that the TOE relies on) (https://ubuntu.com/security/notices?order=newest&release=bionic&details=)
- Ubuntu CVEs (for the Ubuntu 18.04 LTS operating system that the TOE relies on) (https://ubuntu.com/security/cves)

The evaluation team used the following search terms in the searches of these repositories:

- one identity/one identity safeguard/safeguard for privileged sessions/balabit (product name, current developer, former developer)
- intel/xeon/kaby lake/skylake (processor make and model)
- sc113/superserver/1029u-t/supermicro (hardware chassis make and model)
- ubuntu 18.04 LTS (TOE base operating system)
- nginx (TOE web server)
- openssl (TOE TLS and general cryptographic engine)
- postgresql/rabbitmq/cherrypy/openldap/freerdp/log4j/syslog-ng/sudo/angular2 (significant third-party software components contained within the TOE software)

The results of these searches did not identify unmitigated vulnerabilities that are applicable to the TOE. The conclusion drawn from the vulnerability analysis is that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential as defined by the Certification Body in accordance with the guidance in the CEM.

The TOE is not distributed so the requirements that relate specifically to distributed TOEs do not apply.