www.GossamerSec.com

# Assurance Activity Report for Cisco Network Convergence System 1004 (NCS1004) running IOS-XR 24.1

Version 0.4
03/04/25

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

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

Document: AAR-11508

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

## Revision History

| Revision | Date | Authors | Summary |
|---|---|---|---|
| Version 0.1 | 02/07/25 | Gossamer | Initial draft |
| Version 0.2 | 2/11/25 | Gossamer | Response to validator comments |
| Version 0.3 | 2/27/25 | Gossamer | Response to validator comments |
| Version 0.4 | 3/4/25 | Gossamer | Response to validator comments |
| | | | |
| | | | |
| | | | |

**The TOE Evaluation was Sponsored by**:

Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134

**Evaluation Personnel**:

- Ryan Hagedorn
- Cody Cummins

**Common Criteria Versions**:

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

**Common Evaluation Methodology Versions**:

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

## Table of Contents

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

# 1. Introduction

This document presents evaluations results of the Cisco Network Convergence System 1004 (NCS1004) Running IOS-XR 24.1 NDcPP22e evaluation. This document contains a description of the assurance activities and associated results as performed by the evaluators.

## 1.1 CAVP Certificate Justification

The TOE is the Cisco Network Convergence System 1004 (NCS1004) Running IOS-XR 24.1. The TOE includes the Cisco FIPS Object Module 7.3a which resides in the IOS-XR software

| Algorithm | Description | Supported Mode | CAVP Cert. # | Module | SFR |
|---|---|---|---|---|---|
| AES | Used for symmetric encryption/decryption | CBC, CTR, GCM (128, 256) | A4446 | FOM 7.3a | FCS_COP.1/DataEncryption |
| SHS (SHA-1, SHA-256 and SHA-512) | Cryptographic hashing services | Byte Oriented | A4446 | FOM 7.3a | FCS_COP.1/Hash |
| HMAC (HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-512) | Keyed hashing services and software integrity test | Byte Oriented | A4446 | FOM 7.3a | FCS_COP.1/KeyedHash |
| DRBG | Deterministic random bit generation services in accordance with ISO/IEC 18031:2011 | HMAC_DRBG (256) | A4446 | FOM 7.3a | FCS_RBG_EXT |
| RSA | Signature Verification and key transport | FIPS PUB 186-4 Key Generation<br><br>PKCS #1 v1.5 2048, 3072, and 4096 bit keys | A4446 | FOM 7.3a | FCS_CKM.1<br><br>FCS_CKM.2<br><br>FCS_COP.1/SigGen |
| FFC | Signature Verification and key transport | NIST Special Publication 800-56A | Tested with a known good | FOM 7.3a | FCS_CKM.1<br><br>FCS_CKM.2 |

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

| Algorith m | Description | Supported Mode | CAVP Cert. # | Modul e | SFR |
|---|---|---|---|---|---|
| | | Revision 3, Safe-Primes | implementatio n | | |

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

# 2. Protection Profile SFR Assurance Activities

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

## 2.1 Security audit (FAU)

### 2.1.1 Audit Data Generation (NDcPP22e:FAU_GEN.1)

#### 2.1.1.1 NDcPP22e:FAU_GEN.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.1.1.2 NDcPP22e:FAU_GEN.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

Section 6.1, Table 18 (FAU_GEN.1) in the [ST] indicates that the audit log content includes the type of event that occurred such as generating keys, including the type of key(rsa) and the label of the key.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

The Section "Audit Records Description" in [Admin Guide] lists all of the required auditable events identified in the ST. The TOE generates an audit record whenever an auditable event occurs. The types of events that cause audit records to be generated include cryptography related events, identification and authentication related events, and administrative events (the specific events and the contents of each audit record are listed in Table 11: Audit Events and Sample Record). Each of the events is specified in syslog records in enough detail to identify the user for which the event is associate, when the event occurred, the outcome of the event, and the type of event that occurred. During testing, the evaluator mapped the audit event examples in the [Admin Guide] Table 11 to the TOE generated events and confirmed that the [Admin Guide] includes examples/descriptions of all required audit events.

The evaluator verified the administrative commands when performing all other guidance AAs. Specific references to commands can be found throughout this AAR.

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

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

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

The evaluator created a list of the required audit events and collected these audit events when running the other security functional tests described by the protection profiles. The evaluator then recorded these audit events in the proprietary Detailed Test Report (DTR). The security management events are handled in a similar manner. The evaluator collected the required audits for all TOE components, via syslog, during the course of testing. When the administrator was required to set a value for testing, the audit record associated with the administrator action was collected and recorded in the DTR. The audit events that were collected matched the format specified in the [Admin Guide].

### 2.1.2 User identity association (NDcPP22e:FAU_GEN.2)

#### 2.1.2.1 NDcPP22e:FAU_GEN.2.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

See FAU_GEN.1

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

See FAU_GEN.1

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

For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one test on one component where another component instigates an auditable event. The

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

See FAU_GEN.1

## 2.1.3 Protected Audit Event Storage (NDcPP22e:FAU_STG_EXT.1)

### 2.1.3.1 NDcPP22e:FAU_STG_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.1.3.2 NDcPP22e:FAU_STG_EXT.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.1.3.3 NDcPP22e:FAU_STG_EXT.1.3

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

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

Section 6.1, Table 18 (FAU_STG_EXT.1) in the [ST] states that the TOE is configured to export syslog records to a specified, external syslog server in real time. The TOE protects communications with an external syslog server via TLS. The TOE transmits its audit events to all configured syslog servers at the same time logs are written to the local log buffer and to the console. The TOE also stores a limited set of audit records locally in a circular file on the TOE, and continues to do so if the communication with the syslog server goes down. The default value for the size of the logging buffer on the TOE is 2097152 bytes. If the TLS connection fails, the TOE will buffer a small amount of audit records on the TOE when it discovers it can no longer communicate with its configured syslog server, and will transmit the buffer contents when connectivity to the syslog server is restored. When the local audit data storage is full, the TOE will overwrite the oldest stored audit records when writing new audit records. The local audit records are stored in a directory that does not allow administrators to modify the contents.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

Section “Security Relevant Events” in the [Admin Guide] states that the TOE is able to generate audit records that are stored internally within the TOE whenever an audited event occurs, as well as simultaneously offloaded to an external syslog server. The local log buffer is circular. Newer messages overwrite older messages after the buffer is full. The first message displayed is the oldest message in the buffer. When configured for a syslog backup the TOE will simultaneously offload events from a separate buffer to the external syslog server. This buffer is used to queue events to be sent to the syslog server if the connection to the server is lost. It is a circular buffer, so when the events overrun the storage space overwrites older events.

Section “Logging Protection” in the [Admin Guide] states that in order to protect against audit data loss, the TOE must be configured to send the audit records securely (through TLS) to an external Secure Syslog Server which is expected to be the current syslog version updated with the latest patches.  By default, system messages are logged to the console and the logfile, for the evaluated configuration the severity level must be set to “debugging” to ensure all required audit events related to the TOE Security functions are audited.

Section “Logging Configuration” in the [Admin Guide] indicates that the logging must be enabled for all commands executed and provides instructions for doing so.

Section “Turn Logging on/off” in the [Admin Guide] describes how to enable and disable configuration logging.

Section “Set Logging Size” in the [Admin Guide] provides instructions for configuring the log file size and states that the default and required value for the size of the logging buffer on the TOE is 2097152 bytes.

Section “X.509 Certificates” in the [Admin Guide] provides instructions for setting up secure syslog such that the remote server will be authenticated via a trustpoint configuration. This includes declaring a CA, configuring a trustpoint and issuing a certificate request.

Section “Logging to Syslog Server via TLS” in the [Admin Guide] provides the steps for configuring TLS for communications between the TOE and the syslog server.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

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

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

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

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

Test 1: The external audit server was utilizing rsyslogd version 8.16.0. The evaluator initiated a packet capture collecting traffic between the TOE and the syslog server and verified all data was encrypted and that the TOE

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

initiated the syslog connection without admin intervention. The evaluator then verified audit information was generated and transferred to the syslog server.

Test 2: The TOE consists of a single standalone component that stores audit data locally. The evaluator verified that when the local audit storage was filled, the existing audit data was overwritten using the following rule: Overwrite oldest records first.

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

Test 4: Not applicable. The TOE is not distributed.

## 2.2 Cryptographic support (FCS)

### 2.2.1 Cryptographic Key Generation (NDcPP22e:FCS_CKM.1)

#### 2.2.1.1 NDcPP22e:FCS_CKM.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FCS_CKM.1) in the [ST] states that the TOE generates asymmetric keys in accordance with the RSA schemes using key sizes of 2048-bit or greater that conform to the FIPS PUB 186-4, Appendix B.3. The TOE can create an RSA public-private key pair that can be used to generate a Certificate Signing Request (CSR). The TOE acts as both a sender and receiver for RSA -based key establishment schemes. The RSA key establishment meets the RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. The TOE implements Diffie-Hellman (DH) group 14 (2048) bit key finite field-based key generation in conformance with RFC 3526 section 3 and RFC 7919. The TOE acts as both a sender and receiver for Diffie-Helman based key establishment schemes. The DH key establishment meets the RFC 3526, Section 3 and RFC 7919.

Section 6.1, Table 18 (FCS_CKM.1) provides a table mapping the RSA and FFC (DH) schemes to corresponding SFRs and services. RSA schemes are used for SSH and TLS key establishment and for SSH remote administration. FFC Schemes using DH are used for key exchange in SSH remote administration and in TLS which is used for communications with an external syslog server.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

Section “X.509 certificates” in the [Admin Guide] describes configuring the rsa keypair in the certificate request.

Section “Logging Protection” in the [Admin Guide] indicates that the supported TLS ciphersuites are available by default in FIPS mode.

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

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

Key Generation for FIPS PUB 186-4 RSA Schemes

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

a) Random Primes:

- Provable primes

- Probable primes

b) Primes with Conditions:

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

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

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

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

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

FIPS 186-4 Public Key Verification (PKV) Test

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Key Generation for Finite-Field Cryptography (FFC)

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

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

- Primes q and p shall both be provable primes

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

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process

- Generator g constructed through an unverifiable process.

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

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

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

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

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

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

- g != 0,1

- q divides p-1

- g^q mod p = 1

- g^x mod p = y

for each FFC parameter set and key pair.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

## 2.2.2 Cryptographic Key Establishment (NDcPP22e:FCS_CKM.2)

### 2.2.2.1 NDcPP22e:FCS_CKM.2.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

```
Scheme          |          SFR          |        Service

------------------------------------------------------------------------------------------

RSA             | FCS_TLSS_EXT.1  | Administration

------------------------------------------------------------------------------------------

ECDH            | FCS_SSHC_EXT.1  | Audit Server

------------------------------------------------------------------------------------------

ECDH            | FCS_IPSEC_EXT.1 | Authentication Server

------------------------------------------------------------------------------------------
```

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

(TD0580 applied)

Section 6.1, Table 18 (FCS_CKM.1) in the [ST] states that the TOE generates asymmetric keys in accordance with the RSA schemes using key sizes of 2048-bit or greater that conform to the FIPS PUB 186-4, Appendix B.3. The TOE can create an RSA public-private key pair that can be used to generate a Certificate Signing Request (CSR). The TOE acts as both a sender and receiver for RSA -based key establishment schemes. The RSA key establishment meets the RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1. The TOE implements Diffie-Hellman (DH) group 14 (2048) bit key finite field-based key generation in conformance with RFC 3526 section 3. The TOE acts as both a sender and receiver for Diffie-Helman based key establishment schemes. The DH key establishment meets the RFC 3526, Section 3.

Section 6.1, Table 18 (FCS_CKM.1) provides a table mapping the RSA and FFC (DH) schemes to corresponding SFRs and services. RSA schemes are used for SSH and TLS key establishment and for SSH remote administration. FFC Schemes using DH are used for key exchange in SSH remote administration and in TLS which is used for communications with an external syslog server.

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

- Key Exchange Algorithms: diffie-hellman-group14-sha1

Section “X.509 certificates” in the [Admin Guide] describes configuring the rsa keypair in the certificate request.

Section “Logging Protection” in the [Admin Guide] indicates that the supported TLS ciphersuites are available by default in FIPS mode.

**Component Testing Assurance Activities**: Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Validity Test

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

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

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

RSA-based key establishment

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

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

The TOE has been CAVP tested.  Refer to the CAVP certificates identified in Section 1.1.

### 2.2.3 Cryptographic Key Destruction (NDcPP22e:FCS_CKM.4)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

#### 2.2.3.1 NDcPP22e:FCS_CKM.4.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section 6.1, Table 18 (FCS_CKM.4) in the [ST] states that the TOE meets all requirements specified in FIPS 140-2 for destruction of keys and Critical Security Parameters (CSPs) in that none of the symmetric keys, pre-shared keys, or private keys are stored in plaintext.

Annex A, Table 19 describes the key zeroization and identifies all relevant keys, including the storage location of each, all relevant key destruction, and the destruction method used in each case (including the command/function used by the TOE to destroy the keys).

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

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

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

**Component Testing Assurance Activities**: None Defined

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

#### 2.2.4.1 NDcPP22e:FCS_COP.1.1/DataEncryption

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section 6.1, Table 18 (FCS_COP.1/DataEncryption) in [ST] states that the TOE implements AES in the following protocols: SSH and TLS. The TOE provides symmetric encryption and decryption using AES in CBC, CTR, and GCM mode (128, 256 bits) as described in ISO 18033-3 and ISO 10116 and ISO 19772.

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

The section "Steps to configure SSH server on the router" in the [Admin Guide] provides detail on how to configure the TOE to only accept SSHv2 client connections.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

The section "Logging Protection" in the [Admin Guide] states that the TOEs TLS implementation has support for the following ciphers that are available by default in FIPS mode.

TLS_RSA_WITH_AES_128_CBC_SHA

TLS_RSA_WITH_AES_256_CBC_SHA

**Component Testing Assurance Activities**: AES-CBC Known Answer Tests

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

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

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

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

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

AES-CBC Multi-Block Message Test

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

AES-CBC Monte Carlo Tests

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

```
# Input: PT, IV, Key

for i = 1 to 1000:

if i == 1:

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

PT = IV

else:

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

PT = CT[i-1]
```

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

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

AES-GCM Test

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

128 bit and 256 bit keys

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

AES-CTR Known Answer Tests

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

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

AES-CTR Multi-Block Message Test

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

AES-CTR Monte-Carlo Test

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

# Input: PT, Key

for i = 1 to 1000:

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

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

There is no need to test the decryption engine.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

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

### 2.2.5.1 NDcPP22e:FCS_COP.1.1/Hash

**TSS Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FCS_COP.1/Hash) in the [ST] states that the TOE provides cryptographic hashing services using SHA-1, SHA-256, and SHA-512 as specified in ISO/IEC 10118-3:2004. Through the implementation of the CAVP validated cryptographic module, the TOE provides Secure Hash Standard (SHS) hashing in support of SSH and TLS for secure communications.

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

Section "Enabling FIPS Mode" in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section "Remote Administration Protocols" in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

Section "X.509 certificates" in the [Admin Guide] describes configuring the rsa keypair in the certificate request.

Section "Logging Protection" in the [Admin Guide] indicates that the supported TLS ciphersuites are available by default in FIPS mode.

**Component Testing Assurance Activities**: The TSF hashing functions can be implemented in one of two modes. The first mode is the byte-oriented mode. In this mode the TSF only hashes messages that are an integral number of bytes in length; i.e., the length (in bits) of the message to be hashed is divisible by 8. The second mode is the bit-

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

oriented mode. In this mode the TSF hashes messages of arbitrary length. As there are different tests for each mode, an indication is given in the following sections for the bit-oriented vs. the byte-oriented testmacs.

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

Short Messages Test - Bit-oriented Mode

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

Short Messages Test - Byte-oriented Mode

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

Selected Long Messages Test - Bit-oriented Mode

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

Selected Long Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8 messages, where m is the block length of the hash algorithm (e.g. 512 bits for SHA-256). The length of the ith message is m + 8*99*i, where 1 <= i <= m/8. The message text shall be pseudorandomly generated. The evaluators compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Pseudorandomly Generated Messages Test

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

The TOE has been CAVP tested.  Refer to the CAVP certificates identified in Section 1.1.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

#### 2.2.6.1 NDcPP22e:FCS_COP.1.1/KeyedHash

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FCS_COP.1/KeyedHash) in the [ST] states that the TOE provides keyed-hashing message authentication services using HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-512, key size 160, 256, 512 bits, and message digest sizes 160, 256, 512 as specified in ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

Through the implementation of the CAVP validated cryptographic module, the TOE provides SHS hashing and HMAC message authentication in support of SSHv2, TLSv1.1 and TLSv1.2 for secure communications. SHS hashing and HMAC message authentication (SHA-1) is used in the establishment of TLS and SSHv2 sessions.

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

Section "Enabling FIPS Mode" in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section "Remote Administration Protocols" in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

Section “X.509 certificates” in the [Admin Guide] describes configuring the rsa keypair in the certificate request.

Section “Logging Protection” in the [Admin Guide] indicates that the supported TLS ciphersuites are available by default in FIPS mode.

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

The TOE has been CAVP tested.  Refer to the CAVP certificates identified in Section 1.1.

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

#### 2.2.7.1 NDcPP22e:FCS_COP.1.1/SigGen

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

The TOE has been CAVP tested.  Refer to the CAVP certificates identified in Section 1.1

Table 18 in the [ST] states that through the implementation of the CAVP validated cryptographic module, the TOE pro-vides cryptographic signatures in support of SSH and TLS for secure communications.  Management of the cryptographic algorithms is provided through the CLI with auditing of those commands.  The TOE provides the RSA option in support of SSH and TLS key establishment.  RSA (2048-bit, 3072-bit, and 4096-bit) is used in the establishment of SSHv2 key establishment.  For SSH, RSA host keys are supported.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

The section entitled "Enabling FIPS Mode" in the [Admin Guide] provides instructions on how to run FIPS mode on the TOE. It states that it limits the TOE to only the cryptographic operations claimed.

**Component Testing Assurance Activities**: ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

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

ECDSA FIPS 186-4 Signature Verification Test

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

RSA Signature Algorithm Tests

Signature Generation Test

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

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

Signature Verification Test

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

## 2.2.8 Random Bit Generation (NDcPP22e:FCS_RBG_EXT.1)

### 2.2.8.1 NDcPP22e:FCS_RBG_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.2.8.2 NDcPP22e:FCS_RBG_EXT.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

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

Section 6.1, Table 18 (FCS_RBG_EXT.1) in the [ST] states that the TOE implements a NIST-approved HMAC Deterministic Random Bit Generator (DRBG), as specified in ISO/IEC 18031:2011 seeded by an entropy source that accumulates entropy from a TSF-hardware based noise source. The deterministic RBG is seeded with a minimum of 256 bits of entropy, which is at least equal to the greatest security strength of the keys and hashes that it will generate.

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

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

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms in the evaluated configuration.

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

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

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

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

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

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

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

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.

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

### 2.2.9.1 NDcPP22e:FCS_SSHS_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.2.9.2 NDcPP22e:FCS_SSHS_EXT.1.2

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

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

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] states that the TOE supports both public key based and password based authentication. The TOE implementation of SSHv2 supports the following:

- public key algorithms for client authentication: ssh-rsa, rsa-sha2-256, rsa-sha2-512

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

- Password-based authentication for administrative users accessing the TOE's CLI through SSHv2

The client authentication algorithms are consistent with the selections made in FCS_COP.1/SigGen

The section also states that for public key-based authentication, the TOE verifies that the SSH client's presented public key matches one that is stored within the SSH server's authorized_keys file

**Guidance Assurance Activities**: None Defined

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

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

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

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

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

(TD0631 applied)

Test 1: The evaluator configured a user to be able to login using the SSH interface with public-key based authentication, and observed the user login was successful.

Test 2: The evaluator attempted to login using the SSH interface with public-key authentication without configuring a public key for that user and observed that the login attempt was not successful.

Test 3: The evaluator configured the TOE for password authentication on the SSH interface. The evaluator logged in using an SSH client and the correct password. The login was successful.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Test 4: The evaluator attempted an SSH connection using an invalid password. The evaluator was not able to login.

#### 2.2.9.3 NDcPP22e:FCS_SSHS_EXT.1.3

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] states that SSHv2 connections will be dropped if the TOE receives a packet larger than 1262094 bytes. Large packets are detected by the SSH implementation and dropped internal to the SSH process.

**Guidance Assurance Activities**: None Defined

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

The evaluator attempted to connect to the TOE using an SSH client and then sent a packet too large (1262095 bytes). The evaluator observed that the session was disconnected by the TOE.

#### 2.2.9.4 NDcPP22e:FCS_SSHS_EXT.1.4

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] indicates that the TOE implementation of SSHv2 supports the following encryption algorithms: aes128-ctr, aes256-ctr, aes128-gcm@openssh.com and aes256-gcm@openssh.com to ensure confidentiality of the session. This is consistent with the FCS_SSHS_EXT.1.4 requirement.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The evaluator attempted to connect to the TOE using a SSH client alternately using each of the ciphers that can be claimed to determine which ciphers are supported with successful connections. The evaluator confirmed that there were only successful connections for claimed algorithms.

#### 2.2.9.5 NDcPP22e:FCS_SSHS_EXT.1.5

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] states that the TOE supports both public key based and password based authentication. The TOE implementation of SSHv2 supports the following:

public key algorithms for authentication: rsa-sha2-256, rsa-sha2-512.

This is consistent with the FCS_SSHS_EXT.1.5 requirement.

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

Section "Enabling FIPS Mode" in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section "Steps to configure SSH server on the router" in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section "SSH public-key based authentication" in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section "Remote Administration Protocols" in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

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

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

Has effectively been moved to FCS_SSHS_EXT.1.2.

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

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

(TD0631 applied)

Test 1 - The evaluator attempted to connect to the TOE using a SSH client alternately using each of the authentication algorithms that can be claimed and confirmed that each of these ciphers are supported with successful connections.

Test 2 – The evaluator attempted to connect to the TOE using a SSH client using a disallowed authentication algorithm and verified that it was not accepted.

#### 2.2.9.6 NDcPP22e:FCS_SSHS_EXT.1.6

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] indicates that the TOE implementation of SSHv2 supports the following hashing algorithms:  hmac-sha1, hmac-sha2-256, and hmac-sha2-512 to ensure the integrity of the session.  When aes*-gcm@openssh.com is negotiated as the encryption algorithm, the MAC algorithm field is ignored and GCM is implicitly used as the MAC. This is consistent with the FCS_SSHS_EXT.1.6 requirement.

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

Section "Enabling FIPS Mode" in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section "Steps to configure SSH server on the router" in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections.  This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section "SSH public-key based authentication" in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section "Remote Administration Protocols" in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session. When aes*-gcm@openssh.com is negotiated as the encryption algorithm, the MAC algorithm field is ignored and GCM is implicitly used as the MAC.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

Test 1 and Test 2- The evaluator attempted to connect to the TOE using an SSH client alternately using each of the MAC algorithms that can be claimed and confirmed that each of these algorithms are supported with successful connections. The evaluator followed up with a disallowed MAC algorithm and verified that it was not accepted.

#### 2.2.9.7 NDcPP22e:FCS_SSHS_EXT.1.7

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] states that the key exchange method used by the TOE is diffie-hellman-group14-sha1. This is consistent with the FCS_SSHS_EXT.1.7 requirement.

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

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

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

Test 1 and Test 2 - The evaluator first attempted to connect to the TOE using an SSH client configured with an unallowable key exchange algorithm (diffie-hellman-group1-sha1) and verified that the attempt failed. Next the evaluator attempted to connect using each of the key exchange algorithms that can be claimed and confirmed that the claimed key exchange method resulted in a successful connection.

#### 2.2.9.8 NDcPP22e:FCS_SSHS_EXT.1.8

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

a) Both thresholds are checked by the TOE.

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

Section 6.1, Table 18 (FCS_SSHS_EXT.1) in the [ST] states that remote CLI SSHv2 sessions are limited to an administrator configurable session timeout period and will be rekeyed after no more than 1 gigabyte of data is transmitted or an hour has passed. Both of these thresholds are checked by the TOE and a rekeying is performed on whichever threshold is reached first.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section "SSH public-key based authentication" in the [Admin Guide] provides the commands for configuring the ssh server rekey time and volume. The "ssh server rekey-time *minutes*" and "ssh server rekey-volume *data in megabytes*" commands configure the SSH REKEY to a limit of 60 minutes and 1 gigabyte (binary) of data. In the evaluated configuration, the administrator must set the limits to 60 minutes and 1 gigabyte (binary) of data. Based on time the administrator will need to wait for 60 minutes before the rekey occurs. The TOE will react to the first threshold reached (time or volume).

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

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

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

The evaluator then opened an SSH session and monitored the session for a re-key message being sent from the TOE. The evaluator observed that a re-key was sent by the TOE at before 60 minutes. The evaluator then opened a new SSH session and performed actions within the SSH session that caused data transfers. The evaluator observed that the TOE re-keyed before 1GB of data was transferred.

**Component TSS Assurance Activities**: None Defined

**Component Guidance Assurance Activities**: None Defined

**Component Testing Assurance Activities**: None Defined

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

#### 2.2.10.1 NDcPP22e:FCS_TLSC_EXT.1.1

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

Section 6.1, Table 18 (FCS_TLSC_EXT.1) in the [ST] states that the TOE supports TLSv1.1 and TLSv1.2 to protect the sessions to the remote audit server. Any session where the client offers the following in the client hello: SSL 2.0, SSL 3.0 and TLS 1.0 will be rejected by the TOE (client). The TOE’s implementation of TLS supports the following ciphersuites:

TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268

This is consistent with the TLS ciphersuites listed in the requirement.

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

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

The “Logging Protection” section in the [Admin Guide] states that using a secure TLS connection for Syslog Server is required in the evaluated configuration.  The minimum TLS version allowed for use are TLSv1.1 and TLSv1.2 with support for the following ciphers that are available by default in FIPS mode.

- TLS_RSA_WITH_AES_128_CBC_SHA
- TLS_RSA_WITH_AES_256_CBC_SHA

Section “Logging to Syslog Server via TLS” in the [Admin Guide] provides the steps for configuring TLS for communications between the TOE and the syslog server.

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

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

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

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

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

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

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

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

For the following tests the evaluator configured the test server to require mutual authentication and configured the TOE to establish a TLS session with a test server.

Test 1: The evaluator established a TLS session from the TOE to a test server with the test server configured to accept connections with only one of the claimed ciphersuites at a time. The evaluator used a network sniffer to capture the TLS session negotiation. The evaluator examined each traffic capture and observed that the expected TLS cipher was negotiated.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Test 2: The evaluator first connected the TOE to a TLS server with a valid certificate. The evaluator observed via packet capture that the connection was successful. Next, the evaluator connected the TOE to a TLS server that had an invalid certificate missing the serverAuth key usage. The packet capture indicated that the connection was rejected by the TOE.

Test 3: The evaluator sent a server certificate in the TLS connection that did not match the server-selected ciphersuite. The evaluator observed via packet capture that the connection was rejected.

Test 4a: The evaluator configured a test server to accept only the TLS_NULL_WITH_NULL_NULL ciphersuite. The evaluator then attempted to establish a TLS session from the TOE to that test server. The packet capture indicated that the TOE rejected the ciphersuite and disconnected the session.

Test 4b: The evaluator configured a test server to send a ciphersuite not presented in the client hello. The evaluator then attempted to establish a TLS session from the TOE to that test server. The packet capture indicated that the TOE rejected the invalid ciphersuite and disconnected the session.

Test 4c: Not applicable as the TOE does not support DHE ciphers.

Test 5a: The evaluator configured a test server to send an invalid TLS version (1.4). The evaluator then attempted to establish a TLS session from the TOE to that test server. The packet capture indicated that the TOE rejected the invalid TLS version and disconnected the session.

Test 5b: Not applicable as the TOE does not support DHE ciphers

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

#### 2.2.10.2 NDcPP22e:FCS_TLSC_EXT.1.2

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

Section 6.1, Table 18 (FCS_TLSC_EXT.1) in the [ST] states that the TOE requires Subject Alternative Names (SANs) “the reference identifiers” for a successful connection. SANs contain one or more alternate names and use any variety of name forms for the entity that is bound by the Certificate Authority (CA) to the certified public key. These alternate names are called “Subject Alternative Names" (SANs).

Possible names include:

DNS name.

IP addresses are not supported in the SAN or CN.

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

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

The “X.509 Certificates” section in the [Admin Guide] provides instructions for configuring the host name and IP domain of the syslog server which is used as the reference identifier. The TOE requires Subject Alternative Names (SANs) “the reference identifiers” for a successful connection. The Domain Name system (DNS) name in the SAN is used to match to the configured reference identifier.

IP addresses are not supported in the SAN or CN.

The syslog connection fails if the audit server certificate does not meet any one of the following criteria:

- The certificate is not signed by the CA with CA flag set to TRUE.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

• The certificate is not signed by a trusted CA in the certificate chain.

• The certificate Common Name (CN) or Subject Alternative Name (SAN) does not match the expected DNS name(i.e., reference identifier).

• The extendedKeyUsage is present, and either the keyEncipherment bit or the keyAgree-ment bit are set (both bits may be set).

• The certificate has been revoked.

• The certificate has been modified.

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

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

or

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

or

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

Note that for some tests additional conditions apply.

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

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

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

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

(TD0790 applied, supersedes TD0670)

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

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

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

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

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

Test 1 - The evaluator configured the TOE to connect with the test server using TLS alternately configured with matching certificate identifiers and certificate identifiers that do not match the reference identifier in either the

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

SAN or the CN. The evaluator confirmed that the connections were only successful when the identifier fulfilled the required rules.

Test 2-4 - These tests were demonstrated in Test 1.

Test 5: The evaluator tested hostname wildcards by configuring an expected DNS on the TOE with the test server's certificates configured with wildcard DNS names. The TOE does not support the use of wildcards in certificates. The TOE rejected all certificates that included hostname wildcards.

| Certificate Contents | Host ID | Expected Result |
|---|---|---|
| CN=bar.*.example.com | bar.foo.example.com | No Connection |
| SAN=bar.*.example.com | bar.foo.example.com | No Connection |
| CN=*.example.com | bar.foo.example.com | No Connection |
| SAN=*.example.com | bar.foo.example.com | No Connection |
| CN=*.example.com | foo.example.com | Successful Connection |
| SAN=*.example.com | foo.example.com | No Connection |
| CN=*.com | example.com | No Connection |
| SAN=*.com | example.com | No Connection |

Test 6: The TOE does not support IP addresses in the SAN or CN.

Test 7: The TOE does not support FPT_ITT.1

### 2.2.10.3 NDcPP22e:FCS_TLSC_EXT.1.3

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

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

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

Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

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

Test 1 and 2 were performed as part of testing of FIA_X509_EXT.1.1/Rev Test 1.  Test 3 is not applicable as the TOE does not offer administrative override of certificate validation failures.

#### 2.2.10.4 NDcPP22e:FCS_TLSC_EXT.1.4

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

Section 6.1, Table 18 (FCS_TLSC_EXT.1) in the [ST] states that the TOE does not support NIST curves in the TLS Client Hello.

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

As indicated in the ST, the TOE does not support Elliptic Curves Extension.

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Not applicable as the TOE does not support elliptic curves.

**Component TSS Assurance Activities**: None Defined

**Component Guidance Assurance Activities**: None Defined

**Component Testing Assurance Activities**: None Defined

## 2.3 Identification and authentication (FIA)

### 2.3.1 Authentication Failure Management (NDcPP22e:FIA_AFL.1)

#### 2.3.1.1 NDcPP22e:FIA_AFL.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.3.1.2 NDcPP22e:FIA_AFL.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section 6.1, Table 18 (FIA_AFL.1) in the [ST] states that the TOE enforces a timed lockout after an Administrator defined number of unsuccessful password attempts is exceeded. While the TOE supports a range from 2-6, in the evaluated configuration, the maximum number of failed attempts is recommended to be set to 3. Once the remote user is locked out, their account will not be accessible until the configured timer for lockout has been exceeded. Once the lockout time is over, then the administrator user can attempt to login again. At no point is administrator access completely unavailable when remote administrators are locked out due to unsuccessful password attempts. Local console access is always available. Administrator lockouts are not applicable to the local console.

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

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

The “User Lockout” section in the [Admin Guide] provides instructions for configuring lockout of user accounts after a specified number of authentication failures including the maximum number of authentication attempts and the lockout duration. The section also notes that Local accounts are never locked out.

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

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

b) Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

If the administrator action selection in FIA_AFL.1.2 is included in the ST then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator's access results in successful access (when using valid credentials for that administrator).

If the time period selection in FIA_AFL.1.2 is included in the ST then the evaluator shall wait for just less than the time period configured in Test 1 and show that an authorisation attempt using valid credentials does not result in successful access. The evaluator shall then wait until just after the time period configured in Test 1 and show that an authorisation attempt using valid credentials results in successful access.

Test 1 & 2: The evaluator configured a limit on failed authentication attempts (i.e., 3 failures) as well as the amount of time to lock the account after reaching that limit (i.e, 45 seconds). The evaluator then performed more login attempts using incorrect credentials than the configured limit. The evaluator observed that the use of valid credentials immediately after exceeding the limit does not result in a successful login. The evaluator then waited for the configured time to pass (45 seconds), then observed that the user could login successfully with the correct password and that the count of failed login attempts was reset to zero. These steps were completed for SSH.

## 2.3.2 Password Management - per TD0792 (NDcPP22e:FIA_PMG_EXT.1)

### 2.3.2.1 NDcPP22e:FIA_PMG_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

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

(TD0792 applied)

Section 6.1, Table 18 (FIA_PMG_EXT.1) of the [ST] states that the TOE supports the local definition of users with corresponding passwords. The passwords can be composed of any combination of upper and lower case letters,

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

numbers, and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”. Minimum password length is settable by the Authorized Administrator and can be configured for minimum password lengths of 15 characters.

**Component Guidance Assurance Activities**: The evaluator shall examine the guidance documentation to determine that it:

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

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

The “Passwords” section in the [Admin Guide] indicates that password complexity is not enforced by the router by default, and must be administratively set in the configuration. To prevent administrators from choosing insecure passwords, for the evaluated configuration passwords must be a minimum length of 15 characters and composed of any combination of upper and lower case letters, numbers, and the following special characters: *“!”,* “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “), *[no other characters]*]. The password minimum length is configured using the ‘min-length’ command. This section further provides the commands for configuring the number of special characters allowed in the password policy and for encrypting the password.

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

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

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

Test 1 & 2 - The evaluator attempted to set/change a password for a user’s account using several attempts. Throughout those attempts, every upper case letter, lower case letter, digit, and special character (as specified by the SFR in [ST]) were used in a password. The evaluator confirmed that a minimum length of 15 was required by attempting to set a password with 14 characters (and observing the TOE reject the password) and attempting to set a password of 15 characters (and observing that the TOE accepted the password change). The evaluator also confirmed that a password one larger than the maximum claimed by [ST] was rejected.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

### 2.3.3 Protected Authentication Feedback (NDcPP22e:FIA_UAU.7)

#### 2.3.3.1 NDcPP22e:FIA_UAU.7.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: None Defined

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

The TOE always protects authentication data as it is entered, no administrative actions are needed for this feature.

**Component Testing Assurance Activities**: The evaluator shall perform the following test for each method of local login allowed:

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

The evaluator observed that passwords are obscured while logging in at the console and via SSH.

### 2.3.4 Password-based Authentication Mechanism (NDcPP22e:FIA_UAU_EXT.2)

#### 2.3.4.1 NDcPP22e:FIA_UAU_EXT.2.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**Testing Assurance Activities**: None Defined

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

See FIA_UIA_EXT.1

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

See FIA_UIA_EXT.1

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

See FIA_UIA_EXT.1.

### 2.3.5 User Identification and Authentication (NDcPP22e:FIA_UIA_EXT.1)

#### 2.3.5.1 NDcPP22e:FIA_UIA_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.3.5.2 NDcPP22e:FIA_UIA_EXT.1.2

**TSS Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

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

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

Section 6.1, Table 18 (FIA_UIA_EXT.1) in the [ST] states that the TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed except for the login warning banner that is displayed prior to user authentication.

Administrative access to the TOE is facilitated through the TOE’s CLI. The TOE mediates all administrative actions through the CLI. Once a potential administrative user attempts to access the CLI of the TOE through either a directly connected console or remotely through an SSHv2 secured connection, the TOE prompts the user for a user name and password. Only after the administrative user presents the correct authentication credentials will access to the TOE administrative functionality be granted. No access is allowed to the administrative functionality of the TOE until an administrator is successfully identified and authenticated. The process for authentication is the same for administrative access whether administration is occurring via a directly connected console or remotely via SSHv2 secured connection.

The TOE provides a local password-based authentication mechanism for authentication of authorized administrators.  The TOE also provides a password based or SSH public key based mechanism for remote authentication of authorized administrators.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section "Initial Setup via Direct Console Connection" in the [Admin Guide] provides instructions for initial setup of the TOE via direct console connection. Section "System Admin Console" then provides the steps for accessing the local admin console including configuring a username and password for each administrator and configuring local authentication.

Section "Management Interface" in the [Admin Guide] describes how to configure the management interface for system management and remote communication including configuring an IP address and subnet mask for the management ethernet interface.

Section "Enabling FIPS Mode" in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section "Steps to configure SSH server on the router" in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section "SSH public-key based authentication" in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section "Remote Administration Protocols" in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section “X.509 certificates” in the [Admin Guide] describes configuring the rsa keypair in the certificate request.

Section “Logging Protection” in the [Admin Guide] indicates that the supported TLS ciphersuites are available by default in FIPS mode.

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

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

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

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

d) Test 4: For distributed TOEs where not all TOE components support the authentication of Security Administrators according to FIA_UIA_EXT.1 and FIA_UAU_EXT.2, the evaluator shall test that the components authenticate Security Administrators as described in the TSS.

The TOE offers the following user interfaces where authentication is provided.

- Local Console
- SSH using passwords
- SSH using public/private key pairs

Test 1 - Using each interface the evaluator performed an unsuccessful and successful logon of each type using bad and good credentials respectively.

Test 2 - Using an Nmap scan, the evaluator determined which services were available to an external remote entity. The list of available services was confirmed by the evaluator to be limited to those specified in the requirement.

Test 3 – Using the local console login and ssh login, the evaluator found that, prior to login, no functions were available to the administrator prior to logging in.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Test 4 - The TOE is not distributed, thus tests 1 through 3 above test the only TOE component.

## 2.3.6 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/Rev)

### 2.3.6.1 NDcPP22e:FIA_X509_EXT.1.1/Rev

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

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

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

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

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

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

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

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

f) Test 6: The evaluator shall modify any byte in the last byte of the certificate and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

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

h) The following tests are run when a minimum certificate path length of three certificates is implemented.

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

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

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

Test 8c: The evaluator shall establish a subordinate CA certificate, where the elliptic curve parameters are specified as a named curve, that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is accepted into the TOE's trust store. The evaluator shall then establish a

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

subordinate CA certificate that uses an explicit format version of the elliptic curve parameters, and that is signed by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

(TD0527 12/2020 update applied)

Test 1a & 1b -- The evaluator configured the TOE and a peer with valid certificates. The evaluator then attempted to make a connection between the peer devices. A successful connection was made. The evaluator then configured a server certificate with an invalid certification path by deleting the root CA so that the certificate chain was invalid because of a missing (or deleted) certificate. The connection between the peers was refused.

Test 2 -- The evaluator attempted to make a connection between the TOE and a test server. The test server then presented an expired certificate during the negotiation resulting in a failed connection.

Test 3 -- The evaluator attempted to make a connection between the TOE and a test server using TLS. The test server then presented a certificate during the protocol negotiation where the certificate was valid. A packet capture was obtained of these protocol negotiations which shows that the connection was successful. The evaluator revoked certificates in the chain (individually) and attempted the same connection. The attempt after revoking the certificate was not successful.

Test 4 -- The evaluator configured a test server to sign a CRL with a certificate that does not have the cRLsign key usage bit set. The evaluator established a connection between the TOE and the test server and ensured that the connection was not negotiated successfully.

Test 5 -- The evaluator configured a test server to present a certificate that had a byte in the first eight bytes modified to the TOE. The evaluator then attempted to make a connection between the peer devices. When the TOE attempted to connect, the connection failed.

Test 6 -- The evaluator configured a test server to present a certificate that had a byte in the last eight bytes modified to the TOE. The evaluator then attempted to make a connection between the peer devices. When the TOE attempted to connect, the connection failed.

Test 7 -- The evaluator configured a test server to present a certificate that had a byte in the public key of the certificate modified to the TOE. The evaluator then attempted to make a connection between the peer devices. When the TOE attempted to connect, the connection was refused.

Test 8 -- Not applicable as the TOE does not support EC curves

#### 2.3.6.2 NDcPP22e:FIA_X509_EXT.1.2/Rev

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation). For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

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

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

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

Test 1: The evaluator configured a test syslog server to present a certificate chain containing a CA certificate lacking the basicConstraints extension. The evaluator then used the TOE TLS client (syslog) to attempt to connect to the test server. The evaluator observed that the TOE rejected the connections.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Test 2: The evaluator configured a test server to present a certificate chain containing a CA certificate having the basicConstraints section but with the cA flag not set (i.e., FALSE). The evaluator then used the TOE TLS client (syslog) to attempt to connect to the test server. The evaluator observed that the TOE rejected the connection.

**Component TSS Assurance Activities**: The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

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

Section 6.1, Table 18 (FIA_X509_EXT.1) in the [ST] states that the TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for TLS connections. Public key infrastructure (PKI) credentials, such as private keys and certificates are stored securely. The identification and authentication, and authorization security functions protect an unauthorized user from gaining access to the storage. The certificate request message includes the public key and the following information per RFC 2986:

Common Name

Organization

Organizational unit

Country

The certificate validation checking takes place during the TLS session establishment and at time of import. The TOE conforms to standard RFC 5280 for certificate and path validation (i.e., peer certificate checked for expiration, peer certificate checked if signed by a trusted CA in the trust chain, peer certificate checked for unauthorized modification, peer certificate checked for revocation).

The TOE supports Self-signed certificate enrollment for a trust point to obtain a certificate from a CA:

The certificate chain establishes a sequence of trusted certificates, from a peer certificate to the root CA certificate. Within the PKI hierarchy, all enrolled peers can validate the certificate of one another if the peers share a trusted root CA certificate or a common subordinate CA. Each CA corresponds to a trust point. When a certificate chain is received from a peer, the default processing of a certificate chain path continues until the first trusted certificate, or trust point, is reached. Both the certificate request message and the certificates themselves

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

provide protection in that they are digitally signed.  If a certificate is modified in any way, it would be invalidated. The digital signature verifications process would show that the certificate had been tampered with when the hash value would be invalid.

Checking is also done for the basicConstraints extension and the CA flag to determine whether they are present and set to TRUE. The local certificate that was imported must contain the basic constraints extension with the CA flag set to true, the check also ensure that the key usage extension is present, and the keyEncipherment bit or the keyAgreement bit or both are set. If they are not, the certificate is not accepted. Only one certificate is imported since the only device is a syslog server, so the TOE chooses this certificate. BasicConstraints checking is performed at the time of authentication during the connection attempt. If the connection to determine the certificate validity cannot be established, the certificate is not accepted.

The administrators can configure a trust chain by importing the CA certificate(s) that signed and issued the server (syslog) certificate. This will tell the TOE which CA certificate(s) to use during the validation process. If the TOE does not find the trusted root CA, the TLS connection to the syslog server will fail. When the TOE is able to contact the CRL distribution point for certificate revocation checking, the TOE will reject the TLS session if the remote trust point's (e.g. syslog server's) certificate has been revoked.

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

The section "X.509 Certificates" in the [Admin Guide] provides detail on how X.509 Certificates are used to authenticate a remote server for secure syslog. It describes that the TOE requires Subject Alternative Names (SANs) "the reference identifiers" for a successful connection. The Domain Name system (DNS) name in the SAN is used to match to the configured reference identifier in the instructions below. The syslog connection fails if the audit server certificate that does not meet any one of the following criteria:

The certificate is not signed by the CA with cA flag set to TRUE.

The certificate is not signed by a trusted CA in the certificate chain.

The certificate Common Name (CN) or Subject Alternative Name (SAN) does not match the expected DNS name(i.e., reference identifier).

The extendedKeyUsage is present, and either the keyEncipherment bit or the keyAgree-ment bit are set (both bits may be set).

The certificate has been revoked or modified.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The section also states that the Revocation check is done via CRL as specified in RFC 5759 Section 5. Revocation is checked when initially importing the peer (Syslog Server) certificate into the TOEs trust store and subsequently when TLS sessions are established with the peer.

**Component Testing Assurance Activities**: None Defined

## 2.3.7 X.509 Certificate Authentication (NDcPP22e:FIA_X509_EXT.2)

### 2.3.7.1 NDcPP22e:FIA_X509_EXT.2.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.3.7.2 NDcPP22e:FIA_X509_EXT.2.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

Section 6.1, Table 18 (FIA_X509_EXT.2) in the [ST] states that the TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for TLS connections. The TOE supports Self-signed certificate enrollment for a trust point to obtain a certificate from a CA. The administrators can configure a trust chain by importing the CA certificate(s) that signed and issued the server (syslog) certificate. This will tell the TOE which CA certificate(s) to use during the validation process. If the TOE does not find the trusted root CA, the TLS connection to the syslog server will fail. When the TOE is able to contact the CRL distribution point for certificate revocation checking, the TOE will reject the TLS session if the remote trust point’s (e.g. syslog server’s) certificate has been revoked.  If the connection to determine the certificate validity cannot be established, the certificate is not accepted.

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

The sections entitled "Logging Protection" and "X.509 Certificates" explain that the TOE uses TLS and X509 Certificates to securely transfer audit records. They also describe how the administrator needs to configure the TOE in order to enable this functionality, including configuring a hostname and domain IP on the router, declaring a certificate authority and configuring a trustpoint, and configuring certificate enrollment.

The section describes that reasons for TLS connection failures and/or X.509 certificate validation failures are logged locally and may be used to provide insight as to reason for failure. If any of the established trusted channels/paths are unintentionally broken, the connection will need to be re-established following the configuration settings as described in this document.  If TLS sessions fail due to inability to contact the CRL, restore connectivity to the CRL server before reattempting to establish the TLS sessions.

**Component Testing Assurance Activities**: The evaluator shall perform the following test for each trusted channel:

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

The evaluator established a syslog connection from the TOE to a remote test server protected by TLS and obtained a packet capture of the activity.  This was repeated when the CRL responder was available and unavailable. When available the connection succeeded.  When the CRL responder was unavailable, the connection failed.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

### 2.3.8 X.509 Certificate Requests (NDcPP22e:FIA_X509_EXT.3)

#### 2.3.8.1 NDcPP22e:FIA_X509_EXT.3.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.3.8.2 NDcPP22e:FIA_X509_EXT.3.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: If the ST author selects 'device-specific information', the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

Section 6.1, Table 18 (FIA_X509_EXT.3) in the [ST] states that the certificate request message includes the public key and device-specific information which includes:

Common Name

Organization

Organizational unit

Country

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The “X.509 Certificates” section in the [Admin Guide] provides the steps for generating a certificate request with the public key and device-specific information and for requesting certificates from a CA.

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

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

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

Test 1- The evaluator generated a certificate signing request by following the instructions in the administration guidance for generating the request. The evaluator viewed the contents of the CSR and confirmed that it provided public key and other required information including the information that the evaluator supplied during the generation process.

Test 2 - The evaluator attempted to import a certificate that was signed by a root CA that was not installed on the TOE. The attempt failed. The evaluator then loaded the necessary CA to validate the certificate, then successfully installed the certificate resulting from the CSR request in test 1.

## 2.4 Security management (FMT)

### 2.4.1 Management of security functions behaviour (NDcPP22e:FMT_MOF.1/ManualUpdate)

#### 2.4.1.1 NDcPP22e:FMT_MOF.1.1/ManualUpdate

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: For distributed TOEs it is required to verify the TSS to ensure that it describes how every function related to security management is realized for every TOE component and shared

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

There are no specific requirements for non-distributed TOEs.

The TOE is not a distributed TOE.

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

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

The "Secure Acceptance of the TOE" section in the [Admin Guide] describes the approved method for downloading a Common Criteria evaluated software image from Cisco.com. Once the file is downloaded, the authorized administrator verifies that it was not tampered with by using a hash utility to verify the SHA512 published hash by comparing the SHA512 published hash that is listed on the Cisco web site. If the hashes do not match, the user is instructed to contact Cisco Technical Assistance Center (TAC) <https://www.cisco.com/c/en/us/support/index.html>.

Section "Product Updates" in the [Admin Guide] provides the necessary steps for an administrator to manually install packages and perform software upgrades.

**Component Testing Assurance Activities**: The evaluator shall try to perform the update using a legitimate update image without prior authentication as Security Administrator (either by authentication as a user with no administrator privileges or without user authentication at all - depending on the configuration of the TOE). The attempt to update the TOE should fail.

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

The set of functions available to a user prior to login do not include TOE update as specified by NDcPP22e:FIA_UIA_EXT.1.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

As can be seen in the NDCPP22e: FIA_UIA_EXT.1-t3 test evidence, no functions are offered to users prior to a successful login. The evaluator found no method of updating the TOE prior to logging in as a Security Administrator. Testing activities for FPT_TUD_EXT.1 - Test 1 demonstrate a successful update attempt performed by a Security Administrator with prior authentication.

## 2.4.2 Management of TSF Data (NDcPP22e:FMT_MTD.1/CoreData)

### 2.4.2.1 NDcPP22e:FMT_MTD.1.1/CoreData

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

Section 6.1, Table 18 (FIA_UIA_EXT.1) in the [ST] states that the TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed except for the login warning banner that is displayed prior to user authentication.

Section 6.1, Table 18 (FMT_MTD.1/CoreData) in the [ST] states that the TOE provides administrative users with a CLI to interact with and manage the security functions of the TOE. The term “Authorized Administrator” refers to any user which has been assigned to a privilege level that is permitted to perform the relevant action; therefore, has the appropriate privileges to perform the requested functions. Therefore, semi-privileged administrators with only a subset of privileges may also manage and modify TOE data based on the privileges assigned.

The TOE provides the ability for Authorized Administrators to access TOE data, such as user accounts and roles, audit data, audit server information, configuration data, security attributes, X509 certificates, login banners, inactivity timeout values, password complexity setting, TOE updates and session thresholds via the CLI. The TOE

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

restricts the access to manage TSF data that can affect security functions of the TOE to the Authorized Administrator/Security Administrator roles.

Section 6.1, Table 18 (FIA_X509_EXT.1) states that the TOE supports Self-signed certificate enrollment for a trust point to obtain a certificate from a CA. Authorized administrators can configure a trust chain by importing the CA certificate(s) that signed and issued the server (syslog) certificate. This will tell the TOE which CA certificate(s) to use during the validation process. If the TOE does not find the trusted root CA, the TLS connection to the syslog server will fail.

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

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

The TSF data manipulating functions and the corresponding configuration information from specific sections of the [Admin Guide] are identified or referenced throughout this AAR with the requirement to which they apply.

Section “User Roles” in the [Admin Guide] describes how users and their privileges are configured. It states that authenticated users are entitled to execute commands and access data elements based on the command rules and data rules that are created and applied to user groups.

Section “X.509 Certificates” in the [Admin Guide] describes how Authorized Administrators can configure a trust point and securely import certificates to the TOE’s trust store.

**Component Testing Assurance Activities**: No separate testing for FMT_MTD.1/CoreData is required unless one of the management functions has not already been exercised under any other SFR.

All of the management functions have been exercised under the other SFRs as demonstrated throughout the AAR.

### 2.4.3 Management of TSF Data (NDcPP22e:FMT_MTD.1/CryptoKeys)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

### 2.4.3.1 NDcPP22e:FMT_MTD.1.1/CryptoKeys

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: For distributed TOEs see chapter 2.4.1.1.

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

Section 6.1, Table 18 (FIA_UIA_EXT.1) in the [ST] states that the TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed except for the login warning banner that is displayed prior to user authentication.

Section 6.1, Table 18 (FMT_MTD.1/CryptoKeys) in the [ST] states that the TOE provides administrative users with a CLI to interact with and manage the security functions of the TOE. The term "Authorized Administrator" refers to any user which has been assigned to a privilege level that is permitted to perform the relevant action; therefore, has the appropriate privileges to perform the requested functions. Therefore, semi-privileged administrators with only a subset of privileges may also manage and modify TOE data based on the privileges assigned.

The TOE provides the ability for Authorized Administrators to access TOE data, such as user accounts and roles, audit data, audit server information, configuration data, security attributes, X509 certificates, login banners, inactivity timeout values, password complexity setting, TOE updates and session thresholds via the CLI. The TOE provides the ability for Authorized Administrators to manage cryptographic keys. This ability takes the form of a management activity that will create keys, import keys, configure the use of keys, and destroy keys. The keys that can be managed are associated with a CSR, x509v3 certificates, and SSH public keys. The TOE restricts the access to manage TSF data that can affect security functions of the TOE to the

**Component Guidance Assurance Activities**: For distributed TOEs see chapter 2.4.1.2.

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

The TSF data manipulating functions and the corresponding configuration information from specific sections of the [Admin Guide] are identified or referenced throughout this AAR with the requirement to which they apply.

The section "Steps to configure SSH server on the router" in the [Admin Guide] describes how to generate rsa keys.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The section “SSH public-key based authentication” in the [Admin Guide] describes how to import public keys for ssh public-key based authentication.

The section “X.509 Certificates” in the [Admin Guide] describes how to configure certificate enrollment, including setting the keypair for the trustpoint.

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

The test NDcPP22e:FIA_UIA_EXT.1-t3 demonstrates that there are no functions (including managing the cryptographic keys) available to users prior to login. The test FIA_X509_EXT.3-t1 demonstrates an authenticated administrator generating X509 keypairs for certificate generation.

### 2.4.4 Specification of Management Functions - per TD0631 (NDcPP22e:FMT_SMF.1)

#### 2.4.4.1 NDcPP22e:FMT_SMF.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: The security management functions for FMT_SMF.1 are distributed throughout the cPP and are included as part of the requirements in FTA_SSL_EXT.1, FTA_SSL.3, FTA_TAB.1, FMT_MOF.1(1)/ManualUpdate, FMT_MOF.1(4)/AutoUpdate (if included in the ST), FIA_AFL.1, FIA_X509_EXT.2.2 (if included in the ST), FPT_TUD_EXT.1.2 & FPT_TUD_EXT.2.2 (if included in the ST and if they include an administrator-configurable action), FMT_MOF.1(2)/Services, and FMT_MOF.1(3)/Functions (for all of these SFRs that are included in the ST), FMT_MTD, FPT_TST_EXT, and any cryptographic management functions specified in the reference standards. Compliance to these requirements satisfies compliance with FMT_SMF.1.

(containing also requirements on Guidance Documentation and Tests)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

Section 6.1, Table 18 (FMT_SMF.1) in the [ST] states that the TOE provides all the capabilities necessary to securely manage the TOE. The administrative user can connect to the TOE using the CLI to perform these functions via SSHv2, a terminal server, or at the local console. The [Admin Guide] provides the configuration syntax, commands, and information related to each of these functions.

See the other requirements in this AAR as referenced. All security management functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply.

The “System Admin Console” section of the [Admin Guide] states the NCS local administrative interface is a serial console. The administrator can ensure that the interface they are using is local by verifying their connection is through the serial console.

**Component Guidance Assurance Activities**: See TSS Assurance Activities

See TSS assurance activities above.

**Component Testing Assurance Activities**: The evaluator tests management functions as part of testing the SFRs identified in section 2.4.4. No separate testing for FMT_SMF.1 is required unless one of the management functions in FMT_SMF.1.1 has not already been exercised under any other SFR.

All TOE security functions are identified in the guidance documentation and have been tested as documented throughout this AAR.

### 2.4.5 Restrictions on Security Roles (NDcPP22e:FMT_SMR.2)

#### 2.4.5.1 NDcPP22e:FMT_SMR.2.1

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.4.5.2 NDcPP22e:FMT_SMR.2.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.4.5.3 NDcPP22e:FMT_SMR.2.3

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

**Component TSS Assurance Activities**: The evaluator shall examine the TSS to determine that it details the TOE supported roles and any restrictions of the roles involving administration of the TOE.

Section 6.1, Table 18 in the [ST] provides the following:

FMT_SMR.2 states The TOE platform maintains both privileged and semi-privileged administrator roles. The terms "Authorized Administrator" and "Security Administrator" are used interchangeable in this ST to refer to any user that has been assigned to a privilege level that is permitted to perform the relevant action; therefore, has the appropriate privileges to perform the requested functions. The assigned role determines the functions the user can perform; hence the authorized administrator with the appropriate privileges.

The TOE supports both local administration via a directly connected console and remote authentication via SSH

FMT_MTD.1/CoreData states that the TOE provides administrative users with a CLI to interact with and manage the security functions of the TOE.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The term “Authorized Administrator” is used in this ST to refer to any user which has been assigned to a privilege level that is permitted to perform the relevant action; therefore, has the appropriate privileges to perform the requested functions. Therefore, semi-privileged administrators with only a subset of privileges may also manage and modify TOE data based on the privileges assigned.

The TOE provides the ability for Authorized Administrators to access TOE data, such as user accounts and roles, audit data, audit server information, configuration data, security attributes, X509 certificates, login banners, inactivity timeout values, password complexity setting, TOE updates and session thresholds via the CLI. The TOE restricts the access to manage TSF data that can affect security functions of the TOE to the Authorized Administrator/Security Administrator roles.

Manual software updates can only be done by the authorized administrator through CLI. These updates include software upgrades.

The Security Administrators (a.k.a Authorized Administrators) can query the software version running on the TOE and can initiate updates to (replacements of) software images. When software updates are made available by Cisco, the Authorized Administrators can obtain, verify the integrity of, and install those updates.

FMT_SMF.1 states that The TOE provides all the capabilities necessary to securely manage the TOE. The administrative user can connect to the TOE using the CLI to perform these functions via SSHv2, a terminal server, or at the local console. Refer to the Guidance documentation for configuration syntax, commands, and information related to each of these functions.

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

Section “Initial Setup via Direct Console Connection” in the [Admin Guide] provides instructions for initial setup of the TOE via direct console connection. Section “System Admin Console” then provides the steps for accessing the local admin console including configuring a username and password for each administrator and configuring local authentication.

Section “Management Interface” in the [Admin Guide] describes how to configure the management interface for system management and remote communication including configuring an IP address and subnet mask for the management ethernet interface.

Section “User Roles” in the [Admin Guide] provides instructions for creating administrative users and assigning privileges.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections. This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

**Component Testing Assurance Activities**: In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

The TOE is administered through either an SSH protected CLI, or the console CLI. Both were used for varying configuration operations.

Both interfaces were exercised during NDcPP22e:FIA_UIA_EXT.1 and the various FTA test cases.

## 2.5 Protection of the TSF (FPT)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

## 2.5.1 Protection of Administrator Passwords (NDcPP22e:FPT_APW_EXT.1)

### 2.5.1.1 NDcPP22e:FPT_APW_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.5.1.2 NDcPP22e:FPT_APW_EXT.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FPT_APW_EXT.1) in the [ST] states that all passwords are obscured via hashing in a secure directory. The passwords are non readable. In this manner, the TOE ensures that plaintext user passwords will not be disclosed even to administrators. This is provided by default.

**Component Guidance Assurance Activities**: None Defined

**Component Testing Assurance Activities**: None Defined

## 2.5.2 Protection of TSF Data (for reading of all pre-shared, symmetric and private keys) (NDcPP22e:FPT_SKP_EXT.1)

### 2.5.2.1 NDcPP22e:FPT_SKP_EXT.1.1

**TSS Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FPT_SKP_EXT.1) in the [ST] states that the TOE stores all private keys in a secure directory that is not readily accessible to administrators. All pre-shared and symmetric keys are stored in a hashed format that is non-readable, hence no interface access.

**Component Guidance Assurance Activities**: None Defined

**Component Testing Assurance Activities**: None Defined

### 2.5.3 Reliable Time Stamps – per TD0632 (NDcPP22e:FPT_STM_EXT.1)

#### 2.5.3.1 NDcPP22e:FPT_STM_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.5.3.2 NDcPP22e:FPT_STM_EXT.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section 6.1, Table 18 (FPT_STM_EXT.1) in the [ST] states that the TOE provides a source of date and time information used in audit event timestamps and in validating service requests.  This function can only be accessed from within the configuration exec mode via the privileged mode of operation of the router. The clock function is reliant on the system clock provided by the underlying hardware.

This date and time are used to track inactivity of administrative sessions and validate certificates.

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

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

Section “Clock Management” in the [Admin Guide] provides instructions for using the ‘clock set’ command which updates both SW as well as HW clock.

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

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

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

If the audit component of the TOE consists of several parts with independent time information, then the evaluator shall verify that the time information between the different parts are either synchronized or that it is possible for all audit information to relate the time information of the different part to one base information unambiguously.

c) Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay between the setting the time on the VS and when the time is reflected on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

Test 1 - The evaluator followed the guidance instructions to configure the time on the TOE. The evaluator read the time from the TOE using the 'show clock' command and verified the time was configured.

Test 2: Not applicable as the TOE does not obtain time from an NTP server

Test 3: Not applicable. The TOE does not obtain its time from the underlying virtual server, the time is configured by the Security Administrator.

## 2.5.4 TSF testing (NDcPP22e:FPT_TST_EXT.1)

### 2.5.4.1 NDcPP22e:FPT_TST_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

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

Section 6.1, Table 18 (FPT_TST_EXT.1) in the [ST] states that the TOE runs a suite of self-tests during initial start-up to verify its correct operation. If any of the tests fail the security administrator will have to log into the CLI to

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

determine which test failed and why. If the tests pass successfully the router will continue bootup and normal operation.

During the system bootup process (power on or reboot), all the Power on Startup Test (POST) components for all the cryptographic modules perform the POST for the corresponding component (hardware or software). These tests include:

- AES Known Answer Test - For the encrypt test, a known key is used to encrypt a known plain text value resulting in an encrypted value. This encrypted value is compared to a known encrypted value to ensure that the encrypt operation is working correctly. The decrypt test is just the opposite. In this test a known key is used to decrypt a known encrypted value. The resulting plaintext value is compared to a known plaintext value to ensure that the decrypt operation is working correctly

- RSA Signature Known Answer Test (both signature/verification) - This test takes a known plaintext value and Private/Public key pair and used the public key to encrypt the data. This value is compared to a known encrypted value to verify that encrypt operation is working properly. The encrypted data is then decrypted using the private key. This value is compared to the original plaintext value to ensure the de-crypt operation is working properly

- RNG/DRBG Known Answer Test - For this test, known seed values are provided to the DRBG implementation. The DRBG uses these values to generate random bits. These random bits are compared to known random bits to ensure that the DRBG is operating correctly

- HMAC Known Answer Test - For each of the hash values listed, the HMAC implementation is fed known plaintext data and a known key. These values are used to generate a MAC. This MAC is compared to a known MAC to verify that the HMAC and hash operations are operating correctly

- SHA-1/256/512 Known Answer Test - For each of the values listed, the SHA implementation is fed known data and key. These values are used to generate a hash. This hash is compared to a known value to verify they match and the hash operations are operating correctly.

- Software Integrity Test - The Software Integrity Test is run automatically whenever the IOS-XR system image is loaded and confirms that the image file that is about to be loaded has maintained its integrity. The software contains a SHA-512 hash. This hash is compared to a pre-loaded hash. If the hash values match, the test passes; otherwise, the test fails.

If any component reports failure for the POST, the system crashes and appropriate information is displayed on the screen, and saved in the crashinfo file. All ports are blocked from moving to forwarding state during the POST. If all components of all modules pass the POST, the system is placed in FIPS PASS state and ports are allowed to forward data traffic.

These tests are sufficient to verify that the correct version of the TOE software is running as well as that the cryptographic operations are all performing as expected.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

The "Administration of Cryptographic Self-Tests" section in the [Admin Guide] states that if any of these FIPS self-tests fail, the whole system is moved to the FIPS error state. In this state, as per the FIPS requirement, all cryptographic keys are deleted, and all line cards are shut down. This mode is exclusively meant for debugging purposes. Once the switch is in the FIPS error state, any reload of a line card moves it to the failure state. To move the switch back to FIPS mode, it has to be rebooted. However, once the switch is in FIPS mode, any power-up self-test failure on a subsequent line card reload or insertion affects only that line card, and only the corresponding line card is moved to the failure state.

If any of the self-tests fail, the TOE transitions into an error state. In the error state, all secure data transmission is halted and the TOE outputs status information (a SELF_TEST_FAILED system log) indicating the failure.

Example Error Message: *Error Message SECURITYD-2-FIPS_SELF_TEST_FAILED: FIPS self-test failure : [chars]*

Explanation: FIPS self-test failed [chars] for service [chars].

Section "Self Tests" in the [Admin Guide] describes how to verify that the self tests have been successful via the FIPS POST logs. It further states that if any of the POST fail, the following actions should be taken:

- Use the *system cores* command to set up core dumps on the system. This will provide additional information on the cause of the crash:

    ```
    RP/0/RP0/CPU0:ios#   configure
    RP/0/RP0/CPU0:ios(config)# system cores slot0:core_file
    ```

    Example:

    ```
    RP/0/RP0/CPU0:ios # system cores tftp://x.x.x.x/filename
    RP/0/RP0/CPU0:ios # show system cores
    ```

    Note: The filename (indicated by filename) must exist in the TFTP server directory.

- Restart the TOE to perform POST and determine if normal operation can be resumed

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

If the problem persists, contact Cisco Technical Assistance via

<http://www.cisco.com/techsupport> or 1 800 553-2447

Section "Modes of Operation" in the [Admin Guide] states that following operational error, the TOE reboots (once power supply is available) and enters booting mode. The only exception to this is if there is an error during the Power on Startup Test (POST) during bootup, then the TOE will shut down. If any component reports failure for the POST, the system crashes and appropriate information is displayed on the screen, and saved in the crashinfo file. Within the POST, self-tests for the cryptographic operations are performed.

All ports are blocked from moving to forwarding state during the POST. Only when all components of all modules pass the POST is the system placed in FIPS PASS state and ports are allowed to forward data traffic.

If any of the POST fail, the following actions should be taken:

- If possible, review the crashinfo file. This will provide additional information on the cause of the crash
- Restart the TOE to perform POST and determine if normal operation can be resumed
- If the problem persists, contact Cisco Technical Assistance via http://www.cisco.com/techsupport or 1 800 553-2447
- If necessary, return the TOE to Cisco under guidance of Cisco Technical Assistance

**Component Testing Assurance Activities**: It is expected that at least the following tests are performed:

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

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

Although formal compliance is not mandated, the self-tests performed should aim for a level of confidence comparable to:

a) FIPS 140-2, chap. 4.9.1, Software/firmware integrity test for the verification of the integrity of the firmware and executable software. Note that the testing is not restricted to the cryptographic functions of the TOE.

b) FIPS 140-2, chap. 4.9.1, Cryptographic algorithm test for the verification of the correct operation of cryptographic functions. Alternatively, national requirements of any CCRA member state for the security evaluation of cryptographic functions should be considered as appropriate.

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

During a reboot of the TOE, the evaluator confirmed that the TOE performed self-tests to verify the firmware integrity and the cryptographic functions. The output of these tests indicate that they were successful. The firmware integrity test passed and all other tests were successfully completed with no errors.

### 2.5.5 Trusted update (NDcPP22e:FPT_TUD_EXT.1)

#### 2.5.5.1 NDcPP22e:FPT_TUD_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.5.5.2 NDcPP22e:FPT_TUD_EXT.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.5.5.3 NDcPP22e:FPT_TUD_EXT.1.3

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

The evaluator shall verify that the TSS describes all TSF software update mechanisms for updating the system firmware and software (for simplicity the term 'software' will be used in the following although the requirements apply to firmware and software). The evaluator shall verify that the description includes a digital signature

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

verification of the software before installation and that installation fails if the verification fails. Alternatively an approach using a published hash can be used. In this case the TSS shall detail this mechanism instead of the digital signature verification mechanism. The evaluator shall verify that the TSS describes the method by which the digital signature or published hash is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature or published hash of the update, and the actions that take place for both successful and unsuccessful signature verification or published hash verification.

If the options 'support automatic checking for updates' or 'support automatic updates' are chosen from the selection in FPT_TUD_EXT.1.2, the evaluator shall verify that the TSS explains what actions are involved in automatic checking or automatic updating by the TOE, respectively.

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

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

Section 6.1, Table 18 (FPT_TUD_EXT.1) in the [ST] states that an Authorized Administrator can query the software version running on the TOE and can initiate updates to software images. When software updates are made available by Cisco, an administrator can obtain, verify the integrity of, and install those updates. The updates can be downloaded from the software.cisco.com. The cryptographic hashes (i.e., public hashes/SHA-512) are used to verify software/firmware update files (to ensure they have not been modified from the originals distributed by Cisco) before they are used to actually update the applicable TOE components. An authorized administrator can compare the hash of the downloaded file to the hash that is posted on the Cisco website. The hash value can be displayed by hovering over the software image name under details on the Cisco.com web site. If the hashes do not match, Cisco Technical Assistance Center (TAC) should be contacted.

To activate a package, use the “install activate” command.

To verify the system software release version at the CLI the command “show version” is used, while the command “show install active” command will display the currently running system image filename and the system software release version. Detailed instructions for how to verify the hash value are provided in the administrator guidance for this evaluation.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

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

For distributed TOEs the evaluator shall verify that the guidance documentation describes how the versions of individual TOE components are determined for FPT_TUD_EXT.1, how all TOE components are updated, and the error conditions that may arise from checking or applying the update (e.g. failure of signature verification, or exceeding available storage space) along with appropriate recovery actions. The guidance documentation only has to describe the procedures relevant for the Security Administrator; it does not need to give information about the internal communication that takes place when applying updates.

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

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

The “Secure Acceptance of the TOE” section in the [Admin Guide] describes the approved method for downloading a Common Criteria evaluated software image from Cisco.com. Once the file is downloaded, the authorized administrator verifies that it was not tampered with by using a hash utility to verify the SHA512 published hash by comparing the SHA512 published hash that is listed on the Cisco web site. If the hashes do not match, the user is instructed to contact Cisco Technical Assistance Center (TAC) <https://www.cisco.com/c/en/us/support/index.html>. Step 9 then provides more detailed instructions for the user to install the downloaded and verified software image onto the NCS 1000 device. Step 9 also provides the “show install prepare” command which can be used to display the inactive software packages and versions. Step 10 instructs the user how to use the “show version” and “show install active” commands to verify the system software release version.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below, and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

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

2) An image that has not been signed

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

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

c) Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted). If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the Security Administrator to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE.

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

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

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

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Test 1- The evaluator followed the guidance to verify the hash of the updated image prior to installing on the TOE. The evaluator then did the following:

- Verified the currently installed version
- Installed the new version
- Verified the update was the inactive version and that the current version had not changed
- Activated the update

Once the update was activated, the evaluator verified the version was updated to the correct new version.

Test 2 - Not applicable. The TOE only claims hash.

Test 3 - Not applicable. The TOE only claims hash. Verification of the hash value over the update file against the published hash is not performed by the TOE.

## 2.6 TOE Access (FTA)

### 2.6.1 TSF-initiated Termination (NDcPP22e:FTA_SSL.3)

#### 2.6.1.1 NDcPP22e:FTA_SSL.3.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, table 18 (FTA_SSL.3) in the [ST] states the following:
An administrator can configure maximum inactivity times individually for both local and remote administrative sessions through the use of the “exec-timeout” setting applied to the console and vty for remote sessions. These settings are not immediately activated for the current session. The current line console session must be exited. When the user logs back in, the inactivity timer will be activated for the new session. If a local user session is inactive for a configured period of time, the session will be terminated and will require re-authentication to

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

establish a new session. If a remote user session is inactive for a configured period of time, the session will be terminated and will require authentication to establish a new session.

**Component Guidance Assurance Activities**: The evaluator shall confirm that the guidance documentation includes instructions for configuring the inactivity time period for remote administrative session termination.

Section "Session Termination" in the [Admin Guide] indicates that inactivity settings trigger termination of the administrator session. By default, the remote administrative session terminates after 10 minutes of inactivity. Administrators are advised to maintain this value at 10 minutes or less, but greater than zero. This section further provides the steps and commands to configure the timeout period using the 'exec-timeout minutes seconds' command.

**Component Testing Assurance Activities**: For each method of remote administration, the evaluator shall perform the following test:

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

The evaluator followed the guidance to configure the session timeout periods for SSH CLI remote sessions and confirmed that the session was terminated after the configured time period. The inactivity time period was configured using the 'line default exec-timeout ' command for periods of 5 seconds, 25 seconds, and 60 seconds.

## 2.6.2 User-initiated Termination (NDcPP22e:FTA_SSL.4)

### 2.6.2.1 NDcPP22e:FTA_SSL.4.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section 6.1, table 18 (FTA_SSL.4) in the [ST] states an administrator is able to exit out of both local and remote administrative sessions.

**Component Guidance Assurance Activities**: The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Section “Logout” in the [Admin Guide] indicates that an administrator can manually logout from the evaluated configuration either from the local console or remotely with the following command: ‘exit’.

**Component Testing Assurance Activities**: For each method of remote administration, the evaluator shall perform the following tests:

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

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

Test 1: The evaluator logged in to the local console and then typed in the command ‘exit’. The evaluator observed that the session ended and a login prompt was presented.

Test 2: The evaluator repeated this test using an SSH connection and observed that the session ended and the SSH connections was terminated.

### 2.6.3 TSF-initiated Session Locking (NDcPP22e:FTA_SSL_EXT.1)

#### 2.6.3.1 NDcPP22e:FTA_SSL_EXT.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section 6.1, table 18 (FTA_SSL_EXT.1) in the [ST] states the following:
An administrator can configure maximum inactivity times individually for both local and remote administrative sessions through the use of the "exec-timeout" setting applied to the console. These settings are not immediately activated for the current session. The current line console session must be exited. When the user logs back in, the inactivity timer will be activated for the new session. If a local user session is inactive for a configured period of time, the session will be terminated and will require re-authentication to establish a new session.

**Component Guidance Assurance Activities**: The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Section "Session Termination" in the [Admin Guide] indicates that inactivity settings trigger termination of the administrator session. By default, the local administrative session terminates after 10 minutes of inactivity. Administrators are advised to maintain this value at 10 minutes or less, but greater than zero. This section further provides the steps and commands to configure the timeout period using the 'exec-timeout minutes seconds' command.

**Component Testing Assurance Activities**: The evaluator shall perform the following test:

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

The evaluator followed the guidance to configure the idle timeout periods for the Local Console session and confirmed that the session was terminated after the configured time period. The inactivity time period was configured using the 'exec-timeout' command for periods of 1, 3, and 5 minutes.

### 2.6.4 Default TOE Access Banners (NDcPP22e:FTA_TAB.1)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

#### 2.6.4.1 NDcPP22e:FTA_TAB.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FTA_TAB.1) states that the TOE displays a privileged Administrator specified banner on the CLI management interface prior to allowing any administrative access to the TOE. This interface is applicable for both local (via console) and remote (via SSH) TOE administration.

**Component Guidance Assurance Activities**: The evaluator shall check the guidance documentation to ensure that it describes how to configure the banner message.

Section "Login Banners" in the [Admin Guide] provides steps for configuring the login banner using the 'banner motd' command.

**Component Testing Assurance Activities**: The evaluator shall also perform the following test:

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

The evaluator configured a banner and verified that the banner was displayed appropriately for console and SSH CLI logins.

## 2.7 Trusted path/channels (FTP)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

### 2.7.1 Inter-TSF trusted channel (NDcPP22e:FTP_ITC.1)

#### 2.7.1.1 NDcPP22e:FTP_ITC.1.1

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.7.1.2 NDcPP22e:FTP_ITC.1.2

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.7.1.3 NDcPP22e:FTP_ITC.1.3

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FTP_ITC.1) in the [ST] states that the TOE protects communications with the external audit server using TLS to secure the communications channel. The TOE acts as a TLS client. TLS uses the keyed hash as defined in FCS_COP.1/KeyedHash and cryptographic hashing functions FCS_COP.1/Hash. This protects the data from modification by hashing that verifies that data has not been modified in transit. In addition, encryption of the data as defined in FCS_COP.1/DataEncryption is provided to ensure the data is not disclosed in transit.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

Section "Logging Protection" in the [Admin Guide] refers to the TLS protocol being used between the TOE and an external syslog server. It states that if any of the established trusted channels/paths are unintentionally broken, the connection will need to be re-established following the configuration settings as described in the **Admin Guide**.

Section "Logging to Syslog Server via TLS" in the [Admin Guide] describes how to enable logging to the syslog server via TLS.

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

The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

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

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

d) Test 4: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities.

The evaluator shall, for each instance where the TOE acts as a client utilizing a secure communication mechanism with a distinct IT entity, physically interrupt the connection of that IT entity for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration shorter than the application layer timeout but of sufficient length to interrupt the network link layer.

The evaluator shall ensure that, when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.

In the case where the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Further assurance activities are associated with the specific protocols.

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

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

Test 1 - This test case was demonstrated as part of Test 4.

Test 2 - This test case was demonstrated as part of Test 4.

Test 3 - This test case was demonstrated as part of Test 4.

Test 4 - The evaluator configured the syslog server and TOE so that a TLS protected connection should be accepted. The evaluator then initiated a syslog connection from the TOE by generating audit data, and then unplugged two in-line switches between the TOE and the servers for approximately 103 seconds. The evaluator then reinitiated the connection, and observed that the syslog TLS session did not renegotiate, and that the channel remained secure both during and after the disruption. The evaluator repeated this for a duration approximately 90 minutes and observed that the TOE initiated a new TLS negotiation to establish a new TLS session between the TOE and the syslog server.

The evaluator then verified within the packet captures that TLS was used (test 1), that the traffic was not cleartext (test 2) and that the encrypted tunnel could be initiated from the TOE (test 3).

## 2.7.2 Trusted Path (NDcPP22e:FTP_TRP.1/Admin)

### 2.7.2.1 NDcPP22e:FTP_TRP.1.1/Admin

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

### 2.7.2.2 NDcPP22e:FTP_TRP.1.2/Admin

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

#### 2.7.2.3 NDcPP22e:FTP_TRP.1.3/Admin

**TSS Assurance Activities**: None Defined

**Guidance Assurance Activities**: None Defined

**Testing Assurance Activities**: None Defined

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

Section 6.1, Table 18 (FTP_TRP.1/Admin) in the [ST] states that all remote administrative communications take place over a secure encrypted SSHv2 session. The SSHv2 session is encrypted using AES encryption. The remote users are able to initiate SSHv2 communications with the TOE.

This matches protocols included in the FTP_TRP.1.1/Admin SFR in the ST.

**Component Guidance Assurance Activities**: The evaluator shall confirm that the guidance documentation contains instructions for establishing the remote administrative sessions for each supported method.

Section “Management Interface” in the [Admin Guide] describes how to configure the management interface for system management and remote communication including configuring an IP address and subnet mask for the management ethernet interface.

Section “Enabling FIPS Mode” in the [Admin Guide] provides instructions for enabling FIPS mode on the TOE which restricts the algorithms to ensure that only the permitted algorithms are in the evaluated configuration.

Section “Steps to configure SSH server on the router” in the [Admin Guide] provides instructions for generating the 2048 bit RSA crypto key pair for SSH and enabling the SSH server to only accept SSHv2 client connections.  This section also indicates that ECDSA key pair is not supported in the evaluated configuration and provides the commands that need to be run in order to ensure that no ECDSA key pair is available.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

Section “SSH public-key based authentication” in the [Admin Guide] provides the steps to configure the TOE to support public-key based authentication and how to configure key exchange algorithms including generating a 2048 bit RSA keypair.

Section “Remote Administration Protocols” in the [Admin Guide] indicates that the TOE supports SSHv2 with the following by default:

- encryption algorithms, aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com to ensure confidentiality of the session.
- hashing algorithms and SSH transport implementation: hmac-sha1, hmac-sha2-256, hmac-sha2-512, to ensure the integrity of the session.
- SSH public key based authentication: rsa-sha2-256, rsa-sha2-512.
- Key Exchange Algorithms: diffie-hellman-group14-sha1

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

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

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

Further assurance activities are associated with the specific protocols.

For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.

The TOE supports remote administration via an SSH protected Command Line Interface.

The evaluator performed the following on the SSH protected CLI:

a) The evaluator initiated a packet capture of traffic between a remote administrative workstation and the TOE.

b) The evaluator connected to the TOE and performed a login using an administrator account

c) The evaluator then terminated the connection by 'Exit' and terminated the packet capture.

Upon completion of these activities, the resulting transcript and packet capture were inspected. The evaluator confirmed that the communication using each specified remote administrative method was successful and that there was no data sent in plaintext.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

# 3. Protection Profile SAR Assurance Activities

The following sections address assurance activities specifically defined in the claimed Protection Profile that correspond with Security Assurance Requirements.

## 3.1 Development (ADV)

### 3.1.1 Basic Functional Specification (ADV_FSP.1)

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

The EAs presented in this section address the CEM work units ADV_FSP.1-1, ADV_FSP.1-2, ADV_FSP.1-3, and ADV_FSP.1-5.

The EAs are reworded for clarity and interpret the CEM work units such that they will result in more objective and repeatable actions by the evaluator. The EAs in this SD are intended to ensure the evaluators are consistently performing equivalent actions.

The documents to be examined for this assurance component in an evaluation are therefore the Security Target, AGD documentation, and any required supplementary information required by the cPP: no additional 'functional specification' documentation is necessary to satisfy the EAs. The interfaces that need to be evaluated are also identified by reference to the EAs listed for each SFR, and are expected to be identified in the context of the Security Target, AGD documentation, and any required supplementary information defined in the cPP rather than as a separate list specifically for the purposes of CC evaluation. The direct identification of documentation requirements and their assessment as part of the EAs for each SFR also means that the tracing required in ADV_FSP.1.2D (work units ADV_FSP.1-4, ADV_FSP.1-6 and ADV_FSP.1-7) is treated as implicit and no separate mapping information is required for this element.

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

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. The intent is that these interfaces will be adequately tested, and

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied.

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

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

The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

The evaluator uses the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 2, including the EAs associated with testing of the interfaces.

It should be noted that there may be some SFRs that do not have an interface that is explicitly 'mapped' to invoke the desired functionality. For example, generating a random bit string, destroying a cryptographic key that is no longer needed, or the TSF failing to a secure state, are capabilities that may be specified in SFRs, but are not invoked by an interface.

However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

The Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional 'functional specification' documentation is necessary to satisfy the Evaluation Activities specified in the SD.

## 3.2 Guidance documents (AGD)

### 3.2.1 Operational User Guidance (AGD_OPE.1)

**Assurance Activities**: The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps (per TD0536):

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

In addition, the evaluator performs the EAs specified below.

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

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

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

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

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

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

b) The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

1) Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).

2) Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes instructions that describe at least one method of validating the hash/digital signature.

c) The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The guidance documentation shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The [Admin Guide] provides detailed instructions for configuring the cryptographic algorithms and security related parameters used for the evaluated configuration. The [Admin Guide] includes notes of clarification throughout which make it clear which security functionality does not fall in the scope of the evaluation and which is covered by the Evaluation activities.

Section "Purpose" in the [Admin Guide] indicates that it was written to highlight the specific TOE configuration and administrator functions and interfaces that are necessary to configure and maintain the TOE in the evaluated configuration. The evaluated configuration is the configuration of the TOE that satisfies the requirements as defined in the Security Target (ST). It does not mandate configuration settings for the features of the TOE that are outside the evaluation scope, such as information flow polices and access control, which should be set according to your organizational security policies.

Section "Excluded Functionality" in the [Admin Guide] indicates that a "Non-FIPS 140-2 mode of operation" is excluded functionality in the evaluated configuration.

The "Product Updates" section in the [Admin Guide] provides a summary for performing software updates and refers to Section 2 "Secure Acceptance of the TOE".

The section "Secure Acceptance of the TOE" in the [Admin Guide] describes how the update is obtained via the Cisco.com website and downloaded onto a trusted computer system in order to make it available to the TOE. Once the file is downloaded, the authorized administrator verifies that it was not tampered with by using a hash utility to verify the SHA512 published hash by comparing the SHA512 published hash that is listed on the Cisco web site and in Table 9. Step 10 also describes how to check the current version of the TOE using the 'show version' command which displays the system software release version.

The "Secure Installation and Configuration" section in the [Admin Guide] provides the settings that need to be applied in order to ensure that the TOE is in its evaluated configuration. This includes steps for enabling FIPS Mode. The "Enabling FIPS Mode" section states that in the evaluated configuration, the TOE is run in the FIPS mode of operation which restricts the algorithms to ensure that only the permitted algorithms are available in the evaluated configuration. No other mode of operation was tested and this limits the TOE to only the cryptographic operations claimed by the Common Criteria evaluation.

### 3.2.2 Preparative Procedures (AGD_PRE.1)

**Assurance Activities**: As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

It is noted that specific requirements for Preparative Procedures are defined in [SD] for distributed TOEs as part of the Evaluation Activities for FCO_CPC_EXT.1 and FTP_TRP.1(2)/Join.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

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

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

In addition, the evaluator performs the EAs specified below.

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

The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

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

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

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

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

The preparative procedures must

a) include instructions to provide a protected administrative capability; and

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

The evaluation team had the following documents to use when configuring the TOE:

- Cisco Network Convergence System 1004 (NCS 1004) Running IOS-XR 24.1 Common Criteria Operational User Guidance and Preparative Procedures

In some instances, the document referenced general Cisco manuals which the evaluation could find on the Cisco web site. The completeness of the documentation is addressed by their use in the AA's carried out in the evaluation.

## 3.3 Life-cycle support (ALC)

### 3.3.1 Labelling of the TOE (ALC_CMC.1)

**Assurance Activities**: This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a 'hard label' (e.g., stamped into the metal, paper label) or a 'soft label' (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

The evaluator verified that the ST, TOE and Guidance are all labeled with the same hardware versions and software. The information is specific enough to procure the TOE and it includes hardware models and software versions. The evaluator checked the TOE software version and hardware identifiers during testing by examining the actual machines used for testing.

### 3.3.2 TOE CM Coverage (ALC_CMS.1)

**Assurance Activities**: Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

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

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

See section 3.3.1 above for an explanation of how all CM items are addressed.

## 3.4 Tests (ATE)

### 3.4.1 Independent Testing - Conformance (ATE_IND.1)

**Assurance Activities**: Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes 'evaluated configuration' instructions). The focus of the testing is to confirm that the requirements specified in Section 5.1.7 are being met. The Evaluation Activities in [SD] identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

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

Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section B.4.3.1.

The evaluator created a Detailed Test Report (DTR) to address all aspects of this requirement. The DTR discusses the test configuration, test cases, expected results, and test results. The test configuration consisted of the following TOE platforms along with supporting products.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

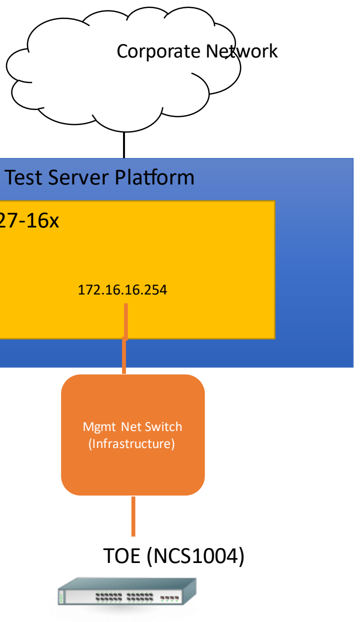
Corporate Network
Gossamer Test Server Platform
tl27-16x
172.16.16.254
Mgmt Net Switch (Infrastructure)
TOE (NCS1004)

**Figure 1 Test Setup**

**IP and MAC Addresses:**

| Device | IP Address | MAC Address |
|---|---|---|
| TOE: NCS1004 | | |
| Management | 172.16.16.10 | b0:26:80:ff:d9:b8 |
| Tl27-16x.example.com | | |
| Management | 172.16.16.254 | 00:15:5d:90:19:03 |

**TOE Platforms:**

- NCS 1004

**Supporting Software:**

- Windows 10 and 11 (Evaluator Laptops)
- SSH Client – Putty version 8.1
- Wireshark version 4.4.3

The Gossamer Test servers with an Ubuntu environment acted as platforms to initiate testing. The test servers also acted as a syslog server.

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

- Openssh-client version 8.2p1
- Big Packet Putty, Various versions
- Nmap version 7.01
- Tcpdump version 4.9.3
- Libpcap version 1.7.4
- Stunnel version 5.30
- Openssl version 1.0.2g
- Rsyslog version 8.16.0

## 3.5 Vulnerability assessment (AVA)

### 3.5.1 Vulnerability Survey (AVA_VAN.1)

**Assurance Activities**: While vulnerability analysis is inherently a subjective activity, a minimum level of analysis can be defined and some measure of objectivity and repeatability (or at least comparability) can be imposed on the vulnerability analysis process. In order to achieve such objectivity and repeatability it is important that the evaluator follows a set of well-defined activities, and documents their findings so others can follow their arguments and come to the same conclusions as the evaluator. While this does not guarantee that different evaluation facilities will identify exactly the same type of vulnerabilities or come to exactly the same conclusions, the approach defines the minimum level of analysis and the scope of that analysis, and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities.

In order to meet these goals some refinement of the AVA_VAN.1 CEM work units is needed. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

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

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

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

The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis. (TD0547 applied)

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

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

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

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.5.1.2 and 3.6.1.2.

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

The vulnerability analysis is in the Detailed Test Report (DTR) prepared by the evaluator. The vulnerability analysis includes a public search for vulnerabilities and fuzz testing. None of the public search for vulnerabilities or the fuzz testing uncovered any residual vulnerability.

The evaluation team performed a public search for vulnerabilities in order to ensure there are no publicly known and exploitable vulnerabilities in the TOE from the following sources:

National Vulnerability Database (https://web.nvd.nist.gov/vuln/search),

Vulnerability Notes Database (http://www.kb.cert.org/vuls/),

Rapid7 Vulnerability Database (https://www.rapid7.com/db/vulnerabilities),

Tipping Point Zero Day Initiative (http://www.zerodayinitiative.com/advisories ),

Tenable Network Security (http://nessus.org/plugins/index.php?view=search),

Offensive Security Exploit Database (https://www.exploit-db.com/)

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.

The search was performed on 2/27/2025 (from 1/1/2020). The search was conducted with the following search terms: "IOS-XR", "Cisco NCS", "Network Convergence System", "SSH", "TLS", "Atom C3758", "Cisco FIPS Object Module".

© 2025 Gossamer Security Solutions, Inc.
All rights reserved.