Assurance Activity Report

Bivio Networks, Inc.
Bivio 6310-NC

VID 11106

20-5018-R-0020 V1.2
December 2, 2020

Evaluated by:

UL Verification Services Inc.
709 Fiero Lane, Suite 25
San Luis Obispo, CA 93401

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

TOE Evaluation Sponsor and Developer
Bivio Networks, Inc.
4457 Willow Rd Suite 240
Pleasanton, CA, 94588

ST Author:
Ekta Binwani
UL Verification Services Inc.
709 Fiero Lane, Suite 25
San Luis Obispo, CA 93401

Evaluation Personnel:
Oleg Andrianov
Michael C. Baron
Gerrit Kruitbosch

Applicable Common Criteria Version
CC Version 3.1 R5, April 2017

Common Evaluation Methodology Version
CEM Version 3.1 R5, April 2017

Applicable Common Criteria Protection Profiles
collaborative Protection Profile for Network Devices
Version 2.2e, March 23, 2020

Table of Contents

1 Overview

This document presents evaluation results of the Bivio 6310-NC against the collaborative Protection Profile for Network Devices, Version 2.2e, March 23, 2020 [PP]. This document contains a description of the assurance activities and associated results as performed by UL, an accredited Common Criteria Testing Laboratory. This Evaluation was conducted with the oversight and guidance provided by the National Information Assurance Partnership and its contributors.

Abbr.	Title	Version	Date
Compliant Protection Profiles			
[PP]	collaborative Protection Profile for Network Devices	2.2e	March 23, 2020
[SD]	Supporting Document Mandatory Technical Document. Evaluation Activities for Network Device cPP	2.2	December-2019
Evidence			
[ST]	Bivio 6310-NC Security Target	0.8	November 25, 2020
[AGD]	Bivio 6310-NC Common Criteria Administrative Guidance	1.10	November 23, 2020
[EQV]	Bivio 6310-NC Equivalence Justification for Platform Testing	1.0	September 28, 2020
Supporting Documentation			
[CEM]	Common Methodology for Information Technology Security Evaluation – Evaluation methodology CCMB-2017-04-004	3.1r5	April 2017
Technical Decisions			
	TD applied through 0556		November 23, 2020

1.1 Test Equivalency

The evaluator performed testing on Model number B6310-NC-C15M1D1N1. The [ST] includes multiple hardware models, which were determined by the evaluator to be equivalent for the purpose of this evaluation. Differences between hardware models do not affect TOE security functionality as per equivalency argument analysis. A single build of software is executed on all hardware models.

1.2 Test Environment

The test environment used by the CCTL during the course of testing is briefly summarized below and conforms to the expected use-case of the TOE (Network Device).

The evaluation team performed the independent testing activities to confirm the TOE operates to the TOE security functional requirements as specified in the [ST] for a product claiming conformance to [PP]. The evaluation team devised a Test Plan based on the Testing Assurance Activities specified in [PP]. The Test Plan described how each test activity was to be performed. The evaluation team executed the tests specified in the Test Plan and documented the results in the Evaluation Technical Report. The evaluation team consisted of Oleg Andrianov and Michael Baron from the CCTL.

The test laboratory was configured by UL and physically located at the UL San Luis Obispo facility in an access-controlled environment.

The CC test environment consists of the following physical components:

Number	Device	Purpose
1	HP ProLiant Blade Server	ESXi 6.0 Server w/8 LAN ports
2	Aruba 2920-24G 24-port layer-3-aware Ethernet switch	Infrastructure Routing Switch
3	Dell PowerEdge Desktop Workstation running Ubuntu 18.04	Management & Control console

The test lab can present three separate test environments (referred to as strings) via the extensive use of virtualization. Access to the environments is through the management console.

The management console provides the access point for the evaluator either through physical access in the lab, via SSH, or by remote desktop. These provide the ability to subsequently connect to the various devices in each string without exposing the entire environment to the external network. Each workstation has one network interface on the test network and a second interface connected to the Aruba 2920 switch. The test string was configured for one network segment:

- Management: network segment used for remote (either public or private facing subnet) administration of the TOE and the VMs in the environment. Network has IPv4 address 192.168.3.0/255.255.255.0.

The HP ProLiant blade server provides virtual machines for the test string, containing the following virtual hosts:

- Server 1 – Ubuntu 18.04 Linux, supporting an Apache web server for distributing CRLs and responding to other web requests. Also provides syslog, NTP, and remote SSH access to the rest of the hosts in the test environment via their administrative / management interfaces, as well as SSH and TLS clients for testing.
- Server 2 – Ubuntu 18.04 Linux, NTP, and remote SSH access to the rest of the hosts in the test environment via their administrative / management interfaces, as well as SSH and TLS clients for testing.
- IT Support – Ubuntu 16.04 Linux. The IT support machine also provides syslog, NTP, and remote SSH access to the rest of the hosts in the test environment via their administrative / management interfaces, as well as SSH and TLS clients for testing.
- Kali Linux- A Kali Linux 2020 host which serves as a vulnerability assessment and penetration testing tool. This can perform many functions duplicated by other VMs, such as packet capture, but also provides a robust and flexible pen-testing platform.

Test Machines inside the test string do not have internet access. System time was manually updated at the beginning of the evaluation.

Figure 1 – Test Network Configuration shows the logical configuration used for the ATE.

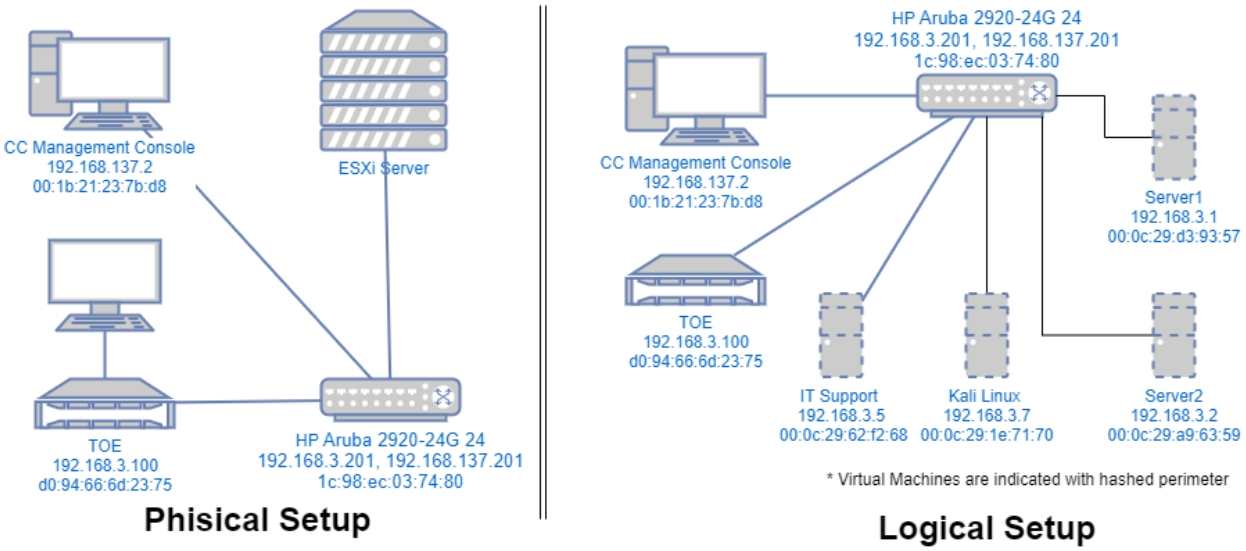

Figure 1 - Test Network Configuration

The evaluator used a following software testing tools:

1. Wireshark 3.2.6
2. Nmap 7.80
3. OpenSSL 1.1.1
4. OpenSSH-based UL test tool (ulSSHv0.4.1)
5. S2n-based UL test tool
6. Robot attack script (https://github.com/robotattackorg/robot-detect)
7. A number of python-based scripts facilitating testing of x.509, NTP and conducting protocol fuzzing

2 SFR Assurance Activities and Results

2.1 FAU_GEN.1 Audit Data Generation

The main reasons for collecting audit information are to detect and identify error conditions, security violations, etc. and to provide sufficient information to the Security Administrator to resolve the issue. The audit information to be collected according to FAU_GEN.1, and the failure conditions identified in tables 2, 4, and 5 need to enable the Security Administrator at least to detect and identify the problem and provide at least basic information to resolve the issue. Also for this level of detail, the other FAU requirements apply, in particular the need for local and remote storage of audit information according to FAU_STG_EXT.1.

The level of detail that needs to be provided to the Security Administrator to actually resolve an issue usually depends on the complexity of the underlying use case. It is expected that a product provides additional levels of auditing to support resolution of error conditions, security violations, etc. beyond the level required by FAU_GEN.1, but it should also be clear that a high level of granularity cannot be maintained on most systems by default due to the high number of audit events that would be generated in such a configuration. It is expected that the TOE will be capable of auditing sufficient information to meet the requirements of FAU_GEN.1. This may include audits that are generated only when configured if the TOE configuration can be modified without taking the TOE out of the evaluated configuration.

The issue described above explicitly refers to the use of X.509 certificates. In case a certificate-based authentication fails, an error message telling the Security Administrator that ‘something is wrong with the certificate’ shall not be considered as sufficient information about the ‘reason for failure’ as a basic information to resolve the issue. The log message will inform the Security Administrator of at least the following:

- ‘Trust issue’ with the certificate, e.g. due to failed path validation
- Use of an ‘expired certificate’
- Absence of basicConstraints extension
- CA flag not set for a certificate presented as a CA
- Signature validation failure for any certificate in the certificate path; failure to establish revocation status; revoked certificate

As such for audit information related to the use of X.509 certificates that it uniquely identifies the certificate that could not be successfully verified. For example, identification of a certificate could include Key Subject and Key ID, where key subject is an identifier contained in the CN or SAN and where Key ID is a certificate's serial number and issuer name or subject key identifier (SKI) and authority key identifier (AKI).

In general, when using open source libraries like OpenSSL, passing on error messages from such libraries to the Security Administrator is regarded as good practice.

TSS

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.

Guidance Documentation

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.

Tests

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.

2.1.1 TSS

The TOE is not a distributed TOE.

[ST] Section 7.1.1 describes that the unique key name or key reference shall be logged for operations of generating, import of, changing or deletion of cryptographic keys.

2.1.2 Guidance Documentation

[AGD] Section 9 contains examples for all auditable events required by the SFR. For every auditable event those examples contains all required information.

2.1.3 Tests

Test Number	Test 1

Test Objective	Verify the TOE generates audit records for events as required by the SFR.
Test Steps Performed	The evaluator performed every operation for which audit record data is required to be generated by the TOE and verified that for each event the required information was generated by the TOE.
Test Result	Pass

2.2 FAU_GEN.2 User Identity Association

TSS & Guidance Documentation

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

Tests

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

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

2.2.1 TSS

See Section 2.1.1.

2.2.2 Guidance Documentation

See Section 2.1.2.

2.2.3 Tests

Test Number	Test 1
Test Objective	This test was accomplished in conjunction with testing FAU_GEN.1.1. A missing subject identity in FAU_GEN.1.1 validation will constitute a failure under this SFR. This is not a distributed TOE.
Test Steps Performed	The evaluator determined there was a subject identified in every audit record examined during the evaluator activity for FAU_GEN.1.
Test Result	Pass

2.3 FAU_STG.1 Protected Audit Trail Storage (Optional)

TSS

The evaluator shall examine the TSS to ensure it describes the amount of audit data that are stored locally and how these records are protected against unauthorized modification or deletion. The evaluator

shall ensure that the TSS describes the conditions that must be met for authorized deletion of audit records.

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

Guidance Documentation

The evaluator shall examine the guidance documentation to determine that it describes any configuration required for protection of the locally stored audit data against unauthorized modification or deletion.

Tests

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall access the audit trail without authentication as Security Administrator (either by authentication as a non- administrative user, if supported, or without authentication at all) and attempt to modify and delete the audit records. The evaluator shall verify that these attempts 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 access the audit trail 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.

b) Test 2: The evaluator shall access the audit trail as an authorized administrator and attempt to delete the audit records. The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records authorized for deletion are deleted.

For distributed TOEs the evaluator shall perform test 1 and test 2 for each component that is defined by the TSS to be covered by this SFR.

2.3.1 TSS

[ST] Section 7.1.2 describes that whole partition is available as a local log storage. It is stated that those files require users to be authorized as administrators to get access to the logs.

The TOE is not a distributed TOE.

2.3.2 Guidance Documentation

[AGD] Section 9 “Security Audit Data Generation” contains the description of the audit logging process. No configuration is required for protection of the locally stored audit data against unauthorized modification or deletion.

[ST] Section 7.1.1 states that only an authorized administrator can read log files, delete log files, modify log files, or archive log files through the CLI, and such actions require being first authenticated as an authorized administrator using a Trusted Path.

[AGD] Section 9 is consistent with that claim, providing a description for accessing log files only through the interface and requiring authentication; moreover, commands to access log files as described in Section 9 of the [AGD] will require additional elevation of privileges via “sudo” command as described in Section 6 of the [AGD].

2.3.3 Tests

Test Number	Test 1

Test Objective	Verify users are unable to access the audit trail without prior authentication as a security administrator.
Test Steps Performed	The TOE does not support non-administrative users, testing for non-authenticated users are satisfied by testing in FMT_MOF.1.1/Functions – Test 1
Test Result	Pass

Test Number	Test 2
Test Objective	Verify a user with administrative privileges is able to delete audit records.
Test Steps Performed	1. Authenticate as a Security Administrator and attempt to remove audit records. 2. Verify that audit records were removed.
Test Result	Pass

2.4 FAU_STG_EXT.1 Protected Audit Event Storage

TSS

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

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

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

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

The 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.

Guidance Documentation

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.

Tests

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

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

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.

2.4.1 TSS

The TOE is a standalone TOE that stores audit data locally and on external audit server.

[ST] Section 7.1.1 describes that audit data are transferred to an external audit server via SSH trusted channel in real-time.

[ST] Section 7.1.2 describes that audit data can be stored locally in /var/log. Whole partition (effectively about 50 GB of storage space) is available for local audit storage, but an Administrator can configure this amount. These records can be accessed only by authorized administrators.

[ST] Section 7.1.2 describes that when the audit log partition falls below the configured “space_left” parameter, the TOE will stop storing audit records locally and will only send them to the remote syslog server.

[ST] Section 7.1.2 describes that audit data are protected from unauthorized access to the log files because to access log files remote entities would have to be authenticated as administrators using a trusted path.

2.4.2 Guidance Documentation

[AGD] Section 10 describes configuring SSH tunnel for transporting audit logs to the remote server.

[AGD] Section 2 describes that TOE will require, “Syslog server conformant to RFCs 5424 (Syslog over TCP), capable of receiving an SSH tunnel from the Bivio 6310-NC.”

[AGD] Section 10 describes that audit data are forwarded to the remote audit log server simultaneously with storing them locally.

[AGD] Section 2 contains the requirement for the remote audit server “Syslog server conformant to RFCs 5424 (Syslog over TCP), capable of receiving an SSH tunnel from the Bivio 6310-NC”. [AGD] Sections 10 and 13 describe SSH tunnel parameters that need to be supported for the TOE to be able to establish this connection.

[ST] Section 6.1.1.4 states that the TOE will drop new audit data and send said audit data to an external IT entity when local storage space is full.

[AGD] Section 9 contains a description of this behaviour by describing the “space_left” parameter which is used to determine when the TOE considers local storage space full. This description is consistent with the description in the TSS (Section 7.1.2 of the [ST]).

2.4.3 Tests

Test Number	Test 1
Test Objective	Objective 1 - Verify the audit data from the TOE is being sent as encrypted data. Objective 2 - Verify audit logs are sent automatically and without administrator intervention.
Test Steps Performed	Objective 1 is satisfied by the testing performed in FTP_ITC.1 Inter-TSF Trusted Channel – Test 3. Objective 2:

	1. Initiate auditable event on the TOE without authenticating to the TOE. 2. Verify corresponding audit records were automatically transferred to the remote audit server.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify audit data is stored locally (storage space permitting), and TSF continues to send the audit data to an external IT entity when the local storage space for audit data is full.
Test Steps Performed	1. Inspect local audit files for records for auditable events. Verify local audit files contain required records. 2. Force local storage space for audit log to run out. 3. Verify audit events are no longer stored in the local storage. 4. Verify those records are stored in the remote audit server.
Test Result	Pass

Test Number	Test 3
Test Objective	[ST] does not contain FAU_STG_EXT.2/LocSpace SFR. Test is automatically satisfied.
Test Steps Performed	None.
Test Result	Pass

Test Number	Test 4
Test Objective	TOE is not a distributed TOE. Test is automatically satisfied.
Test Steps Performed	None.
Test Result	Pass

2.5 FAU_STG_EXT.3/LocSpace Action in Case of Possible Audit Data Loss (Optional)

This activity should be accomplished in conjunction with the testing of FAU_STG_EXT.1.2 and FAU_STG_EXT.1.3.

TSS

The evaluator shall examine the TSS to ensure that it details how the Security Administrator is warned before the local storage for audit data is full.

For distributed TOEs the evaluator shall examine the TSS to ensure it describes to which TOE components this SFR applies and how each TOE component realises this SFR. Since this SFR is optional, it might only apply to some TOE components but not all. This might lead to the situation where all TOE components store their audit information themselves but FAU_STG_EXT.3/LocSpace is supported only by one of the components. In particular, the evaluator has to verify, that the TSS describes for every component supporting this functionality, whether the warning is generated by the component itself or through another component and name the corresponding component in the latter case. The evaluator has to verify that the TSS makes clear any situations in which audit records might be 'invisibly lost'.

Guidance Documentation

The evaluator shall also ensure that the guidance documentation describes how the Security Administrator is warned before the local storage for audit data is full and how this warning is displayed or stored (since there is no guarantee that an administrator session is running at the time the warning is issued, it is probably stored in the log files). The description in the guidance documentation shall correspond to the description in the TSS.

Tests

The evaluator shall verify that a warning is issued by the TOE before the local storage space for audit data is full.

For distributed TOEs the evaluator shall verify the correct implementation of display warning for local storage space for all TOE components that are supporting this feature according to the description in the TSS. The evaluator shall verify that each component that supports this feature according to the description in the TSS is capable of generating a warning itself or through another component.

2.5.1 TSS

[ST] Section 7.1.2 describes that the TOE will place a notice to syslog events when the systems run low on disk space allocated for audit data. This warning will not be stored in a local audit log file.

The TOE is a standalone TOE.

2.5.2 Guidance Documentation

[AGD] Section 9, subsection “Storage space for audit data”, contains configuration options for low storage space warning, guidance to manage it and related recommendations. It also describes the space_left parameter. This warning is issued to the System Administrator as a message in remote syslog.

[AGD] Section 9, subsection “Audit storage space running low – warning” contains an example of the audit record generated for this warning.

This description is consistent with the description in the TSS (Section 7.1.2 of the [ST]).

2.5.3 Tests

Test Number	Test 1
Test Objective	Generate audit data sufficient to fill data storage then verify a warning is issued by the TOE before the space is full.
Test Steps Performed	1. Force local storage space for audit log to run out. 2. Verify a ‘low storage’ warning message was sent to the remote audit server by the TOE.
Test Result	Pass

2.6 FCS_CKM.1 Cryptographic Key Generation

TSS

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.

Guidance Documentation

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.

Tests

Note: The following tests require the developer to provide access to a test platform that provides the evaluator with tools that are typically not found on factory products. Generation of long-term cryptographic keys (i.e. keys that are not ephemeral keys/session keys) might be performed automatically (e.g. during initial start-up). Testing of key generation must cover not only administrator invoked key generation but also automated key generation (if supported).

Key Generation for FIPS PUB 186-4 RSA Schemes

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

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

a) Random Primes:
   - Provable primes
   - Probable primes

b) Primes with Conditions:
   - Primes p1, p2, q1, q2, p and q shall all be provable primes
   - Primes p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes
   - Primes p1, p2, q1, q2, p and q shall all be probable primes

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

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

FIPS 186-4 Public Key Verification (PKV) Test

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

Key Generation for Finite-Field Cryptography (FFC)

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

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

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

and two ways to generate the cryptographic group generator g:

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

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

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

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

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

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

- g != 0,1
- q divides p-1
- g^q mod p = 1
- g^x mod p = y

for each FFC parameter set and key pair.

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

Testing for FFC Schemes using Diffie-Hellman group 14 and/or safe-prime groups is done as part of testing in CKM.2.1.

2.6.1 TSS

[ST] Section 7.2.1 identifies cryptographic key generation schemes, key sizes and their usage for SSH and TLS tunnels. Specifically, RSA 2048-bit key generation, Diffie-Hellman group 14 keys of 2048 bits are used for SSH, and ECC scheme with P-256 curve keys and RSA keys of 2048 bits are used for TLS.

2.6.2 Guidance Documentation

[ST] Section 6.1.2.1 contains selections for RSA, ECC based on NIST curve p-256, and FFC Schemes using "safe-prime" groups key generation algorithms. [ST] Section 7.2.1 identifies FFC safe-primes used by the TOE as Diffie-Hellman Group 14.

The evaluator discovered that [AGD] contains information about configured key generation schemes and key sizes, but they are already configured when the TOE is shipped from the factory and [AGD] instructs an Administrator not to change them from pre-configured values.

[AGD] Section 12 describes settings for key generation as related to SSH Server functionality (Diffie-Hellman group 14 for key exchange, RSA host key configuration and generation).

[AGD] Section 13 describes settings for key generation as related to SSH Client functionality (Diffie-Hellman group 14 for key exchange).

[AGD] Sections 10 and 11 describe RSA 2048-bit key generation for SSH Client functionality for remote audit server and user authentication.

[AGD] Section 14 describes key generation scheme for TLS Server functionality, using ECHDE with prime256v1. Section 15 describes RSA 2048-bit certificate generation for TLS Server functionality.

2.6.3 Tests

[ST] Section 6.1.2.1 has selection “RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3”. This is consistent with the information listed in CAVP Certificate #C1935. This part of the test is considered satisfied by CAVP certification.

[ST] Section 6.1.2.1 has selection “ECC schemes using “NIST curves” P-256 that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4”. This is consistent with the information listed in CAVP Certificate #C1935. This part of the test is considered satisfied by CAVP certification.

[ST] Section 6.1.2.1 has selection FFC Schemes using ’safe-prime’ groups that meet the following: “NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and RFC 3526. This testing is done as part of FCS_CKM.2.1.

2.7 FCS_CKM.2 Cryptographic Key Establishment

TSS

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

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

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

Scheme	SFR	Service
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
Diffie- Hellman (Group 14)	FCS_SSHC_EXT.1	Backup Server
ECDH	FCS_IPSEC_EXT.1	Authentication Server

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

Guidance Documentation

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

Tests

Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

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

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

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

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

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

Validity Test

The Validity test verifies the ability of the TOE to recognize another party’s valid and invalid key agreement results with or without key confirmation. To conduct this test, the evaluator shall obtain a list of the supporting cryptographic functions included in the SP800-56A key agreement implementation to determine which errors the TOE should be able to recognize. The evaluator generates a set of 24 (FFC) or 30 (ECC) test vectors consisting of data sets including domain parameter values or NIST approved

curves, the evaluator's public keys, the TOE's public/private key pairs, MACTag, and any inputs used in the KDF, such as the other info and TOE id fields.

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

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

RSA-based key establishment

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

Diffie-Hellman Group 14

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

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.

2.7.1 TSS

[ST] Section 7.2.1. describes that Diffie-Helman key establishment is used for SSH protocol, while ECDH Key establishment and RSA key establishment is used for TLS tunnel as follows:

Scheme	SFR	Service
Diffie-Hellman Group-14 SHA-1	FCS_SSHS_EXT.1 FCS_SSHC_EXT.1	1. Administration via SSH v2 2. Audit Server (uses SSH tunnel)
RSA ECDHE_RSA	FCS_TLSS_EXT.1	Administration via TLS-v1.2

[ST] Section 7.2.1 states that Diffie-Hellman key exchange is implemented as per RFC 3526 Section 3.

[ST] Section 7.2.1 states that RSA- based key establishment is preformed according to 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.

2.7.2 Guidance Documentation

[ST] Section 6.1.2.2 contains selections for RSA, ECC, Finite-field based key establishment and FFC Schemes using "safe-prime" groups establishment schemes. [ST] Section 7.2.1 identifies FFC safe-primes used by the TOE as Diffie-Hellman Group 14.

The evaluator discovered that [AGD] contains information about configured key generation schemes and key sizes, but they are already configured when the TOE is shipped from the factory and [AGD] instructs the Administrator not to change them from pre-configured values. [AGD] Section 12 describes settings for key establishment scheme as related to SSH Server functionality (Diffie-Hellman group 14 for key exchange).

[AGD] Section 13 describes settings for key establishment scheme as related to SSH Client functionality (Diffie-Hellman group 14 for key exchange).

[AGD] Section 14 describes setting up key establishment scheme for TLS Server functionality, using ECHDE with prime256v1 and RSA key exchange.

2.7.3 Tests

[ST] Section 6.1.2.2 contains selection “RSA-based key establishment schemes that meet the following: 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 correctness of the TSF implementation is verified against known-good implementations:

- For RSA-based key establishment the testing is satisfied by testing of FCS_TLSS_EXT.1.1 Test 1.

[ST] Section 6.1.2.2 contains selection “Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography””. This is consistent with the information listed in CAVP certificate #C1935. This part of the test is considered satisfied by CAVP certification.

[ST] Section 6.1.2.2 contains selection “FFC Schemes using “safe-prime” groups that meet the following: ‘NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and RFC 3526.” [ST] Section 7.2.1 identifies FFC safe-primes used by the TOE as Diffie-Hellman Group 14. No CAVP test exists for this cryptographic selection. The correctness of the TSF implementation is verified against known-good implementations:

- For SSH-based trusted channel safe-primes key exchange the testing is satisfied by testing of FCS_SSHC_EXT.1.7 Test 1.
- For SSH-based trusted path safe-primes key exchange the testing is satisfied by testing of FCS_SSHS_EXT.1.7 Test 2.

2.8 FCS_CKM.4 Cryptographic Key Destruction

TSS

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

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

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.

Guidance Documentation

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

2.8.1 TSS

[ST] Section 7.2.1 states that temporary session keys that are established or generated when SSH or TLS sessions are created, reside in RAM and are not stored outside of RAM.

[ST] Section 7.2.1 states that the storage location of all persistent keys for SSH and TLS Server is in TOE filesystem (non-volatile).

[ST] Section 7.2.1 states that zeroization of keys in RAM happens when the associated session is ended and is performed by overwriting with zeroes, followed by a read-verify. If verification fails, the zeroization operation is performed again.

[2] 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).

[ST] Section 7.2.1 states that persistent keys stored in non-volatile storage are destroyed by block erase, followed by read-verify. If verification fails, the zeroization operation is performed again. This zeroization is performed by an Administrator command.

2.8.2 Guidance Documentation

[AGD] Sections 14 and 15 describe key zeroization when generating a key for the TLS Server. This is done by invoking zeroize-keyfile commands.

[AGD] Sections 12 and 11 describe key zeroization when generating keys for the SSH Server and Client using the zeroize-keyfile command.

[AGD] does not describe a situation where this operation could be delayed or may not strictly conform to the requirement. Section 7.2.1 of the [ST] describes that specific block erase is being performed during zeroization.

2.8.3 Tests

None.

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

TSS

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.

Guidance Documentation

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.

Tests

AES-CBC Known Answer Tests

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

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

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

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

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

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

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

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

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

AES-CBC Multi-Block Message Test

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

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

AES-CBC Monte Carlo Tests

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

```
# Input: PT, IV, Key
for i = 1 to 1000:
        if i == 1:
                CT[1] = AES-CBC-Encrypt(Key, IV, PT)
                PT = IV
        else:
                CT[i] = AES-CBC-Encrypt(Key, PT)
                PT = CT[i-1]
```

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

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

AES-GCM Test

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

128 bit and 256 bit keys

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

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

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

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

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

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

AES-CTR Known Answer Tests

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

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

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

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

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

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

AES-CTR Multi-Block Message Test

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

AES-CTR Monte-Carlo Test

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

```
# Input: PT, Key
for i = 1 to 1000:
CT[i] = AES-ECB-Encrypt(Key, PT) PT = CT[i]
```

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

There is no need to test the decryption engine.

2.9.1 TSS

[ST] Section 7.2.2 describes that data encryption and decryption is performed by the TOE using AES in CBC and GCM modes, using key sizes of 128 and 256 bits.

2.9.2 Guidance Documentation

[AGD] Section 12 contains a description of an encryption keys configuration for SSH Server, but it is noted that the Security Administrator must not change anything.

[AGD] Section 13 contains a description of an encryption keys configuration for SSH Client, but it is noted that the Security Administrator must not change anything.

[AGD] Section 14 contains a description of the encryption keys used in TLS Server (as part of ciphersuites), but it is noted that the Security Administrator must not change anything.

2.9.3 Tests

[ST] Section 6.1.2.4 claims AES modes of CBC, GCM using key sizes of 128-bits and 256-bits. This information is consistent with the information listed in CAVP Certificate #C1935. This test is considered satisfied by CAVP certification.

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

TSS

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

Guidance Documentation

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.

Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

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

ECDSA FIPS 186-4 Signature Verification Test

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

RSA Signature Algorithm Tests

Signature Generation Test

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

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

Signature Verification Test

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

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

2.10.1 TSS

[ST] Section 7.2.2 describes that TOE uses RSA-based digital signatures with key length of 2048 bits for all TOE digital signature purposes.

2.10.2 Guidance Documentation

[ST] Section 7.2.2 states that the TOE does not allow the configuration of cryptographic algorithms and key sizes for signature services.

The [AGD] does not contain instructions for the configuration of cryptographic algorithms and key sizes for signature services.

2.10.3 Tests

[ST] Section 6.1.2.5 contains selection “RSA Digital Signature Algorithm and cryptographic key sizes (modulus) 2048 bits, that meet the following: For RSA schemes: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3”. This is consistent with information listed in CAVP Certificate #C1935. This test is satisfied by CAVP certification.

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

TSS

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.

Guidance Documentation

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

Tests

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

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

Short Messages Test - Bit-oriented Mode

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

Short Messages Test - Byte-oriented Mode

The evaluators devise an input set consisting of m/8+1 messages, where m is the block length of the hash algorithm. The length of the messages range sequentially from 0 to m/8 bytes, with each message being an integral number of bytes. The message text shall be pseudorandomly generated. The evaluators

compute the message digest for each of the messages and ensure that the correct result is produced when the messages are provided to the TSF.

Selected Long Messages Test - Bit-oriented Mode

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

Selected Long Messages Test - Byte-oriented Mode

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

Pseudorandomly Generated Messages Test

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

2.11.1 TSS

[ST] Section 7.2.2 describes that hash functions are used for RSA digital signatures, NTP message authentication, keyed-hash message authentication in SSH and TLS sessions, password hashing and file integrity checking. Table 6 in [ST] Section 7.2 also lists hash functions and their usages.

2.11.2 Guidance Documentation

The evaluator determined that [AGD] does not contain instructions for configuration of hash sizes, nor this configuration is required as hash sizes are not configurable.

Hash of corresponding size is being used automatically (for example sha256sum command used as prescribed in [AGD] Section 19 will invoke hash size of 256).

[AGD] Section 4 describes that relevant ID/Key pairs and hash function must be used for NTP messages authentication.

2.11.3 Tests

[ST] Section 6.1.2.6 has selections for “SHA-1, SHA-256, SHA-384, and SHA-512”. This information is consistent with the information listed in CAVP Certificate #C1935. This test is satisfied by CAVP certification.

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

TSS

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.

Guidance Documentation

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.

Tests

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.

2.12.1 TSS

[ST] Section 7.2.2 specifies cryptographic key length, digest sizes and block sizes for HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-384 and HMAC-SHA-512. HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 functions use cryptographic key sizes of 160, 256, 384, 512, respectively. HMAC message digest sizes of 160, 256, 384, or 512 bits are used, with block sizes of 64 bytes for SHA-1 and SHA-256, and 128 bytes for SHA-384 and SHA-512 hash functions.

2.12.2 Guidance Documentation

[AGD] Section 12 contains a description of a keyed hash configuration for SSH Server, but it is noted that the Security Administrator must not change anything.

[AGD] Section 13 contains a description of a keyed hash configuration for SSH Client, but it is noted that the Security Administrator must not change anything.

[AGD] Section 14 contains a description of a keyed hash used in TLS Server (as part of ciphersuites), but it is noted that the Security Administrator must not change anything.

2.12.3 Tests

[ST] Section 6.1.2.7 claims “HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512”. This information is consistent with the information listed in CAVP Certificate #C1935. This test is satisfied by the CAVP Certification.

2.13 FCS_NTP_EXT.1 NTP Protocol (Selection-Based)[3]

TSS

FCS_NTP_EXT.1.1

The evaluator shall examine the TSS to ensure it identifies the version of NTP supported, how it is implemented and what approach the TOE uses to ensure the timestamp it receives from an NTP timeserver (or NTP peer) is from an authenticated source and the integrity of the time has been maintained.

The TOE must support at least one of the methods or may use multiple methods, as specified in the SFR element 1.2. The evaluator shall ensure that each method selected in the ST is described in the TSS, including the version of NTP supported in element 1.1, the message digest algorithms used to verify the authenticity of the timestamp and/or the protocols used to ensure integrity of the timestamp.

Guidance Documentation

FCS_NTP_EXT.1.1

[3] Test 1 and Test 2 added to FCS_NTP_EXT.1 by TD0528

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

FCS_NTP_EXT.1.2

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

Assurance Activity Note:

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

FCS_NTP_EXT.1.3

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

Tests

FCS_NTP_EXT.1.1

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

FCS_NTP_EXT.1.2

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

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

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

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

FCS_NTP_EXT.1.3

The evaluator shall configure NTP server(s) to support periodic time updates to broadcast and multicast addresses. The evaluator shall confirm the TOE is configured to not accept broadcast and multicast NTP

packets that would result in the timestamp being updated. The evaluator shall check that the time stamp is not updated after receipt of the broadcast and multicast packets.

FCS_NTP_EXT.1.4 [4]

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

Test 2: (The intent of this test is to ensure that the TOE would only accept NTP updates from configured NTP Servers).
The evaluator shall confirm that the TOE would not synchronize to other, not explicitly configured time sources by sending an otherwise valid but unsolicited NTP Server responses indicating different time from the TOE's current system time. This rogue time source needs to be configured in a way (e.g. degrade or disable valid and configured NTP servers) that could plausibly result in unsolicited updates becoming a preferred time source if they are not discarded by the TOE. The TOE is not mandated to respond in a detectable way or audit the occurrence of such unsolicited updates. The intent of this test is to ensure that the TOE would only accept NTP updates from configured NTP Servers. It is up to the evaluator to craft and transmit unsolicited updates in a way that would be consistent with the behaviour of a correctly-functioning NTP server.

2.13.1 TSS

[ST] Section 7.2.3 identifies the NTP version as version 4. The TOE implements NTP from a crony package.

[ST] Section 6.1.2.8 contains FCS_NTP_EXT.1.2 with SHA1, SHA256, SHA512 message digest selected as the authentication method.

[ST] Section 7.2.3 describes that the TOE uses SHA1, SHA 512 and SHA256 for calculating message digests to verify timestamp authenticity using NTP "Symmetric Key" method. Keys used in authentication are configurable.

2.13.2 Guidance Documentation

FCS_NTP_EXT.1.1

The [AGD] does not contain instruction on how to configure the version of NTP, because only one version is supported by the TOE.

[AGD] Section "Enabling the NTP Client for Time Synchronization" contains descriptions on how to configure multiple NTP servers for the time source.

FCS_NTP_EXT.1.2

[ST] Section 6.1.2.8 FCS_NTP_EXT.1.2 states:

"The TSF shall update its system time using

- Authentication using: SHA1, SHA256, SHA512 as the message digest algorithm(s);"

[4] TD0528 was applied.

[AGD] Section 4, subsection “Enabling the NTP Client for Time Synchronization” contains a description of how to configure SHA1, SHA256 or SHA512 “keys” that are being used for NTP message authentication.

FCS_NTP_EXT.1.3

The [AGD] does not contain instructions on how to disable broadcast and multicast NTP packets. The evaluator determined that this is not a failure, as the TOE NTP client does not support broadcast and multicast packets.

2.13.3 Tests

FCS_NTP_EXT.1.1

Test Number	Test 1
Test Objective	Verify the NTP protocol version used by the TOE.
Test Steps Performed	This test is satisfied by FCS_NTP_EXT.1.2 – Test 1.
Test Result	Pass

FCS_NTP_EXT.1.2

Test Number	Test 1
Test Objective	Verify that NTP functionality functions as claimed in [ST].
Test Steps Performed	1. Configure the TOE NTP client as per [AGD] and observe the TOE clock was updated as expected. 2. Verify the TSF generated audit records for the time update event. 3. Configure the NTP server to use a different message authentication algorithm. 4. Verify the TOE does not update the time if the message authentication algorithm does not match the algorithm configured on the NTP server. 5. Verify the TOE uses NTP protocol version 4 using packet capture.
Test Result	Pass

FCS_NTP_EXT.1.3

Test Number	Test 1
Test Objective	Verify the TOE will not accept broadcast or multicast NTP packets.
Test Steps Performed	1. Configure the NTP server to send broadcast/multicast NTP packets. 2. Verify the TOE does not support accepting broadcast or multicast NTP packets and does not accept them.
Test Result	Pass

FCS_NTP_EXT.1.4

Test Number	Test 1
Test Objective	Verify the TOE can be configured to synchronize with multiple NTP servers.
Test Steps Performed	1. Configure the TOE to use 3 NTP servers as time sources, each of them using different message authentication algorithms. 2. Verify the TOE was able to establish a connection to all 3 servers and used RFC-prescribed logic to choose the correct time source among them.
Test Result	Pass

Test Number	Test 2

Test Objective	Verify the TOE only accepts NTP updates from configured NTP servers.
Test Steps Performed	1. Force an NTP server to send a correct but unsolicited response to the TOE, originating not from the server the TOE was configured to connect. 2. Verify the TOE rejected this response.
Test Result	Pass

2.14 FCS_RBG_EXT.1 Random Bit Generation

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

TSS

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.

Guidance Documentation

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

Tests

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.

2.14.1 TSS

[ST] Section 7.2.4 specifies that the TOE uses AES256 CTR DRBG implemented as OpenSSL v1.1.1c-15-B1 module, seeded with 320-bit length output of hardware-based entropy source (Intel RDSEED). The [ST] claims that key generation uses at least as many random bits are used as the desired key length to ensure sufficient entropy has been provided. The minimum entropy assumed is 100%.

2.14.2 Guidance Documentation

[ST] Section 7.2.4 states that RNG functionality is not configurable. The [AGD] does not contain instructions for the configuration of RNG.

2.14.3 Tests

[ST] Section 6.1.2.9 claims “CTR DRBG (AES)”. This information is consistent with the information listed in CAVP Certificate #C1935. This test is satisfied by CAVP certification.

2.15 FCS_SSHC_EXT.1 SSH Client (Selection-Based)

TSS

FCS_SSHC_EXT.1.2

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

FCS_SSHC_EXT.1.3

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

FCS_SSHC_EXT.1.4

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

FCS_SSHC_EXT.1.5

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

If x509v3-based public key authentication algorithms are claimed, the evaluator shall confirm that the TSS includes the description of how the TOE establishes the server’s identity and how this identity is

confirmed with the one that is presented in the provided certificate. For example, the TOE could verify that a server's configured IP address matches the one presented in the server's x.509v3 certificate.

FCS_SSHC_EXT.1.6

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

FCS_SSHC_EXT.1.7

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

FCS_SSHC_EXT.1.8

The evaluator shall check that the TSS specifies the following:

a) Both thresholds are checked by the TOE.

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

Guidance Documentation

FCS_SSHC_EXT.1.4

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

FCS_SSHC_EXT.1.5

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

FCS_SSHC_EXT.1.6

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

FCS_SSHC_EXT.1.7

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

FCS_SSHC_EXT.1.8

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

Tests

FCS_SSHC_EXT.1.2

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

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

FCS_SSHC_EXT.1.3

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

FCS_SSHC_EXT.1.4

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

FCS_SSHC_EXT.1.5

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

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

FCS_SSHC_EXT.1.6

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

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

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

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

FCS_SSHC_EXT.1.7

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

FCS_SSHC_EXT.1.8

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

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

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

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

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

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

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

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

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

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

FCS_SSHC_EXT.1.9

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

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

2.15.1 TSS

FCS_SSHC_EXT.1.2, FCS_SSHC_EXT.1.5
[ST] Section 7.2.5 describes that the TOE uses only SSH-RSA public key authentication algorithm and password-based authentication. This corresponds to selections made in [ST] Section 6.1.2.10. The TOE does not use x509v3-based public keys.

FCS_SSHC_EXT.1.3
[ST] Section 7.2.5 describes that a 'Large packet" is a packet with a length greater than 262144 and that a session containing such a packet will be terminated when this packet is discovered.

FCS_SSHC_EXT.1.4, FCS_SSHC_EXT.1.6, FCS_SSHC_EXT.1.7
[ST] Section 7.2.5 specifies encryption algorithms are AES128-CBC and AES256-CBC and integrity algorithms are HMAC-SHA2-256 and HMAC-SHA2-512, and DH-Group14-SHA1 as the key exchange method used by the TOE. This description is consistent with selections made in [ST] Section 6.1.2.10.

FCS_SSHC_EXT.1.8
[ST] Section 7.2.5 describes that ReKeyLimit parameter is configured to observe both the required thresholds. Rekeying is performed upon reaching the threshold that is hit first.

2.15.2 Guidance Documentation

[AGD] Section 10 contains a description of the default configuration of the TOE so that SSH Client conforms to the description in Section 7.2.5 of the [ST]: specifically identifies configuration for host key algorithm of ssh-rsa, encryption algorithm (aes128-cbc,aes256-cbc), Key Exchange Algorithm (only diffie-hellman-group14-sha1) and MAC algorithm (only sha2-256,hmac-sha2-512). No option for MAC is not allowed. This also contains settings for the rekey threshold being 1 GB and 1 hour.

[AGD] Section 10 contains instructions that this pre-configured setting must not be changed by the Security Administrator.

2.15.3 Tests

FCS_SSHC_EXT.1.2

Test Number	Test 1
Test Objective	Demonstrate the SSH Client TSF can successfully authenticate to a remote audit server (SSH server) using a password-based credential.
Test Steps Performed	1. Configure the TOE SSH Client to use password-based authentication as per [AGD].

	2. Start the TOE SSH Client and verify the connection is successful.
Test Result	Pass

FCS_SSHC_EXT.1.3

Test Number	Test 1
Test Objective	Ensure the TOE behaves as expected when receiving packets larger than 262144 bytes.
Test Steps Performed	1. Configure the SSH Server to send a large packet shortly after a connection is established. 2. Establish a connection to the SSH Server using the TOE SSH Client. 3. Observe that the SSH Server sends a large packet and the TOE ignores that packet.
Test Result	Pass

FCS_SSHC_EXT.1.4

Test Number	Test 1
Test Objective	Verify the TOE only uses encryption algorithms selected in the [ST] (aes128-cbc or aes256-cbc).
Test Steps Performed	1. Use the TOE SSH Client to establish a connection to the remote server. 2. Verify the TOE only offers aes128-cbc or aes256-cbc as supported ciphers for connection encryption.
Test Result	Pass

FCS_SSHC_EXT.1.5

Test Number	Test 1
Test Objective	Verify the TOE supports all hostkey algorithms selected in the [ST] (ssh-rsa).
Test Steps Performed	1. Configure the SSH Server to present the ssh-rsa -based hostkey. 2. Connect to this server using the TOE SSH Client. 3. Verify the connection is successful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE rejects SSH servers that present a public key that does not match the public key algorithm(s) supported by the TOE.
Test Steps Performed	1. Configure the SSH Server to present the ssh-dss -based hostkey. 2. Connect to this server using the TOE SSH Client. 3. Verify the connection is unsuccessful.
Test Result	Pass

FCS_SSHC_EXT.1.6

Test Number	Test 1
Test Objective	Verify the MAC algorithms selected in the [ST] (hmac-sha2-256, hmac-sha2-512) are supported by the TOE.
Test Steps Performed	1. Configure the SSH Server to use hmac-sha2-256, hmac-sha2-512 as message authentication algorithms.

	2. Connect to this server using the TOE SSH Client. 3. Verify the connection is successful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE rejects MAC algorithms not selected in the [ST].
Test Steps Performed	1. Configure the SSH Server to use hmac-sha1 as the message authentication algorithms. 2. Connect to this server using the TOE SSH Client. 3. Verify the connection is unsuccessful.
Test Result	Pass

FCS_SSHC_EXT.1.7

Test Number	Test 1
Test Objective	Verify the key exchange algorithms selected in the [ST] (diffie-hellman-group14-sha1) are supported by the TOE.
Test Steps Performed	1. Configure the SSH Server to use diffie-hellman-group14-sha1 as the message key exchange algorithm. 2. Connect to this server using the TOE SSH Client. 3. Verify the connection is successful.
Test Result	Pass

FCS_SSHC_EXT.1.8

Test Number	Test 1
Test Objective	Verify the rekey is performed as described in the [ST].
Test Steps Performed	1. Configure the TOE to rekey at 1G and 180 seconds. 2. Establish the connection to the SSH Server. 3. Observe the rekey is happening before 180 seconds have ellapsed. 4. Configure the TOE to rekey at 10 MB and 1 hour. 5. Observe the rekey is happening shortly after the 10 MB limit is reached.
Test Result	Pass

FCS_SSHC_EXT.1.9

Test Number	Test 1
Test Objective	Verify the TOE does not automatically accept SSH servers that are not configured on the TOE as trusted.
Test Steps Performed	1. Remove the SSH server key from the trusted host key storage. 2. Establish a connection to the SSH Server. 3. Verify the connection is rejected
Test Result	Pass

2.16 FCS_SSHS_EXT.1 SSH Server (Selection-Based)

TSS

FCS_SSHS_EXT.1.2

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

FCS_SSHS_EXT.1.3

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

FCS_SSHS_EXT.1.4

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

FCS_SSHS_EXT.1.5

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

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

FCS_SSHS_EXT.1.6

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

FCS_SSHS_EXT.1.7

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

FCS_SSHS_EXT.1.8

The evaluator shall check that the TSS specifies the following:

a) Both thresholds are checked by the TOE.

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

Guidance Documentation

FCS_SSHS_EXT.1.4

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

FCS_SSHS_EXT.1.5

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

FCS_SSHS_EXT.1.6

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

FCS_SSHS_EXT.1.7

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

FCS_SSHS_EXT.1.8

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

Tests

FCS_SSHS_EXT.1.2

Test 1: If password-based authentication methods have been selected in the ST then using the guidance documentation, the evaluator shall configure the TOE to accept password-based authentication and demonstrate that user authentication succeeds when the correct password is provided by the user.

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

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

FCS_SSHS_EXT.1.3

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

FCS_SSHS_EXT.1.4

The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish an SSH connection. To verify this, the evaluator shall start session establishment for an SSH connection from a remote client (referred to as ‘remote endpoint’ below). The evaluator shall capture the traffic exchanged between the TOE and the remote endpoint during protocol negotiation (e.g. using a packet capture tool or information provided by the endpoint, respectively). The evaluator shall verify from the

captured traffic that the TOE offers all the ciphers defined in the TSS for the TOE for SSH sessions, but no additional ones compared to the definition in the TSS. The evaluator shall perform one successful negotiation of an SSH session to verify that the TOE behaves as expected. It is sufficient to observe the successful negotiation of the session to satisfy the intent of the test. If the evaluator detects that not all ciphers defined in the TSS for SSH are supported by the TOE and/or the TOE supports one or more additional ciphers not defined in the TSS for SSH, the test shall be regarded as failed.

FCS_SSHS_EXT.1.5

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

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

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

FCS_SSHS_EXT.1.6

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

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

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

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

FCS_SSHS_EXT.1.7

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.

FCS_SSHS_EXT.1.8

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

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

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

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

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

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

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

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

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

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

2.16.1 TSS

FCS_SSHS_EXT.1.2
[ST] Section 7.2.6 describes that the TOE uses only SSH-RSA, RSA-SHA2-256 and RSA-SHA2-512 public key authentication algorithm and password-based authentication for SSH connections.

FCS_SSHS_EXT.1.3
[ST] Section 7.2.6 describes that a ‘Large packet” is a packet with a length greater than 262144 and that a session containing such packet will be terminated when this packet is discovered.

FCS_SSHS_EXT.1.4, FCS_SSHS_EXT.1.6, FCS_SSHS_EXT.1.7
[ST] Section 7.2.6 specifies encryption algorithms are AES128-CBC and AES256-CBC and integrity algorithms are HMAC-SHA2-256 and HMAC-SHA2-512, and DH-Group14-SHA1 as the key exchange method used by the TOE. This description is consistent with selections made in [ST] Section 6.1.2.11.

FCS_SSHS_EXT.1.5
[ST] Section 7.2.6 describes that when a public key is presented for authentication, the user identity is established by comparing the presented public key with the stored public key in the “authorized_keys” file in the .ssh sub directory of the client directory on the server.

FCS_SSHS_EXT.1.8
[ST] Section 7.2.6 describes that ReKeyLimit parameter is configured to observe both the required thresholds. Rekeying is performed upon reaching the threshold that is hit first.

2.16.2 Guidance Documentation

[AGD] Section 12 contains a description of the default configuration of the TOE so that SSH Server that conforms to the description in Section 7.2.6 of the [ST]: specifically identifies configuration for public key algorithm of ssh-rsa, rsa-sha2-256, rsa-sha2-512, encryption algorithm (aes128-cbc,aes256-cbc), Key Exchange Algorithm (only diffie-hellman-group14-sha1) and MAC algorithm (only hmac-sha2-256,hmac-sha2-512). No option for MAC is not allowed. This also contains settings for the rekey threshold being 1 GB and 1 hour.

[AGD] Section 12 contains instructions that this pre-configured setting must not be changed by Administrator.

2.16.3 Tests

FCS_SSHS_EXT.1.2

Test Number	Test 1
Test Objective	Verify password-based authentication is supported by the TOE.
Test Steps Performed	1. Configure the TOE SSH server as per the [AGD] to support password authentication. 2. Connect to the TOE using the correct password. 3. Verify the connection is successful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify password-based authentication is supported by the TOE and the password is indeed verified by the TSF.
Test Steps Performed	1. Configure the TOE SSH server as per the [AGD] to support password authentication. 2. Connect to the TOE using an incorrect password. 3. Verify the connection is unsuccessful.
Test Result	Pass

FCS_SSHS_EXT.1.3

Test Number	Test 1
Test Objective	Ensure the TOE behaves as expected when receiving packets larger than 262144 bytes.
Test Steps Performed	1. Use SSH client to send a packet that is below the threshold. 2. Verify the packet is accepted. 3. Use the SSH Client to send a packet that is above the threshold. 4. Verify the connection is terminated upon receiving such packet.
Test Result	Pass

FCS_SSHS_EXT.1.4

Test Number	Test 1
Test Objective	Verify the TOE only uses encryption algorithms selected and described in the [ST] (aes128-cbc and aes256-cbc).
Test Steps Performed	1. Connect to the TOE using verbose output and packet capture. 2. Verify the TOE proposed only aes128-cbc and aes256-cbc and encryption algorithms.
Test Result	Pass

FCS_SSHS_EXT.1.5

Test Number	Test 1
Test Objective	Verify the TOE supports the following host key algorithms selected in the [ST]: ssh-rsa, rsa-sha2-256 and rsa-sha2-512.
Test Steps Performed	1. Establish an SSH session to the TOE with rsa-sha-512 host key algorithm. Verify the session is successfully established. 2. Establish an SSH session to the TOE with enforced rsa-sha-256 host key algorithm. Verify the session is successfully established. 3. Establish an SSH session to the TOE with enforced ssh-rsa host key algorithm. Verify the session is successfully established.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the server rejects authentication attempts of clients that present a public key that does not match public key(s) associated by the TOE with the identity of the client (i.e. the peer public keys are unknown to the TOE).
Test Steps Performed	1. Create a public key that is not added to the TOE trusted keys store. 2. Attempt a connection with this key. 3. Verify a connection is not established.
Test Result	Pass

Test Number	Test 3
Test Objective	Verify the TOE will reject public keys with algorithms other then those selected in the [ST] (i.e. other than ssh-rsa, rsa-sha2-256 and rsa-sha2-512).
Test Steps Performed	1. Create an ECDSA public key that is added to the TOE trusted keys store. 2. Attempt a connection with this key. 3. Verify a connection is not established.
Test Result	Pass

FCS_SSHS_EXT.1.6

Test Number	Test 1
Test Objective	Verify the MAC algorithms selected in the [ST] (hmac-sha2-256, hmac-sha2-512) are supported by the TOE.
Test Steps Performed	1. Establish an SSH connection to the TOE with an enforced hmac-sha2-256 MAC algorithm.

	2. Verify the connection is successful. 3. Establish an SSH connection to the TOE with enforced hmac-sha2-512 MAC algorithm. 4. Verify the connection is successful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE rejects MAC algorithms not selected in the [ST].
Test Steps Performed	1. Establish an SSH connection to the TOE with an enforced hmac-sha1 MAC algorithm. 2. Verify the connection is unsuccessful.
Test Result	Pass

FCS_SSHS_EXT.1.7

Test Number	Test 1
Test Objective	Verify the TOE does not allow diffie-hellman-group1-sha1 key exchange algorithms.
Test Steps Performed	1. Establish an SSH connection to the TOE with a diffie-hellman-group1-sha1 key exchange algorithm. 2. Verify the connection is unsuccessful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE does not allow diffie-hellman-group1-sha1 key exchange algorithms.
Test Steps Performed	1. Establish an SSH connection to the TOE with a diffie-hellman-group1-sha1 key exchange algorithm. 2. Verify the connection is unsuccessful.
Test Result	Pass

FCS_SSHS_EXT.1.8

Test Number	Test 1
Test Objective	Verify a rekey is performed as described in the [ST].
Test Steps Performed	1. Configure the TOE to perform a rekey after 1 minute. 2. Establish an SSH connection. 3. Observe a rekey is performed before 60 seconds has elapsed. 4. Configure the TOE to perform a rekey after 15 MB of data. 5. Establish an SSH connection and transfer 15 MB of data. 6. Observe a rekey is happening after 15 MB of data is transferred.
Test Result	Pass

2.17 FCS_TLSS_EXT.1 Extended: TLS Server Protocol without Mutual Authentication (Selection-Based)[5]

TSS

FCS_TLSS_EXT.1.1

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

FCS_TLSS_EXT.1.2

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

FCS_TLSS_EXT.1.3

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

FCS_TLSS_EXT.1.4

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

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

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

Guidance Documentation

FCS_TLSS_EXT.1.1

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

FCS_TLSS_EXT.1.2

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

FCS_TLSS_EXT.1.3

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

Tests

[5] FCS_TLSS_EXT.1.4 Test 3 was modified by TD0556.

FCS_TLSS_EXT.1.1

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

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

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

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

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

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

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

FCS_TLSS_EXT.1.2

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

FCS_TLSS_EXT.1.3

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

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

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

Test 2: [conditional] If DHE ciphersuites are supported, the evaluator shall repeat the following test for each supported parameter size. If any configuration is necessary, the evaluator shall configure the TOE to use a supported Diffie-Hellman parameter size. The evaluator shall attempt a connection using a supported DHE ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a Server Key Exchange Message where p Length is consistent with the message are the ones configured Diffie-Hellman parameter size(s).

Test 3: [conditional] If RSA key establishment ciphersuites are supported, the evaluator shall repeat this test for each RSA key establishment key size. If any configuration is necessary, the evaluator shall configure the TOE to perform RSA key establishment using a supported key size (e.g. by loading a certificate with the appropriate key size). The evaluator shall attempt a connection using a supported RSA key establishment ciphersuite. The evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a certificate whose modulus is consistent with the configured RSA key size.

FCS_TLSS_EXT.1.4

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

Test 1 [conditional]: If the TOE does not support session resumption based on session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2) or session tickets according to RFC5077, the evaluator shall perform the following test:

a) The client sends a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket.

b) The client verifies the server does not send a NewSessionTicket handshake message (at any point in the handshake).

c) The client verifies the Server Hello message contains a zero-length session identifier or passes the following steps:
Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

d) The client completes the TLS handshake and captures the SessionID from the ServerHello.

e) The client sends a ClientHello containing the SessionID captured in step d). This can be done by keeping the TLS session in step d) open or start a new TLS session using the SessionID captured in step d).

f) The client verifies the TOE (1) implicitly rejects the SessionID by sending a ServerHello containing a different SessionID and by performing a full handshake (as shown in Figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC4346 (TLS1.1) or RFC5246 (TLS1.2), the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall conduct a successful handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then initiate a new TLS connection and send the previously captured session ID to show that the TOE resumed the previous session by responding with ServerHello containing the same SessionID immediately followed by ChangeCipherSpec and Finished messages (as shown in Figure 2 of RFC 4346 or RFC 5246).

b) The evaluator shall initiate a handshake and capture the TOE-generated session ID in the Server Hello message. The evaluator shall then, within the same handshake, generate or force an unencrypted fatal Alert message immediately before the client would otherwise send its ChangeCipherSpec message thereby disrupting the handshake. The evaluator shall then initiate a new Client Hello using the previously captured session ID, and verify that the server (1) implicitly rejects the session ID by sending a ServerHello containing a different SessionID and performing a full handshake (as shown in figure 1 of RFC 4346 or RFC 5246), or (2) terminates the connection in some way that prevents the flow of application data.

Test 3 [conditional]: If the TOE supports session tickets according to RFC5077, the evaluator shall carry out the following steps (note that for each of these tests, it is not necessary to perform the test case for each supported version of TLS):

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with an abbreviated handshake described in section 3.1 of RFC 5077 and illustrated with an example in figure 2. Of particular note: if the server successfully verifies the client's ticket, then it may renew the ticket by including a NewSessionTicket handshake message after the ServerHello in the abbreviated handshake (which is shown in figure 2). This is not required, however as further clarified in section 3.3 of RFC 5077.

b) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator will then modify the session ticket and send it as part of a new Client Hello message. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake (as shown in figure 3 or 4 of RFC 5077), or (2) terminates the connection in some way that prevents the flow of application data.

2.17.1 TSS

FCS_TLSS_EXT.1.1
[ST] Section 7.2.7 specifies TLS_RSA_with_AES_128_CBC_SHA, defined in RFC 3268 and TLS_ECDHE_RSA_with_AES_256_GCM_SHA384, defined in RFC 5289 as ciphersuites supported by the TOE. Those are consistent with the selection in [ST] Section 6.1.2.12.

FCS_TLSS_EXT.1.2
[ST] Section 7.2.7 describes that the TOE will reject the connection when connections using TLS version 1.1 or older are requested from the TOE. It describes, that the TOE rejects the connection attempt using unsupported TLS or SSL protocol versions by sending “handshake failure” and “invalid protocol version” error responses to the client.

FCS_TLSS_EXT.1.3
[ST] Section 7.2.7 describes that the TOE will use NIST curve secp256r1 for ECDHE Key agreement.

FCS_TLSS_EXT.1.4
[ST] Section 7.2.7 describes that the TOE does not support session resumption or session tickets.

2.17.2 Guidance Documentation

[AGD] Section 14 contains a description of the TLS Server configuration through /opt/TLS-server/stunnel.conf configuration file. This includes protocol version, ciphersuites and ECDHE curve parameters. Listed parameters conform to description in the TSS (Section 7.2.7 of the [ST]).

[AGD] prescribes that the Administrator should not change these pre-configured parameters.

2.17.3 Tests

FCS_TLSS_EXT.1.1

Test Number	Test 1
Test Objective	Verify the TOE accepts the ciphersuites selected in the [ST]
Test Steps Performed	1. Configure the TLS Server as per [AGD]. 2. Establish a connection to the TOE with TLS_RSA_with_AES_128_CBC_SHA ciphersuite. 3. Verify the connection is successful. 4. Establish a connection to the TOE with TLS_ECDHE_RSA_with_AES_256_GCM_SHA384 ciphersuite. 5. Verify the connection is successful.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE will not accept ciphersuites not selected in the [ST] and will not accept NULL encryption.
Test Steps Performed	1. Attempt to establish a connection with the TOE using ciphersuites not selected in the [ST]. 2. Verify the connection is rejected. 3. Attempt to establish a connection using TLS_NULL_WITH_NULL_NULL ciphersuite. 4. Verify the connection is rejected.
Test Result	Pass

Test Number	Test 3a
Test Objective	Verify the handshake will not be completed if Client Finished handshake message is invalid.
Test Steps Performed	1. Corrupt the Client Finished handshake message when connecting to the TOE. 2. Verify the TOE rejected the connection.
Test Result	Pass

Test Number	Test 3b
Test Objective	Ensure the server's TLS implementation immediately makes use of the key exchange and authentication algorithms to: a) Correctly encrypt the TLS Finished message and b) Encrypt every TLS message after session keys are negotiated.
Test Steps Performed	1. Connect to the TOE using TLS. 2. Inspect the TLS Finished message and Change Cipher Spec using the packet capture message to verify the message is encrypted

	immediately.
Test Result	Pass

FCS_TLSS_EXT.1.2

Test Number	Test 1
Test Objective	Verify the TOE rejects protocol versions as selected in the [ST].
Test Steps Performed	1. Attempt to establish a connection to the TOE using SSLv1, SSLv2, SSLv3, TLSv1.0 and TLSv1.1. 2. Verify the connection was rejected for all those protocol versions.
Test Result	Pass

FCS_TLSS_EXT.1.3

Test Number	Test 1
Test Objective	Verify the TOE properly negotiates the ECDHE curves that are claimed in the [ST] (secp256r1) and no others.
Test Steps Performed	1. Establish a connection to the TOE with TLS_ECDHE_RSA_with_AES_256_GCM_SHA384 ciphersuite. 2. Verify the connection using secp256r1 curve is successful. 3. Establish a connection to the TOE with TLS_ECDHE_RSA_with_AES_256_GCM_SHA384 ciphersuite and secp256k1 and secp384r1 curves. 4. Verify the connection is rejected.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE can properly negotiate a DHE key exchange algorithm.
Test Steps Performed	The [ST] does not contain a DHE key establishment selection.
Test Result	Pass

Test Number	Test 3
Test Objective	Verify the TOE uses correct key sizes for RSA key exchange.
Test Steps Performed	1. Establish a connection to the TOE with TLS_RSA_with_AES_128_CBC_SHA ciphersuite. 2. Verify the RSA key size is 2048 bits.
Test Result	Pass

FCS_TLSS_EXT.1.4

Test Number	Test 1
Test Objective	Demonstrate the TOE will not resume a session for which the client failed to complete the handshake (independent of the TOE support for session resumption).
Test Steps Performed	1. Establish and terminate a connections to the TOE. 2. Attempt to re-establish the connection using session resumption. 3. Verify the session will not be resumed and new sessions will be negotiated.
Test Result	Pass

Test Number	Test 2
Test Objective	Demonstrate the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).
Test Steps Performed	The TOE does not support session resumption as per [ST].
Test Result	Pass

Test Number	Test 3
Test Objective	Demonstrate the TOE will not resume a session for which the client failed to complete the handshake (independent of TOE support for session resumption).
Test Steps Performed	The TOE does not support session tickets.
Test Result	Pass

2.18 FIA_AFL.1 Authentication Failure Management

TSS

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).

Guidance Documentation

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.

Tests

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

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

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

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

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

2.18.1 TSS

[ST] Section 7.3.1 states that each remote administrator successive unsuccessful attempt shall be audited and logged. Remote Administrators are prevented from logging into the TOE when the configurable limit of attempts is reached. This block will expire after a configurable period of time or it can be removed by administrator when authenticated locally.

[ST] Section 7.3.1 describes that authentication from the local console for an "admin" administrator account will always be allowed.

[ST] Section 7.3.3 states that remote administrative sessions are established via TLS using username and password, and via SSH using either public-key authentication or using username and password. [ST] Section 7.3.1. describes that the password lockout policy does not apply to authentication attempts performed using public key-based authentication.

2.18.2 Guidance Documentation

[AGD] Section 7 contains a description for configuring the number of successive unsuccessful attempts for password authentication via SSH and TLS trusted path (3 by default) and lockout time period (default is 600 seconds) through configuration file /etc/pam.d/password-auth.

[AGD] Section 7 describes the action that needs to be performed to unlock the locked account.

[AGD] Section 7 describes that the Administrator account will never be prevented from logging over the local console.

2.18.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE observes failed authentication configuration as per guidance.
Test Steps Performed	1. Configure the TOE to lock out the user after 3 unsuccessful attempts and enforce lockout period of 5 minutes. For TLS access method: 2. Exceed unsuccessful login attempts to the TOE. 3. Verify the authentication is rejected until the lockout time is exceeded. For SSH access method: 4. Exceed unsuccessful login attempts to the TOE. 5. Verify the authentication is rejected until the lockout time is exceeded.
Test Result	Pass

Test Number	Test 2

Test Objective	Verify a locked account can be unlocked using an administrator action.
Test Steps Performed	For TLS Access method: 1. Attempt to login to the TOE using telnet over TLS and input incorrect credentials 3 times triggering account lockout, then attempt to login using the correct credentials. 2. Verify the authentication fails. 3. Unlock the account using the local console. 4. Verify authentication is now successful. For SSH Access method: 1. Attempt to login to the TOE using telnet over SSH and input incorrect credentials 3 times triggering account lockout, then attempt to login using the correct credentials. 2. Verify the authentication fails. 3. Unlock the account using the local console. 4. Verify authentication is now successful.
Test Objective	Verify the access lockout timer is functioning correctly.
Test Steps Performed	1. Attempt to login to the TOE using telnet over TLS and input incorrect credentials 3 times triggering account lockout, then attempt to login using the correct credentials. 2. Verify that authentication fails. 3. Wait until lockout time is exceeded. 4. Verify authentication is now successful.
Test Result	Pass

2.19 FIA_PMG_EXT.1 Password Management

TSS

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

Guidance Documentation

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.

Tests

The evaluator shall perform the following tests.

a) Test 1: The evaluator shall compose passwords that meet the requirements in some way. For each password, the evaluator shall verify that the TOE supports the password. While the evaluator is not required (nor is it feasible) to test all possible compositions of passwords, the

evaluator shall ensure that all characters, and a minimum length listed in the requirement are supported and justify the subset of those characters chosen for testing.

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

2.19.1 TSS

[ST] Section 7.3.2 describes that administrator passwords may be configured to be between 9 and 128 characters.

[ST] Section 7.3.2 describes that administrator passwords may be composed on all printable ASCII characters in UTF-8 formatting.

2.19.2 Guidance Documentation

[AGD] Section 11 describes that, “Any printable character is a valid character for use in a password.”

[AGD] Section 11 subsection “Strong password” contains a description of what is considered a strong password.

[AGD] Section 11 describes that minimal password length should be configured by changing “minlen” parameter in /etc/security/pwquality.conf configuration file.

[AGD] Section 11 contains description that minimal length should be configurable between 9 and 128 characters.

2.19.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE supports passwords consisting of characters stated in the [ST] (all printable characters).
Test Steps Performed	1. Try to set up a password using printable characters. 2. Verify the operation succeeded. 3. Try to set up a password with 72 characters. 4. Verify the operation succeeded.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE enforces the allowed characters and the minimum length for the password.
Test Steps Performed	1. Try to set up a password shorter than the defined minimum length. 2. Verify the operation is unsuccessful. 3. Try to set up password with non-printable characters. 4. Verify the operation is unsuccessful. 5. Try to set up a different password with non-printable characters. 6. Verify the operation is unsuccessful.
Test Result	Pass

2.20 FIA_UIA_EXT.1 User Identification and Authentication

TSS

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.

Guidance Documentation

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.

Tests

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.

2.20.1 TSS

[ST] Section 7.3.3 describes that for SSH remote sessions, administrators use an SSH client to establish a secure channel with the TOE. They authenticate using their username and password or public key authentication. Successful logon is denoted by obtaining a command prompt, and unsuccessful login will result in a terminated connection.

[ST] Section 7.3.3 describes that for TLS remote sessions, administrators use a TLS client to negotiate a secure connection with the TOE. Administrators will authenticate using their username and password. Successful logon is denoted by obtaining a command prompt, and unsuccessful login will result in a terminated connection.

[ST] Section 7.3.3 describes that for local authentication using local console, administrators use a username and password for authentication. Successful authentication is denoted by obtaining a command prompt, while unsuccessful authentication results in a prompt to reauthenticate.

[ST] Section 7.3.3 describes that certain actions are allowed before the TOE requires an external entity to authenticate. The list of these actions in [ST] Section 7.3.3 is consistent with selections in Section 6.1.3.3. No administrative action is allowed prior to authentication.

2.20.2 Guidance Documentation

The following are the available means and methods of logging into the TOE with the appropriate [AGD] section listed:

1. Local Console:
    a. Username/Password:
        i. [AGD] Section 4, subsection “Initial Configuration”, provides the guidance on logging into the TOE via the local console with the default administrative credentials.
        ii. [AGD] Section 4, subsection “Initial Configuration” contains guidance on changing the administrative user password.
2. Remote Sessions:
    a. SSH Sessions:
        i. Username/Password:
            1. [AGD] Section 11, subsection “Changing the Administrator’s password,” contains guidance on changing the administrative user’s password.
            2. [AGD] Section 4, subsection “Initial Configuration” contains guidance that this method is enabled by default and no configuration is required.
            3. [AGD] Section 11 contains instructions on how to login using this method.
        ii. Public-Key-based authentication:
            1. [AGD] Section 11, subsection “Public key based authentication” contains instruction on configuring the TOE to accept this authentication method.
            2. [AGD] Section 11, subsection “Public key based authentication” contains instruction on creating and delivering required public key.
            3. [AGD] Section 11 contains instructions on how to login using this method.
    b. TLS Sessions:
        i. Username/Password:

1. [AGD] Section 11, subsection “Changing the Administrator’s password,” contains guidance on changing the administrative user’s password.
2. [AGD] Section 14, subsection “Note on first login via the TLS server” contains instructions on how to login using this method.

No configuration is necessary to ensure the services provided by the TOE prior to authentication are limited.

2.20.3 Tests

Test Number	Test 1
Test Objective	Objective 1: Verify the TSF supports password-based authentication of the local console. Objective 2: Verify the TSF denies a password-based authentication attempt of the local console using an incorrect password. Objective 3: Verify the TSF supports password-based authentication via SSH. Objective 4: Verify the TSF denies a password-based authentication attempt via SSH using an incorrect password. Objective 5: Verify the TSF supports public-key based authentication via SSH. Objective 6: Verify the TSF denies a public-key authentication attempt via SSH using an incorrect public-key. Objective 7: Verify the TSF supports a password-based authentication attempt via Telnet-over-TLS using the correct credentials. Objective 8: Verify the TSF denies a password-based authentication attempt via Telnet-over-TLS using the incorrect credentials.
Test Steps Performed	**For the Telnet-over-TLS authentication method**: 1. Access the TOE using Telnet through a TLS tunnel using incorrect credentials and correct credentials in succession. 2. Verify access is denied then granted. **For SSH password-based authentication method**: 1. Access the TOE using a SSH client using incorrect credentials and correct credentials in succession. 2. Verify access is denied then granted. **For Serial Console:** 1. Use the serial console to authenticate to the TOE using first incorrect credentials, then correct credentials. 2. Verify access is denied then granted. Objective 5 testing was satisfied by the testing performed for FCS_SSHS_EXT.1.5 SSH Server – TEST 1. Objective 6 testing was satisfied by the testing performed for FCS_SSHS_EXT.1.5 SSH Server – TEST 2.
Test Result	Pass

Test Number	Test 2
Test Objective	Confirm the list of services available to external entities prior to authentication is limited to the following: Objective 1: Display the warning banner in accordance with FTA_TAB.1. Objective 2: Respond to ICMP Echo Request. Objective 3: Respond to ARP requests with ARP replies. Objective 4: Establish TLS connection on TCP port 27777. Objective 5: Automated generation of cryptographic key.
Test Steps Performed	1. Ensure the TOE is in the certified configuration. 2. Use a network scanner to identify available network services provided by the TOE. 3. Verify that only TCP ports 27777 and 22 are available. 4. Initiate ICMP request from unauthenticated entity. 5. Inspect the network capture to verify that the TOE responds to ICMP Echo and ARP requests. Objectives 4 and 5 are satisfied by testing in FCS_TLSS_EXT.1.1 – Test 1. Objective 1 is satisfied by testing in FTA_TAB.1.
Test Result	Pass

Test Number	Test 3
Test Objective	Objective 1: Determine what services are available to the local administrator prior to login and verify that no actions other than the ones listed below are accessible: • Display the warning banner in accordance with FTA_TAB.1; • Respond to ICMP Echo Request, respond to ARP requests with ARP replies, establish TLS connection on TCP port 27777, automated generation of cryptographic keys.
Test Steps Performed	Actual Result: While performing FMT_MOF.1.1/Functions – Test 1 and FTA_TAB.1.1, the evaluator determined that the only service available to the administrator through the local console before authentication is the display of warning banner service. Which is consistent with the selections in the requirements.
Test Result	Pass

2.21 FIA_UAU_EXT.2 Password-based Authentication Mechanism

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

2.22 FIA_UAU.7 Protected Authentication Feedback

Guidance Documentation

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.

Tests

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.

2.22.1 TSS

None.

2.22.2 Guidance Documentation

The [AGD] does not describe any necessary preparatory steps for enabling protected authentication feedback. During functional testing, the evaluator determined that no such configuration is necessary.

2.22.3 Tests

Test Number	Test 1
Test Objective	Verify obscured feedback is provided when entering credentials.
Test Steps Performed	1. Authenticate to the TOE using the serial console. Observe how password feedback is provided. 2. Verify feedback is obscured\not provided.
Test Result	Pass

2.23 FIA_X509_EXT.1/Rev X.509 Certificate Validation (Selection-Based)

TSS

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.

Guidance Documentation

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.

Tests

The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step or when performing trusted updates (if FPT_TUD_EXT.2 is selected). It is not sufficient to verify the status of a X.509 certificate only when it is loaded onto the TOE. It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self- testing is selected). The evaluator shall perform the following tests for FIA_X509_EXT.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

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 certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

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) Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The evaluator shall establish a valid, trusted certificate chain consisting of an EC leaf certificate, an

EC Intermediate CA certificate not designated as a trust anchor, and an EC certificate designated as a trusted anchor, where the elliptic curve parameters are specified as a named curve. The evaluator shall confirm that the TOE validates the certificate chain. The evaluator shall replace the intermediate certificate in the certificate chain for Test 8 with a modified certificate, where the modified intermediate CA has a public key information field where the EC parameters uses an explicit format version of the Elliptic Curve parameters in the public key information field of the intermediate CA certificate from Test 8, and the modified Intermediate CA certificate is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

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).

2.23.1 TSS

[ST] Section 7.3.6 claims that x.509 certificate validation is performed by the TOE.

[ST] Section 7.3.6 states that x509 certificate validation occurs when certificates are imported or installed to the TOE or when server is restarted after being stopped for any reason. Certificate validation does not occur when TLS session is being established as authentication is not being performed at session establishment.

The [ST] does not describe any rules for extendedKeyUsage that are not supported by the TOE or implicitly satisfied.

[ST] Section 7.3.6 states that revocation check is being done as part of certificate validation. A validation check is performed on all certificate chains terminated with a trusted CA certificate.

The [ST] does not state that there is a difference in revocation checking for a leaf certificate and chain.

2.23.2 Guidance Documentation

[AGD] Section 14 describes that CA certificates are validated whenever the CA certificate is imported or when the server certificate is imported.

[AGD] Section 15 describes that the server certificate is validated when a server certificate is imported, and every time the TLS server is started. The server certificate is validated against a CA certificate and revocation checking is performed via CRL specified in a certificate CDP field.

[AGD] Section 14 describes that, if the server certificate contains a CDP, the CRL will be downloaded from the CDP when the server certificate is imported and when the server is started. If the host is accessible, but the file is not present, it is treated as an error and the import operation will fail. If the host is not accessible due to a network problem, any import operation will fail (if the certificate is being imported) and attempts to start the server will fail. If a CDP is not present in the certificate, the certificate will be considered valid.

The [ST] and [AGD] do not describe any rules for extendedKeyUsage fields that are not supported by the TOE.

2.23.3 Tests

FIA_X509.1.1/REV

Test Number	Test 1a, 1b.
Test Objective	Verify that the TOE will accept the certificate that has the valid trust chain. Verify the TOE will reject the certificate if the trust chain cannot be validated.
Test Steps Performed	1. Generate a chain of 2 CA certificates and a leaf certificate. 2. Import a certificate and the chain to the TOE. 3. Verify that import is successful. 4. Remove the intermediate CA from the chain. 5. Import the chain. 6. Verify the import is rejected.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TSF will reject the expired certificate.
Test Steps Performed	1. Generate a chain of 2 CA certificates and a leaf certificate with an expired date. 2. Initiate certificate import. 3. Verify the import fails.
Test Result	Pass

Test Number	Test 3
Test Objective	Verify the TOE correctly handles revocation through CRL.

Test Steps Performed	1. Install a correct trust chain and leaf certificate to the TOE. Run the certificate import procedure. 2. Create a CRL that indicates a leaf certificate is revoked. 3. Verify the leaf certificate validation fails. 4. Create a CRL that indicates a CA certificate is revoked. 5. Verify the CA certificate validation fails.
Test Result	Pass

Test Number	Test 4
Test Objective	Verify the CRL validation is done with regard for CRL issuer certificate purpose.
Test Steps Performed	1. Create a certificate chain with a CA without CRLSign certificate purpose. 2. Create a CRL with this CA. 3. Run the certificate validation and verify it fails.
Test Result	Pass

Test Number	Test 5
Test Objective	Verify the TOE requires a strict beginning of Certificate format.
Test Steps Performed	1. Modify byte 5 in stunnet.crt file on TOE. 2. Run the certificate check and verify it fails.
Test Result	Pass

Test Number	Test 6
Test Objective	Verify the TOE validates the certificate signature.
Test Steps Performed	1. Modify the signatureValue field of the correct certificate. 2. Run the certificate check and verify it fails.
Test Result	Pass

Test Number	Test 7
Test Objective	Verify the TOE will detect public key modification in the imported certificate.
Test Steps Performed	1. Modify the public key of the correct certificate used. 2. Run the certificate check and verify it fails.
Test Result	Pass

Test Number	Test 8
Test Objective	Verify the TOE rejects certificates with explicit curve parameters.
Test Steps Performed	[ST] does not claim support for EC certificates as indicated in FCS_COP.1/SigGen.
Test Result	Pass

FIA_X509.1.2/REV

Test Number	Test 1
Test Objective	Verify the CA certificate without basicConstraints will be rejected.
Test Steps Performed	1. Modify the certificate chain so Intermediate CA certificate would not

	contain a basicConstraints extension. 2. Run the certificate check and verify that it fails.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the CA certificate without Ca=True extension will be rejected.
Test Steps Performed	1. Modify the Intermediate CA certificate so it contains a CA=False extension. 2. Run the certificate check and verify it fails.
Test Result	Pass

2.24 FIA_X509_EXT.2 X.509 Certificate Authentication (Selection-Based)

TSS

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.

Guidance Documentation

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.

Tests

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.

2.24.1 TSS

[ST] Section 7.3.7 describes that administrators can install certificates that are used in negotiating TLS connections and contains reference to [AGD].

The evaluator verified that Section 15 of [AGD] indeed contains guidance for creating certificates, CSR, importing certificates and configuring the TOE to use them.

[ST] Section 7.3.7 states that if the connection to download CRL cannot be established during revocation check, the certificate will be considered invalid.

2.24.2 Guidance Documentation

[AGD] Section 14 describes that a network connection is required to perform Certificate validations using CRL. [AGD] Section 14 and 15 describe the configuration for using certificates. [AGD] Section 14 describes that, if connection cannot be established to the CRL location during validity check certificate, it is considered invalid.

2.24.3 Tests

Test Number	Test 1
Test Objective	Verify when certificate validation cannot be performed, the TOE will not accept the certificate.
Test Steps Performed	1. Install the correct chain and CRL file. 2. Initiate the certificate check and verify it is successful. 3. Remove CRL file from the CRL server. 4. Initiate the certificate check and verify it fails.
Test Result	Pass

2.25 FIA_X509_EXT.3 X.509 Certificate Requests (Selection-Based)

TSS

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.

Guidance Documentation

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

Tests

The evaluator shall perform the following tests:

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

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

2.25.1 TSS

[ST] Section 6.1.3.8 does not contain “device-specific information” as a selection.

2.25.2 Guidance Documentation

[ST] Section 6.1.3.7, FIA_X509_EXT.3.1 states the Public-key, Common Name, Organization, Organizational Unit, and Country fields/information are provided in the CSR generated by the TOE.

[AGD] Section 15 “Making a certificate request and Installing Files” contains guidance on generating certificate requests using BiviOS commands, including establishing the Common Name, Organization, Organizational Unit, and Country fields in the CSR.

2.25.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE is able to generate a CSR that conforms to RFC 2986 and contains a public key and a Common Name, Organization, Organizational Unit and Country.
Test Steps Performed	1. Generate a CSR per [AGD]. 2. Verify the CSR contains all the required data.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify a signed certificate will be imported to the TOE only when a trust chain is provided.
Test Steps Performed	1. Generate the trust chain with a leaf certificate. 2. Import the leaf certificate to the TOE without importing the trust chain. 3. Verify the import fails. 4. Import the leaf certificate with the trust chain. 5. Verify the import is successful.
Test Result	Pass

2.26 FMT_MOF.1/Functions Management of Security Functions Behaviour (Selection-Based)

TSS

For distributed TOEs see chapter 2.4.1.1.

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

Guidance Documentation

For distributed TOEs see chapter 2.4.1.2.

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

Tests

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

authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to modify the security related parameters can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

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

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

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

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

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

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

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

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

Test 3 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen

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

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

2.26.1 TSS

The TOE is not a distributed TOE.

[ST] Section 6.1.4.1, FMT_MOF.1/Functions, states that the Security Administrator is able to determine the behaviour of the function for the transmission of audit data to an external IT entity.

[ST] Section 6.1.4.1 FMT_MOF.1/Functions states that the Security Administrator is able to modify the behaviour of the function for transmission of audit data to an external IT entity.

[ST] Section 7.4.1 describes that the Security Administrator modifies the behavior of the TOE functions for transmission of audit data to an external IT entity by configuring ssh-tunnel-client (specifying the destination server for this transmission).

[ST] Section 7.4.1 describes that the Security Administrator may determine the behaviour of the TOE functions for transmission of audit data to an external IT entity by viewing the configuration file.

2.26.2 Guidance Documentation

[ST] Section 6.1.4.1 states that the TOE is able to modify and determine the behaviour of the transmission of audit data to an external IT entity. [AGD] Section 10 describes how this configuration is done and describes configuration file */opt/ssh-tunnel-client/ssh-tunnel-client.conf* that is used to determine and modify this behaviour.

2.26.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE will not accept changes to the TSF configuration without authentication.
Test Steps Performed	1. Open an administrative session through the local console. 2. Try to issue commands to the TOE before authentication. Verify it fails. 3. Open a remote administrative TLS session. 4. Try to issue commands to the TOE before authentication. Verify it fails. 5. Open a remote administrative SSH session. 6. Try to issue commands to the TOE before authentication. Verify it fails.
Test Result	Pass

Test Number	Test 2

Test Objective	Verify Security Administrators can modify all parameters for the behaviour of transmission of audit data to an external IT entity.
Test Steps Performed	1. Discover the parameters for transmission of audit data. 2. Modify each of those parameters. 3. Restart transmission of the audit data. 4. Verify transmission is performed with modified settings.
Test Result	Pass

Test Number	Test 3
Test Objective	Verify it is impossible to determine the behaviour of security functions without prior authentication as a security administrator
Test Steps Performed	This test is satisfied by Test 1.
Test Result	Pass

Test Number	Test 4
Test Objective	Verify Security Administrators can determine the behaviour of transmission of audit data to an external IT entity.
Test Steps Performed	1. After authentication, as a Security Administrator, attempt to read trusted channel settings by reading configuration files. 2. Verify this is successful.
Test Result	Pass

2.27 FMT_MOF.1/ManualUpdate Management of Security Functions Behaviour

TSS

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

Guidance Documentation

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).

Tests

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

The evaluator 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.

2.27.1 TSS

None.

2.27.2 Guidance Documentation

[AGD] Section 19 contains descriptions of the necessary steps for performing a manual TOE update, including downloading, verifying integrity of update using published hash, and applying the update. It also describes that the update process is followed by the reboot procedure, resulting in system unavailability for a certain time.

2.27.3 Tests

Test Number	Test 1
Test Objective	Verify it is not possible to initiate an update without prior authentication as a Security Administrator. Verify it is possible to initiate an update with prior authentication as a Security Administrator.
Test Steps Performed	This test is covered by FMT_MOF.1/Functions and FPT_TUD.1.
Test Result	Pass

2.28 FMT_MOF.1/Services Management of Security Functions Behaviour (Selection-Based)

TSS

For distributed TOEs see chapter 2.4.1.1.

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

Guidance Documentation

For distributed TOEs see chapter 2.4.1.2.

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

Tests

The evaluator shall try to enable and disable at least one of the services as defined in the Application Notes for FAU_GEN.1.1 (whichever is supported by the TOE) without prior authentication as Security Administrator (either by authenticating as a user with no administrator privileges, if possible, or without prior authentication at all). The attempt to enable/disable this service/these services should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to enable/disable this service/these services can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

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

2.28.1 TSS

The TOE is not a distributed TOE.

[ST] Section 7.4.1 describes that the Security administrator may start and stop any services offered by the TSF, such as the TLS service, SSH service, NTP service, or syslog service. This is performed using Linux the systemd service mechanism or by using specific Bivio commands for TLS Login service and SSH tunnel service.

2.28.2 Guidance Documentation

[AGD] Section 21 contains a list of services that the Security Administrator is able to start and stop and description how this is done.

2.28.3 Tests

Test Number	Test 1
Test Objective	Objective 1 - Verify, without any user authentication, the user might not be able to get to the point where an attempt to enable/disable this service/these services can be executed. Objective 2 - Verify attempts to enable/disable the TLS service, the SSH service, the syslog service is successful when authenticated as a Security Administrator
Test Steps Performed	Objective 1 is satisfied by FMT_MOF.1.1/Functions -Test 1. 1. Authenticate to the TOE as a Security Administrator. 2. Attempt to disable and enable services as listed in the [ST]. 3. Verify it is successful.
Test Result	Pass

2.29 FMT_MTD.1/CoreData Management of TSF Data

TSS

The evaluator shall examine the TSS to determine that, for each administrative function identified in the guidance documentation; those that areaccessiblethroughaninterfacepriorto administratorlog-inareidentified. 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.

Guidance Documentation

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.

Tests

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

2.29.1 TSS

[AGD] identifies following management functions for the TOE:

- Setting up time;
- Enabling NTP time synchronization;
- Configuring login retry threshold and account locking duration;
- Unlocking locked accounts;
- Configuring password length settings;
- Reading audit data;
- Configuring warning banner;
- Starting and stopping TOE services;
- Configuring SSH Client for remote audit server:
    - Generating keys;
    - Zeroizing keys;
    - Setting up SSH configuration;
- Setting login access warning banner.
- Configuring SSH Server:
    - Generating keys;
    - Zeroizing keys;
    - Setting up SSH configuration.
- Configuring TLS server:
    - Generating certificates;
    - Generating Keys and zeroizing keys;
    - Creating CRLs;
    - Importing Server and CA certificate;
    - Cryptographic configuration.
- Software integrity checking.
- Running Cryptographic health tests.
- Updating the TOE, verifying hash of update candidate.
- Querying TOE version.

[ST] Section 7.4.1 states that it is not possible to view or modify any TOE settings and behavior prior to successful authentication. This includes the TOE trust store.

[ST] Section 7.4.2 states that any TFS data can be viewed only by authorized and authenticated administrators.

[ST] Section 7.3.3 describes that only certain non-security related functions are available prior to successful authentication for non-TOE entities: displaying access banner, responding to ICMP, ARP requests, establish TLS connection on TCP port 27777, automated generation of cryptographic keys for the purpose of establishing the connections. The evaluator concluded that no non-administrative users will be able to manipulate TSF data through these services.

2.29.2 Guidance Documentation

For each configuration section throughout the [AGD] documentation, [AGD] identifies the relevant files and parameters that contain the configuration of the TSF and states that the Evaluation Activities only cover those specifically identified files and parameters.

The TSF-data-manipulating functions are defined as the commands and actions an administrator can issue/take to configure the TOE into the CC evaluated configuration. These commands and actions are identified in guidance provided in [AGD].

The [AGD] makes it clear that every manipulation with those configuration files and parameters require not only successful system authentication, but also a privilege elevation using sudo command.

[ST] Section 7.4.3 and 6.1.4.6 state that the TOE is able to manage the TOE trust store and import X.509v3 certificates to the TOE trust store and designate X.509v3 certificates as trust anchors.

[AGD] Section 15 states that the TOE maintains a trusted CA certificate list in the /opt/TLS-server/ca.crt file.

[AGD] Section 16 describes procedures for importing certificates into the TOE trust store. Certificates in the /opt/TLS-server/ca.crt file are considered trust anchor for the TOE.

2.29.3 Tests

The management functions have been exercised during other tests.

2.30 FMT_MTD.1/CryptoKeys Management of TSF Data (Selection-Based)

TSS

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.

Guidance Documentation

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.

Tests

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.

2.30.1 TSS

The TOE is not a distributed TOE.

[ST] Section 7.2.1 describes that the TOE supports cryptographic keys used by sshd, the ssh tunnel client, and the TLS server.

[ST] Section 7.4.3. describes that the administrator can generate keys, replace existing keys, and destroy the keys for sshd and ssh tunnel client.

[ST] Section 7.4.3 describes that the administrator can import and validate certificates (i.e. public keys) used in TLS Server.

[ST] Section 7.2.1. describes that the administrator can generate cryptographic keys for TLS server using the tls-server-csr-gen command.

[ST] Section 7.2.1. describes that the administrator can generate cryptographic keys for SSH server using the ssh-host-keygen command.

[ST] Section 7.2.1. describes that the administrator can generate cryptographic keys for SSH Client using the ssh-tunnel-keygen command.

[ST] Section 7.2.1 describes that zeroization of cryptographic keys is performed by executing the “zeroize-keyfile” command by the administrator.

[ST] Section 7.2.1. describes that the administrator can use secure copy process to import public key for SSH server.

[ST] Section 7.2.1. describes that the administrator will import TLS public keys as part of TLS certificate.

2.30.2 Guidance Documentation

[AGD] Section 14 and 15 describe managing (generating, zeroizing importing) cryptographic keys for TLS server.

[AGD] Section 10 describes managing (generating, zeroizing, importing) cryptographic keys for SSH Client.

[AGD] Section 12 describes managing (generating, zeroizing, importing) cryptographic keys for SSH Server.

2.30.3 Tests

Test Number	Test 1
Test Objective	Objective 1 - Verify, without any user authentication, the user might not be able to get to the point where an attempt to manage cryptographic keys can be executed Objective 2 - Verify an attempt to perform at least one of the related actions (modify, delete, generate/import) is successful when authenticated as a Security Administrator
Test Steps Performed	Objective 1 is satisfied by test FMT_MOF.1.1/Functions -Test 1. Objective 2 is satisfied by test FIA_X509.3.2 Test 1.
Test Result	Pass

2.31 FMT_SMF.1 Specification of Management Functions

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

TSS (containing also requirements on Guidance Documentation and Tests)

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

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

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

Guidance Documentation

See section 2.4.4.1

Tests

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.

2.31.1 TSS

[ST] Section 7.4.3 contains a list of management functions available through the local administrative interface

[ST] Section 7.4.3 contains a list of management functions available through the remote administrative interface.

During functional testing, the evaluator was able to verify that all those management functions are indeed provided by the TOE. Using [AGD] the evaluator was able to setup different warning banners for local and remote administrative interface.

[ST] Section 7.4.1 describes the local administrative interface.

2.31.2 Guidance Documentation

The evaluator conducted functional testing on the TOE and determined that all management functions specified by the [ST] Section 6.1.4.6 are provided by the TOE.

[AGD] Section 4 states that the administrative procedures are identical whether performed locally or remotely (local serial console vs. SSH and TLS).

[AGD] Section 11, subsection “Login warning banner” describes a method for setting up warning banner for local interface that can be used to identify local interface for the administrator. [AGD] Section 4 contains guidance for administrators on how to ensure the interface is local.

2.31.3 Tests

Test Number	Test 1
Test Objective	Exercise management functions as defined in the [ST]: • Ability to administer the TOE locally and remotely; • Ability to configure the access banner; • Ability to configure the session inactivity time before session termination or locking; • Ability to update the TOE, and to verify the updates using hash comparison capability prior to installing those updates; • Ability to configure the authentication failure parameters for FIA_AFL.1; • Ability to modify the behaviour of the transmission of audit data to an external IT entity; • Ability to start and stop services • Ability to manage the cryptographic keys; • Ability to configure the cryptographic functionality; • Ability to configure thresholds for SSH rekeying; • Ability to re-enable an Administrator account; • Ability to set the time which is used for time-stamps; • Ability to configure NTP • Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors; • Ability to import X.509v3 certificates to the TOE's trust store; • No other capabilities.
Test Steps Performed	The evaluator tested management functions as part of testing the corresponding SFRs.
Test Result	Pass

2.32 FMT_SMR.2 Restrictions on security roles

TSS

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.

Guidance Documentation

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.

Tests

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.

2.32.1 TSS

[ST] Section 7.4.4 describes that only the role of Security Administrator is supported by the TOE. No other roles are supported. Only one user account exists on the system.

2.32.2 Guidance Documentation

[AGD] Section 4 states that the administrative procedures are identical whether performed locally or remotely (local serial console vs. SSH and TLS).

Guidance for configuring the TSF for remote authentication is contained throughout the [AGD] document specifically in Sections 4, 11, 12, 14, and 15. See also Sections 6.13, 6.14, and 4.8 of this document.

[AGD] Section 2 provides the IT requirements of the SSH and TLS clients to connect to the TOE.

2.32.3 Tests

Test Number	Test 1
Test Objective	Ensure all supported interfaces were used during the evaluation.
Test Steps Performed	The TOE supports Local, SSH and TLS server management interfaces. The evaluator exercised each of these interfaces in the course of evaluation activities.
Test Result	Pass

2.33 FPT_APW_EXT.1 Protection of Administrator Passwords

TSS

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.

2.33.1 TSS

[ST] Sections 7.5.1 and 7.5.4 claim that all administrator passwords are subject to FPT_APW_EXT.1 SFR. They contain descriptions that passwords are stored in SHA512 hashed form.

[ST] Sections 7.5.1 and 7.5.4 claim that the TOE does not have mechanisms to read or display administrative password or authentication material even to authenticated users.

2.33.2 Guidance Documentation

None.

2.33.3 Tests

None.

2.34 FPT_SKP_EXT.1 Protection of TSF Data (for Reading of All Pre-shared, Symmetric and Private Keys)

TSS

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.

2.34.1 TSS

[ST] Section 7.5.4 claims that symmetric and private keys and other CSPs are stored as protected files on the TOE filesystem and there are no mechanisms that would allow even authorized administrator to directly read those files.

2.34.2 Guidance Documentation

None.

2.34.3 Tests

None.

2.35 FPT_STM_EXT.1 Reliable Time Stamps

TSS

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.

Guidance Documentation

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.

Tests

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.

2.35.1 TSS

[ST] Section 7.5.5 describes that the TOE gets the time from a hardware clock. This clock is adjusted by an administrator once a year to compensate for the estimated drift if TOE does not have NTP sources configured.

[ST] Section 7.5.5 describes that NTP can be used for clock synchronization by the TOE.

[ST] Section 7.5.5 describes that the administrator can manually set up the clock for the TOE.

[ST] Section 7.5.5 lists security functions that rely on time: SSH and TLS handshakes, audit log, session timeouts, and certificate validity check.

2.35.2 Guidance Documentation

[AGD] Section 4, subsection “Setting time” and [AGD] Section 8 contain instruction for Administrator on how to set the system time in the TOE.

[AGD] Section 4, subsection “Enabling the NTP Client for Time Synchronization” contains instructions on how the NTP client on the TOE should be configured.

2.35.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE time can be adjusted.
Test Steps Performed	1. Query the TOE system time. 2. Modify the system time. 3. Verify the system time is modified successfully.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE is able to obtain and setup time from the NTP server.
Test Steps Performed	This test is satisfied by performing FCS_NTP_EXT.1.2 -Test 1 and FCS_NTP_EXT.1.4 -Test 1.
Test Result	Pass

2.36 FPT_TST_EXT.1 TSF Testing

TSS

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.

Guidance Documentation

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.

Tests

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

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

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

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

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

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

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

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

2.36.1 TSS

[ST] Section 7.5.2 describes memory tests that are being performed by the TOE by writing and reading known values to memory. The [ST] claims they are sufficient to demonstrate that the TOE memory is operating correctly.

[ST] Section 7.5.2. describes that the TOE performs software integrity tests by computing a hash of system binaries and configuration files and comparing them with stored values. This check is being done continuously during runtime by the Advanced Intrusion Detection Environment subsystem of RHEL 8.1. The [ST] claims that this test is sufficient to ensure software integrity.

[ST] Section 7.5.2 states that RDRAND self-testing is performed by built-in RDRAND health tests that are compliant with NIST SP 800-90B Section 4.

[ST] Section 7.5.2 states that cryptographic known answer tests are being performed by cryptographic modules, which ensure correct cryptographic operation.

2.36.2 Guidance Documentation

[ST] Section 6.1.5.3 lists the self-tests that the TOE performs. These include the following:

- During Start-Up:
  - Memory
  - RDRAND
  - software integrity
  - cryptographic tests

[AGD] Section 18 lists the self-tests that the TOE performs; they are consistent with those listed in the [ST]. These include the following self-tests:

- RDRAND Tests
  - If test fails, system does not boot up.
  - Admin Action: Contact Bivio Support
- Memory Tests:

  - If test fails, it is indicative of a hardware issue; the system will not boot up.
  - Admin Action: Contact Bivio Support.
- Cryptographic Tests:
  - If test fails, audit records are generated. [AGD] Section 9, subsection 'Running cryptographic tests (on the OpenSSL cryptography suite)' contains the audit records related to the Cryptographic Self Tests.
  - Admin Action: Contact Bivio Support.
- Software Integrity Tests:
  - Check the report in /var/log/aide.log
  - Admin Action: Review logs to determine cause of failure and possible update the integrity database. [AGD] Section 17 contains guidance for that action.

2.36.3 Tests

Test Number	Test 1
Test Objective	Objective 1 - Verify the software integrity tests are being performed at system start-up. Objective 2 - Verify the cryptographic functionality tests are being performed at start-up.
Test Steps Performed	1. Initialize integrity database. 2. Reboot. 3. Verify the integrity check was performed after system start-up by inspecting the test logs. 4. Verify the cryptographic tests were performed after system start-up by inspecting the test logs.
Test Result	Pass

2.37 FPT_TUD_EXT.1 Trusted Update

TSS

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

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

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

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

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

Guidance Documentation

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.

Tests

The evaluator shall perform the following tests:

a) Test 1: The evaluator performs the version verification activity to determine the current version of the product. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). The evaluator obtains a legitimate update using

procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ('activation' could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

b) Test 2 [conditional]: If the TOE itself verifies a digital signature to authorize the installation of an image to update the TOE the following test shall be performed (otherwise the test shall be omitted). The evaluator first confirms that no updates are pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test. The evaluator obtains or produces illegitimate updates as defined below and attempts to install them on the TOE. The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

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

    2) An image that has not been signed

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

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

c) Test 3 [conditional]: If the TOE itself verifies a hash value over an image against a published hash value (i.e. reference value) that has been imported to the TOE from outside such that the TOE itself authorizes the installation of an image to update the TOE, the following test shall be performed (otherwise the test shall be omitted. If the published hash is provided to the TOE by the Security Administrator and the verification of the hash value over the update file(s) against the published hash is performed by the TOE, then the evaluator shall perform the following tests. The evaluator first confirms that no update is pending and then performs the version verification activity to determine the current version of the product, verifying that it is different from the version claimed in the update(s) to be used in this test.

    1) The evaluator obtains or produces an illegitimate update such that the hash of the update does not match the published hash. The evaluator provides the published hash value to the TOE and calculates the hash of the update either on the TOE itself (if that functionality is provided by the TOE), or else outside the TOE. The evaluator confirms that the hash values are different, and attempts to install the update on the TOE, verifying that this fails because of the difference in hash values (and that the failure is logged). Depending on the implementation of the TOE, the TOE might not allow the Security Administrator to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE

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

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

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

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

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

2.37.1 TSS

[ST] Section 7.5.3 describes that the current TOE software version can be determined by reading the contents of the /etc/NRDIST file.

[ST] Sections 6.1.5.4 and 7.5.3 claim that the TOE uses a published hash as a method of validating update integrity.

[ST] Section 7.5.3 describes the TOE update mechanism for updating system software and that the administrator is instructed by Administrative Guidance to verify the published hash before installation.

The evaluator verified that [ST] Section 7.5.3. describes that the update image is obtained from Bivio networks, that the candidate update hash is calculated using TOE functionality, that Administrator needs to compare the hash value with published hash, and that active Administrator authorization is necessary to proceed with installation. When hash verification fails, the Administrator is directed to contact Bivio Support service.

The TOE is not a distributed TOE.

2.37.2 Guidance Documentation

[AGD] Section 8 describes how the current software version can be queried.

[ST] Section 7.5.3. describes that installed software will become active immediately, so the TOE will not use delayed activation of updates.

[AGD] Section 19 describes that the Administrator preforms verification of the update authenticity by comparing it to a published hash. Administrator actions for successful and unsuccessful verification checks are described.

This description corresponds with the description in the TSS ([ST] Section 7.5.3.).

[AGD] Section 19 describes that the Administrator obtains a checksum file with the patch from Bivio Support Portal.

TOE is not a distributed TOE. The TOE does not use digital signatures for software updates verification.

2.37.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE is able to display a version that reflects applied TOE updates.
Test Steps Performed	1. Obtain the product version from the TOE. 2. Obtain the product patch and apply it. 3. Verify the TOE version was updated and reflects the patch.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the TOE correctly verifies digital signatures on the updates.
Test Steps Performed	The TOE does not use digital signatures to verify updates, as per [ST].
Test Result	Pass

Test Number	Test 3
Test Objective	Verify the TOE correctly verifies digital signatures on the updates.
Test Steps Performed	The TOE does not verify the update itself by comparing the calculated hash with a reference value, but expects the Security Administrator to do it and manually initiate the update process.
Test Result	Pass

2.38 FTA_SSL_EXT.1 TSF-initiated Session Locking

TSS

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.

Guidance Documentation

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.

Tests

The evaluator shall perform the following test:

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

terminated after the configured time period. If locking was selected from the component, the evaluator then ensures that re-authentication is needed when trying to unlock the session.

2.38.1 TSS

[ST] Section 7.6.1 describes that local administrative sessions are terminated after an administrator-specified period of inactivity. This period is configured by administrator.

2.38.2 Guidance Documentation

[AGD] Section 11, subsection “Login session timeout” describes that all administrative sessions are subject to termination when timeout is exceeded. The timeout period is configurable by the Administrator by changing TMOUT parameter in /etc/bashrc configuration file.

2.38.3 Tests

Test Number	Test 1
Test Objective	Verify the local administrative sessions are being terminated after the defined inactivity period.
Test Steps Performed	1. Set up the session lockout time as per [AGD]. 2. Establish a local session. Verify the local session is locked out after a specified timeout period. 3. Set up another session lockout time. 4. Establish a local session. Verify that local session is locked out after a specified period.
Test Result	Pass

2.39 FTA_SSL.3 TSF-initiated Termination

TSS

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

Guidance Documentation

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

Tests

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.

2.39.1 TSS

[ST] Section 7.6.2 describes that remote administrative sessions, both SSH and TLS based are terminated after an administrator-specified period of inactivity. This period is configured by administrator.

2.39.2 Guidance Documentation

[AGD] Section 11, subsection “Login session timeout” describes that all administrative sessions are subject to termination when timeout is exceeded. The timeout period is configurable by the Administrator by changing TMOUT parameter in /etc/bashrc configuration file.

2.39.3 Tests

Test Number	Test 1
Test Objective	Verify the TOE will terminate an inactive SSH remote administration session. Verify the TOE will terminate an inactive telnet-over-TLS remote administrative session.
Test Steps Performed	1. Using SSH establish an administrative session to the TOE. 2. Verify the session is terminated after a specified period of inactivity. 3. Using TLS establish an administrative session to the TOE. 4. Verify the session is terminated after a specified period of inactivity. 5. Change timout value. 6. Using SSH establish an administrative session to the TOE. 7. Verify the session is terminated after a new specified period of inactivity. 8. Using TLS establish an administrative session to the TOE. 9. Verify the session is terminated after a new specified period of inactivity.
Test Result	Pass

2.40 FTA_SSL.4 User-initiated Termination

TSS

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

Guidance Documentation

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

Tests

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.

2.40.1 TSS

[ST] Section 7.6.3 details that user sessions are terminated using “logout” or “exit” commands, by ending the SSH session, or by terminating SSH client.

2.40.2 Guidance Documentation

[AGD] describes local interactive session and 2 types of remote interactive sessions: via TLS and via SSH.

[AGD] Section 4 contains a description that all administrative sessions, both remote and local, can be terminated using the ‘exit’ or ‘logout’ commands.

2.40.3 Tests

Test Number	Test 1
Test Objective	Verify the documented method of session termination works as expected.
Test Steps Performed	1. Establish a local administrative session. 2. Use the exit command to terminate the session as per [AGD]. 3. Verify the session is terminated.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify the documented method of session termination works as expected.
Test Steps Performed	1. Use the exit command as per [AGD] while authenticated in the TOE using SSH. 2. Verify the session is terminated. 3. Use the exit command as per [AGD] while authenticated in the TOE using TLS. 4. Verify the session is terminated.
Test Result	Pass

2.41 FTA_TAB.1 Default TOE Access Banners

TSS

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).

Guidance Documentation

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

Tests

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.

2.41.1 TSS

[ST] Section 7.6.4 details remote (via TLS and SSH channels) and local (local console) administrative methods of access. It claims that for any administrative session, a configurable warning message will be displayed.

2.41.2 Guidance Documentation

[AGD] Section 11, subsection “Login warning banner” contains guidance how to configure the warning banner message.

2.41.3 Tests

Test Number	Test 1
Test Objective	Verify the warning banner is displayed for every access method.
Test Steps Performed	1. Modify the TOE warning banner as per [AGD]. 2. Login using each access method (Local Console, TLS, SSH), verify the modified login banner is displayed for each method.
Test Result	Pass

2.42 FTP_ITC.1 Inter-TSF Trusted Channel

TSS

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.

Guidance Documentation

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.

Tests

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

The evaluator shall perform the following tests:

a) Test 1: The evaluators shall ensure that communications using each protocol with each authorized IT entity is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

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

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

d) Test 4: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities.

   The evaluator shall, for each instance where the TOE acts as a client utilizing a secure communication mechanism with a distinct IT entity, physically interrupt the connection of that IT entity for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration shorter than the application layer timeout but of sufficient length to interrupt the network link layer.

   The evaluator shall ensure that, when the physical connectivity is restored, communications are appropriately protected and no TSF data is sent in plaintext.

   In the case where the TOE is able to detect when the cable is removed from the device, another physical network device (e.g. a core switch) shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall not be performed at the virtual node (e.g. virtual switch) and must be physical in nature.

Further assurance activities are associated with the specific protocols.

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

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

2.42.1 TSS

[ST] Section 7.7.1 identifies that a trusted channel is used for communication with remote audit server. This channel use SSHv2 protocol and the TOE is acting as the SSH Client. [ST] Section 7.2.5 describes protocol parameters the SSH client is using when establishing a connection, which was evaluated in FCS_SSHC_EXT.1.

[ST] Section 7.7.1 describes that the TOE will act as a SSH client when connecting to remote audit server and implementing the trusted channel.

2.42.2 Guidance Documentation

[ST] Section 6.1.7.1 FTP_ITC.1 contains declaration that the TOE uses SSH-based trusted channel to the remote audit server. This is the only authorized IT entity.

[AGD] Section 10 contains a description for configuring and establishing SSH trusted channel to remote syslog server. It describes that if public-key based authentication is configured for this trusted channel it will be automatically re-established in case of interruption. Password-based authentication commands that the Administrator needs to perform to establish the channel are described.

2.42.3 Tests

Test Number	Test 1.
Test Objective	Verify the trusted channel is in fact being established.
Test Steps Performed	1. Set up a trusted channel as per [AGD]. 2. Inspect the packet capture to verify that a trusted channel is in fact established. 3. Inspect records on the remote server to ensure that required data is being transferred through this channel.
Test Result	Pass

Test Number	Test 2.
Test Objective	Ensure the TOE is the side initiating the trusted channel.
Test Steps Performed	1. Establish a trusted channel from the TOE. 2. Inspect the packet capture to verify the TOE is the initiator for the trusted channel.
Test Result	Pass

Test Number	Test 3.
Test Objective	Verify the trusted channel packets are encrypted.
Test Steps Performed	1. Inspect the packet captures obtained during Test 1 and Test 2. 2. Verify the packet capture shows encrypted packets.
Test Result	Pass

Test Number	Test 4.
Test Objective	Verify the TOE reacts appropriately to any connection outage or interruption of the route to the external IT entities.
Test Steps Performed	1. Enable the packet capture. Ensure the TOE is actively using a trusted channel to send audit data to the syslog server. 2. Physically interrupt the connection for short period of time. 3. Verify the TOE restores the channel after the connection is restored. 4. Physically interrupt the connection for 20 minutes. 5. Verify the TOE re-negotiates the connection after the connection is restored.
Test Result	Pass

2.43 FTP_TRP.1/Admin Trusted Path

TSS

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.

Guidance Documentation

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

Tests

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.

2.43.1 TSS

[ST] Section 7.7.1 describes that the trusted path is implemented using SSHv2 protocol and TLSv1.2 protocol. The evaluator determined that this description is consistent with SFRs included in [ST] Section 6 and selections made in those SFRs.

[ST] Section 7.2.6 describes protocol parameters the SSH Server is using when establishing a connection, which was evaluated in FCS_SSHS_EXT.1.

[ST] Section 7.7.1 describes that the TOE will act as a server (SSH server for an SSH based connection, and TLS server for a TLS-based connection) for establishment of a trusted path.

2.43.2 Guidance Documentation

[ST] Section 6.1.7.2 FTP_TRP.1 describes that remote sessions are being established over SSH and TLS.

[AGD] Section 11 “Admin Login Authentication” states that username/password combination or public-key authorization are the methods allowed for establishing the remote administrative session to the TOE over SSH. Guidance is provided on how to establish the SSH connection using these two methods. It describes the process for a security administrative user to generate an SSH-RSA public/private key-pair and configure the TOE for administrative authentication to the TOE.

[AGD] Section 2 describes that a TLS remote session can be established via “a TLS enabled Telnet client. If that is not available, a regular Telnet client can be used via a TLS proxy.”

[AGD] Section 14 states that client should authenticate to the TOE using a password.

2.43.3 Testing

Test Number	Test 1
Test Objective	Verify the Telnet-over-TLS access method and SSH-based access method are operational.
Test Steps Performed	Testing for Telnet-over-TLS trusted paths is satisfied by FCS_TLSS_EXT.1.1 Test 1. Testing for SSH-based trusted paths is satisfied by FCS_SSHS_EXT.1.
Test Result	Pass

Test Number	Test 2
Test Objective	Verify for each remote administration communication channel data is not

	sent in plaintext.
Test Steps Performed	Testing for Telnet-over-TLS trusted paths is satisfied by FCS_TLSS_EXT.1.1 Test 3b. Testing for SSH-based trusted paths is satisfied by FCS_SSHS_EXT.1.
Test Result	Pass

3 SAR Assurance Activities and Results

3.1 ASE: Security Target Evaluation

3.1.1 General ASE:

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

Result:
For TSS EAs for SFRs, see Section 2 above.

3.1.2 ASE_TSS.1.1C for distributed TOE:

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

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

Result:
The TOE is not a distributed TOE and does not have distributed components.

3.1.3 TOE summary specification (ASE_TSS.1) for Distributed TOEs:

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

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

Result:
The TOE is not a distributed TOE.

3.2 ADV: Development

3.2.1 Basic Functional Specification (ADV_FSP.1)

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

The EAs presented in this section address the CEM work units ADV_FSP.1- 1, ADV_FSP.1-2, ADV_FSP.1-3, and ADV_FSP.1-5.

The EAs are reworded for clarity and interpret the CEM work units such that they will result in more objective and repeatable actions by the evaluator. The EAs in this SD are intended to ensure the evaluators are consistently performing equivalent actions.

The documents to be examined for this assurance component in an evaluation are therefore the Security Target, AGD documentation, and any required supplementary information required by the cPP: no additional "functional specification" documentation is necessary to satisfy the EAs. The interfaces that need to be evaluated are also identified by reference to the EAs listed for each SFR and are expected to be identified in the context of the Security Target, AGD documentation, and any required supplementary information defined in the cPP rather than as a separate list specifically for the purposes of CC evaluation. The direct identification of documentation requirements and their assessment as part of the EAs for each SFR also means that the tracing required in ADV_FSP.1.2D (work units ADV_FSP.1-4, ADV_FSP.1-6 and ADV_FSP.1-7) is treated as implicit and no separate mapping information is required for this element.

3.2.2 ADV_FSP.1-1

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

Result:
[AGD] Section 4 describes that the TOE provides the following TSF-Enforcing Interfaces:

- Management Ethernet Interface used for performing management functions over ethernet interface:
- Serial Console Port Interface used to perform management functions via serial RS-232 console interface.

The Administrator can invoke Serial Console Port Interface via serial console.

The Administrator can invoke Management Ethernet Interface via 2 logical interfaces: SSH Server (can be invoked via SSH Client) and TLS Server (can be invoked through TLS- enabled Telnet client). Requirements for those clients as methods of use for those interfaces are described in [AGD] Section 2. [AGD] Section 11 describes configuration that is required to invoke SSH Server interface using public-key authentication.

All described TSFIs will provide the Administrator with a command prompt to the TOE operating system after successful authentication. [ST] Section 3 also provides a list of non-administrative actions available through Management Ethernet Interface. When the corresponding ethernet packet is received on this interface, the TOE will:

1. Display the warning banner in accordance with FTA_TAB.1;
2. Respond to ICMP Echo Request,
3. Respond to ARP requests with ARP replies,
4. Establish TLS connection on TCP port 27777.

Those can be considered SFR-Supporting actions (2,3) and SFR-enforcing actions (1,4).

3.2.3 ADV_FSP.1-2

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

Result:

See ADV_FSP.1-1.

3.2.4 ADV_FSP.1-3

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

Result:

[AGD] Sections 12 contain the protocol parameters for SSH (client and server).

[AGD] Section 14 provides the instructions for configuring the TOE to allow for remote management of the TOE via the management Ethernet port using telnet-over-TLS. Protocol parameters are described.

3.2.5 ADV_FSP.1-4

PP Evaluation Activity:
Paragraph 561 from the CEM: “In the case where the developer has provided adequate documentation to perform the analysis called for by the rest of the work units for this component without explicitly identifying SFR-enforcing and SFR-supporting interfaces, this work unit should be considered satisfied.”

Since the rest of the ADV_FSP.1 work units will have been satisfied upon completion of the EAs, it follows that this work unit is satisfied as well.

Result:

Per [SD] Table 1 since the rest of the ADV_FSP.1 work units will have been satisfied upon completion of the EAs, it follows that this work unit is satisfied as well.

3.2.6 ADV_FSP.1-5

PP Evaluation Activity:
The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

Result:

The evaluator was able to perform tracing of TSFIs to the SFRs as follows:

Security Functional Requirements		Local Interface	Ethernet SSH	Ethernet TLS
SFR	Description			
FAU_GEN.1	Audit Data Generation	X	X	X
FAU_GEN.2	User Identity Association	X	X	X
FAU_STG.1	Protected audit trail storage (Optional)		X	
FAU_STG_EXT.1	Protected Audit Event Storage		X	
FAU_STG_EXT.3 /LocSpace	Action in case of possible audit data loss (Optional)		X	
FCS_CKM.1	Cryptographic Key Generation (Refinement)		X	X
FCS_CKM.2	Cryptographic Key Establishment (Refinement)		X	X
FCS_CKM.4	Cryptographic Key Destruction	X	X	X
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)		X	X

Security Functional Requirements		Local Interface	Ethernet SSH	Ethernet TLS
SFR	Description			
FCS_COP.1/SigGen	Cryptographic Operation (Signature Verification)			X
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)	X	X	X
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)		X	
FCS_NTP_EXT.1	NTP Protocol (Selection-based)	X	X	X
FCS_RBG_EXT.1	Random Bit Generation		X	X
FCS_SSHC_EXT.1	SSH Client (Selection-based)	X	X	X
FCS_SSHS_EXT.1	SSH Server (Selection-based)		X	
FCS_TLSS_EXT.1	TLS Server Protocol (Selection-based)			X
FIA_AFL.1	Authentication Failure Management (Refinement)	X	X	X
FIA_PMG_EXT.1	Password Management	X	X	X
FIA_UIA_EXT.1	User Identification and Authentication	X	X	X
FIA_UAU_EXT.2	Password-based Authentication Mechanism	X	X	X
FIA_UAU.7	Protected Authentication Feedback	X	X	X
FIA_X509_EXT.1/Rev	X.509 Certificate Validation (Selection-based)	X	X	X
FIA_X509_EXT.2	X.509 Certificate Authentication (Selection-based)			X
FIA_X509_EXT.3	X.509 Certificate Requests (Selection-based)	X	X	X
FMT_MOF.1 /Functions	Management of security functions behavior (Selection-based)	X	X	X
FMT_MOF.1 /ManualUpdate	Management of security functions behavior	X	X	X
FMT_MOF.1 /Services	Management of security functions behavior (Selection-based)	X	X	X
FMT_MTD.1 /CoreData	Management of TSF Data	X	X	X
FMT_MTD.1 /CryptoKeys	Management of TSF data (Selection-based)	X	X	X
FMT_SMF.1	Specification of Management Functions	X	X	X
FMT_SMR.2	Restrictions on security roles	X	X	X
FPT_APW_EXT.1	Protection of Administrator Passwords	X	X	X
FPT_TST_EXT.1	TSF Testing (Extended)	X	X	X
FPT_TUD_EXT.1	Trusted Update	X	X	X
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)	X	X	X
FPT_STM_EXT.1	Reliable Time Stamps	X	X	X
FTA_SSL_EXT.1	TSF-initiated Session Locking	X	X	X
FTA_SSL.3	TSF-initiated Termination (Refinement)	X	X	X
FTA_SSL.4	User-initiated Termination (Refinement)	X	X	X
FTA_TAB.1	Default TOE Access Banners (Refinement)	X	X	X
FTP_ITC.1	Inter-TSF trusted channel (Refinement)	X	X	X

Security Functional Requirements		Local Interface	Ethernet SSH	Ethernet TLS
SFR	Description			
FTP_TRP.1/Admin	Trusted Path (Refinement)		X	X

3.2.7 ADV_FSP.1-6

PP Evaluation Activity:
EAs that are associated with the SFRs in Section 2, and, if applicable, Sections 3 and 4, are performed to ensure that all the SFRs where the security functionality is externally visible (i.e. at the TSFI) are covered. Therefore, the intent of this work unit is covered.

Result:

Per [SD], these activities are covered by other evaluation activities.

3.2.8 ADV_FSP.1-7

PP Evaluation Activity:
EAs that are associated with the SFRs in Section 2, and, if applicable, Sections 3 and 4, are performed to ensure that all the SFRs where the security functionality is externally visible (i.e. at the TSFI) are addressed, and that the description of the interfaces is accurate with respect to the specification captured in the SFRs. Therefore, the intent of this work unit is covered.

Result:

Per [SD], these activities are covered by other evaluation activities.

3.3 AGD: Guidance Documents

3.3.1 Operational User Guidance (AGD_OPE.1)

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.

PP Evaluation Activities:
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 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[6]:
    i. 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).
    ii. 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.

Result:

[ST] Sections 1.4.1 and 9 contain references to TOE Guidance; therefore, Administrators and users can be expected to be aware of its existence.

[AGD] Section 3 contains a description that Operational Guidance can be obtained via the support portal and will also be included with the shipment of the product.

The [ST] and [AGD] describe the only Operational Environment for the TOE; therefore, the [AGD] addresses a single Operational Environment.

[ST] Section 1.4.1 and [AGD] Section 1 describe the TOE hardware platform.

[AGD] Section 4 describes the difference in the TOE hardware models' form factor that is relevant to TOE preparation and operation.

[AGD] Section 5 contains instruction for the administrator not to change any cryptographic engine parameters as they are pre-configured at the factory. [AGD] does list those pre-configured parameters as a reference material for SSH and TLS functionality (See Sections 6.13-6.15 of this document).

[AGD] Section 1 contains the statement, "This document describes how the Bivio 6310-NC must be configured in order to conform with a CC evaluated configuration. Only the configuration items relevant for compliance are discussed in this document". The evaluator determined that the [AGD] contains only descriptions of functionality described in the [ST].

[ST] Section 1.3.2 describes that applications in the application section of the TOE are out of scope of the evaluation. [AGD] Section 4 describes interfaces assigned for these applications and differentiates them from the management interface covered by the evaluation activities. [AGD] Section 21 describes how these applications can be started.

The evaluator examined update process description and verified that it contains description how to obtain update and validate update hush.

[ST] Section 1.3.2 describes that applications in the application section of the TOE are out of scope of the evaluation. [AGD] Section 4 describes interfaces assigned for these applications and differentiates them from the management interface covered by the evaluation activities.

3.3.2 Preparative Procedures (AGD_PRE.1)

PP Evaluation Activities:

[6] The Evaluation Activities was modified by TD0536

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

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

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

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

The preparative procedures must:

a) include instructions to provide a protected administrative capability; and
b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

Result:
[AGD] Sections 2 contains the IT Requirements for the Operational Environment, including:

- Syslog server conformant to RFCs 5424 (Syslog over TCP), capable of receiving an SSH tunnel;
- Local console with a RS-232 port;
- SSH Client conformant to RFCs 4251, 4252, 4253, 4254, and 6668. The SSH client must support AES128-CBC and AES256-CBC encryption algorithms, using HMAC-SHA2-256 or HMAC-SHA2-512 integrity algorithms, and performing key exchange using Diffie-Hellman Group14-SHA1;
- Telnet Client that supports TLS 1.2 and TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268 and TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

There is no specific verification of the IT operational environment included, and the evaluator determined that none is required.

[AGD] Section 3 contains procedural requirements for the Operational Environment.

The TOE should be placed in a physically secured environment; the Administrator is required to follow and apply the guidance document faithfully.

TOE updates should be installed by the Administrator on a regular basis. Administrator credentials (or keys) must be protected.

The evaluator determined that these requirements are in sufficient detail for the target audience to understood and follow them. There is no specific verification of the operational environment included, and the evaluator determined this is not required.

[AGD] Section 1 contains a list of all model variations for the TOE, which is consistent with the list in Section 1.4.1 of the [ST].

There are 3 types of TOE platform models designated by part numbers based on configuration:

1. B6310-NC-C(x,y)M(1,2,3,4,5)D(1,2,3,4,5,6)N(1,2,3,4)
2. B6310R-NC-C(x,y)M(1,2,3)D(1,2,3,4,5,6)N(1,2,4)
3. PacStar 451

The first two types of models differ only in number of network interfaces, storage size, RAM size and CPU. PacStar 451 (also referred to as C04 in [AGD]) will also have a different hardware layout.

[AGD] Section 4 Subsection “Initial Configuration” provides different instructions for PacStar 451 and other models, addressing the difference in TOE form-factor and hardware ports layout.

[ST] Section 1.3.4 defines only one Operational Environment, this OE described in [AGD] applies to each of the TOE models or configurations as described in [ST] Section 1.4.1.

[AGD] Section 4 includes instructions for TOE installation and initial configuration. Sections 5-15 of [AGD] provide guidance for putting the installed TOE into the evaluated configuration by configuring TOE functions:

- Setting up network configuration
- Setting up time
- User accounts
- Timeouts and warning banners
- SSH Server configuration
- TLS Server configuration
- Password length (if necessary)
- SSH host keys (if necessary)
- The SSH tunnel for conveying audit logs to the remote syslog server
- The TLS server
- Parameters for account locking if password retries exceed the maximum allowed number of retries

The evaluator determined that the TOE is a stand-alone network device. The TOE uses other operational environment instances, like remote syslog server and NTP servers. [AGD] Section 4, subsection “Initial Configuration” contains a description of how network settings, that are dictated by the larger operational environment are configured.

[ADG] Section 4, subsection ‘Initial Configuration” describes that the system has login “root” with default password “root” and it contains guidance to immediately change this default password. [AGD] Section 6 prescribes that “root” login must be stored securely and not to be used in the certified configuration.

[AGD] Section 4, subsection ‘Initial Configuration” describes that the system has login “admin” with default password “admin” and it contains guidance to immediately change this default password. [AGD] Section 6 prescribes that administration should be done using this account and that administrative actions can be performed from this account with “sudo” prefix. It prescribes that elevation of right should be terminated immediately after the administrative task is completed.

[AGD] Section 7 further describes how to setup login retries and the account locking policy to ensure protected administrative capability.

3.4 ALC: Life-cycle Support

3.4.1 Labelling of the TOE (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.
Result:

The evaluator inspected the TOE labelling and concluded that it is labelled in accordance with [ST] Section 1.4.1.

The evaluator inspected labeling on the TOE parts. The delivery package contained an optical disk with guidance that is labeled consistently with the description in the [ST] Section 9.

3.4.2 TOE CM Coverage (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.

Result:
[ST] contains a list and identifications for all parts of the TOE.

[ST] Section 1.2 identifies the TOE itself and [ST] Section 1.4.1 identifies the TOE model numbers and TOE software version.

[ST] Section 9 contains a reference to the TOE Guidance documentation, including document name, date and version.

During the evaluation the evaluator observed that the TOE software was modified by the developer and each software version provided was uniquely identified by version number.

3.5 ATE: Tests

3.5.1 Independent Testing (ATE_IND.1)

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 709 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 A.9.3.1.

Result:
The evaluator performed independent testing of the TOE as prescribed in the [SD] and CEM. Test results are captured in Section 2. Test results confirmed the TOE functionality is correctly described in the TSS and that requirements for the SFR has been met by the TOE.

Due to the equivalency claim, testing was only performed on one model of the TOE.

The TOE is not a distributed TOE.

3.6 AVA: Vulnerability Assessment

3.6.1 Vulnerability Survey (AVA_VAN.1)

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.

Evaluation 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 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.

3.6.1.1 AVA_VAN Type 1 Hypothesis – Public-Vulnerability-Based

The evaluator conducted a search on public sources for known vulnerabilities that are relevant to the TOE using the following search terms: “rhel, redhat, openssh, stunnel, openssl 1.1.1, openssl, chrony, AIDE, TCP, TLS, SSH, BIVIO, intel, xeon, Broadcom, matrox, ntp”. The search was performed on August 13, September 29, October 13, and November 24, 2020.

All identified vulnerabilities were analysed.

After duplicates were removed and entries related to technologies that are not part of the TOE were removed 58 candidate records remained:

CVE-2020-8687, CVE-2020-8674, CVE-2020-5326, CVE-2020-1967, CVE-2020-1749, CVE-2020-15778, CVE-2020-14145, CVE-2020-12301, CVE-2020-12300, CVE-2020-12299, CVE-2020-12287, CVE-2020-12114, CVE-2020-10768, CVE-2020-10767, CVE-2020-10766, CVE-2020-0600, CVE-2020-0598, CVE-2020-0597, CVE-2020-0596, CVE-2020-0595, CVE-2020-0594, CVE-2020-0586, CVE-2020-0568, CVE-2020-0564, CVE-2020-0563, CVE-2020-0562, CVE-2020-0551, CVE-2020-0550, CVE-2020-0549, CVE-2020-0548, CVE-2020-0546, CVE-2020-0545, CVE-2020-0543, CVE-2020-0542, CVE-2020-0541, CVE-2020-0540, CVE-2020-0539, CVE-2020-0538, CVE-2020-0537, CVE-2020-0536, CVE-2020-0535, CVE-2020-0534, CVE-2020-0533, CVE-2020-0532, CVE-2020-0531, CVE-2020-0530, CVE-2020-0529, CVE-2020-0528, CVE-2020-0527, CVE-2020-0526, CVE-2019-19339, CVE-2019-19338, CVE-2019-17006, CVE-2019-14607, CVE-2020-24457, CVE-2020-1968, CVE-2020-24659, CVE-2020-14367, CVE-2020-057, CVE-2020-8671.

Vulnerabilities that are related to the TOE but that would be impossible to exploit in the evaluated configuration were disproved. Remaining vulnerabilities were reported to the vendor as listed below:

CVE-2020-14145 ,CVE-2020-0597, CVE-2020-0596, CVE-2020-0595, CVE-2020-0536, CVE-2020-0535, CVE-2020-0534, CVE-2020-0532, CVE-2019-17006, Advisory RHSA-2020:3280, Advisory RHSA-020:3216, Advisory RHSA-2020:3218, Advisory RHSA-2020:3185, Advisory RHSA-2020:3073, Advisory RHSA-2020:3032, Advisory RHSA-2020:3014, Advisory RHSA-2020:3011, Advisory RHSA-2020:3010, Advisory RHSA-2020:2774, Advisory RHSA-2020:2755, Advisory RHSA-2020:2637, Advisory RHSA-2020:2567, Advisory RHSA-2020:2427, Advisory RHSA-2020:2431, Advisory RHSA-2020:2416, Advisory RHSA-2020:2338, Advisory RHSA-2020:2125, Advisory RHSA-2020:2102, Advisory RHSA-2020:2070, Advisory RHSA-2020:1998, Advisory RHSA-2020:1980, Advisory RHSA-2019:1223.

For those advisories the vendor has provided a justification of non-applicability to the TOE and provided a fix that addressed all identified issues.

3.6.1.2 AVA_VAN Type 2 Hypothesis – iTC-Sourced

Since the TOE claims TLS server, the evaluator verified the ITC-based flaw hypothesis.

The evaluator enabled the TLS server and verified that the TOE is not vulnerable to the ROBOT Padding Oracle attack.

The evaluator did not formulate any hypothesis based on TOE equivalency considerations.

3.6.1.3 AVA_VAN Type 3 Hypothesis – Evaluation-Team-Generated

The evaluator used an nmap port scanner tool to verify that the TOE does not expose open ports that may be used for an attack and that were not covered by the evaluation.

3.6.1.4 AVA_VAN Type 4 Hypothesis – Tool-Generated

The evaluator performed protocol fuzzing as prescribed by the [SD].

The evaluator observed that the TOE reacted as expected to the fuzzing activities and it did not force the TOE to malfunction or enter an insecure state.

3.6.1.5 AVA_VAN Reporting

The certification body report is the ETR.

The public facing report is this Assurance Activity Report.

4 References

Abbr.	Name	Version	Date
[PP]	collaborative Protection Profile for Network Devices	2.2e	March 23, 2020
[SD]	Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP	2.2	December 2019
[ST]	Bivio 6310-NC Security Target	0.8	November 25, 2020
[AGD]	Bivio 6310-NC Common Criteria Administrative Guidance	1.10	November 23, 2020