Assurance Activities Report

for

RedSeal Server v10.5

Version 1.0

October 17, 2025

Prepared by:

Leidos Inc.
https://www.leidos.com/capabilities/cyber/accredited-testing-evaluation
Common Criteria Testing Laboratory
6841 Benjamin Franklin Drive
Columbia, MD 21046

Prepared for:

RedSeal, Inc.
1600 Technology Drive, 4th Floor
San Jose, CA 95110

The Developer of the TOE:

RedSeal, Inc.
1600 Technology Drive, 4th Floor
San Jose, CA 95110

The TOE Evaluation was Sponsored by:

RedSeal, Inc.
1600 Technology Drive, 4th Floor
San Jose, CA 95110

Evaluation Personnel:

Greg Beaver
Armin Najafabadi
Allen Sant
Pascal Patin

Contents

1 Introduction 6

1.1 Applicable Technical Decisions 6
1.2 Evidence 7
1.3 Conformance Claims 8
1.4 SAR Evaluation 8

2 Security Functional Requirement Evaluation Activities 10

2.1 Security Audit (FAU) 10

2.1.1 FAU_GEN.1 Audit Data Generation 10

2.1.2 FAU_GEN.2 User Identity Association 12

2.1.3 FAU_STG_EXT.1 Protected Audit Event Storage 12

2.1.4 FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss 18

2.1.5 FAU_STG.1 Protected Audit Trail Storage 19

2.2 Cryptographic Support (FCS) 21

2.2.1 FCS_CKM.1 Cryptographic Key Generation 24

2.2.2 FCS_CKM.2 Cryptographic Key Establishment 27

2.2.3 FCS_CKM.4 Cryptographic Key Destruction 31

2.2.4 FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption) 34

2.2.5 FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification 35

2.2.6 FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm) 38

2.2.7 FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm) 39

2.2.8 FCS_HTTPS_EXT.1 HTTPS Protocol 40

2.2.9 FCS_NTP_EXT.1 NTP Protocol 41

2.2.10 FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation) 45

2.2.11 FCS_SSH_EXT.1 SSH Protocol 46

2.2.12 FCS_SSHS_EXT.1 SSH Server Protocol 55

2.2.13 FCS_TLSC_EXT.1 TLS Client Protocol 56

2.2.14 FCS_TLSS_EXT.1 TLS Server Protocol 71

2.3 Identification and Authentication (FIA) 83

2.3.1 FIA_AFL.1 Authentication Failure Management 83

2.3.2 FIA_PMG_EXT.1 Password Management 86
2.3.3 FIA_UIA_EXT.1 User Identification and Authentication 87
2.3.4 FIA_X509_EXT.1/Rev X.509 Certificate Validation 90
2.3.5 FIA_X509_EXT.2 X.509 Certificate Authentication 96
2.3.6 FIA_X509_EXT.3 X.509 Certificate Requests 98
2.4 Security Management (FMT) 99
2.4.1 FMT_MOF.1/Functions Management of Security Functions Behaviour 99
2.4.2 FMT_MOF.1/ManualUpdate Management of Functions Behavior 103
2.4.3 FMT_MTD.1/CoreData Management of TSF Data 104
2.4.4 FMT_SMF.1 Specification of Management Functions 108
2.4.5 FMT_SMR.2 Restrictions on Security Roles 110
2.5 Protection of the TSF (FPT) 112
2.5.1 FPT_APW_EXT.1 Protection of Administrator Passwords 112
2.5.2 FPT_SKP_EXT.1 Protection of TSF Data (for Reading of All Pre-shared, Symmetric, and Private Keys) 112
2.5.3 FPT_STM_EXT.1 Reliable Time Stamps 113
2.5.4 FPT_TST_EXT.1 TSF Testing 115
2.5.5 FPT_TUD_EXT.1 Trusted Update 117
2.6 TOE Access (FTA) 121
2.6.1 FTA_SSL.3 TSF-initiated Termination 121
2.6.2 FTA_SSL.4 User-initiated Termination 122
2.6.3 FTA_TAB.1 Default TOE Access Banners 123
2.7 Trusted Path/Channels (FTP) 124
2.7.1 FTP_ITC.1 Inter-TSF Trusted Channel 124
2.7.2 FTP_TRP.1/Admin Trusted Path126
3 Security Assurance Requirements 129
3.1 Class ASE: Security Target Evaluation 129
3.1.1 ASE_TSS.1 TOE Summary Specification for Distributed TOEs 129
3.2 Class ADV: Development 129
3.2.1 ADV_FSP.1 Basic Functional Specification 129
3.3 Class AGD: Guidance Documents 131
3.3.1 AGD_OPE.1 Operational User Guidance 131

3.3.2 AGD_PRE.1 Preparative Procedures 133

3.4 Class ALC: Life-Cycle Support 135
3.4.1 ALC_CMC.1 Labeling of the TOE 135

3.4.2 ALC_CMS.1 TOE CM Coverage 135

3.5 Class ATE: Tests 135
3.5.1 ATE_IND.1 Independent Testing – Conformance 135

3.6 Class AVA: Vulnerability Assessment 135
3.6.1 AVA_VAN.1 Vulnerability Survey 135

1 Introduction

This document presents results from performing evaluation activities associated with the evaluation of RedSeal Server v10.5. This report contains sections documenting the performance of evaluation activities associated with each of the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) as specified in the Evaluation Activities for Network Device cPP, Version 3.0e, 06 December 2023 [ND-SD] and the SFRs in the Functional Package for Secure Shell (SSH), Version 1.0, 2021-05-13 [SSHPKG].

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

1.1 Applicable Technical Decisions

The NIAP Technical Decisions referenced below apply to [CPP_ND_V3.0E]. Rationale is included for those Technical Decisions that do not apply to this evaluation.

NIAP has issued the following Technical Decisions, applicable to [NDcPP] and [SSHPKG]. Rationale is included for those Technical Decisions that do not apply to this evaluation.

The NIAP Technical Decisions referenced below apply to [CPP_ND_V3.0E]. Rationale is included for those Technical Decisions that do not apply to this evaluation.

- TD0836 – NIT Technical Decision: Redundant Requirements in FPT_TST_EXT.1 [NDcPP]
    - This TD is applicable to the TOE.
- TD0868 – NIT Technical Decision: Clarification of time frames in FCS_IPSEC_EXT.1.7 and FCS_IPSEC_EXT.1.8
    - This TD is not applicable because the TOE does not use IPSec
- TD0879 – NIT Technical Decision: Correction of Chapter Headings in CPP_ND_V3.0E
    - This TD is not applicable because the TOE does not offer a local interface
- TD0880 – NIT Technical Decision: Removal of Duplicate Selection in FMT_SMF.1.1
    - This TD is an administrative update to the PP that removes a duplicate selection and changes the wording of the selection. This TD is applicable to the TOE.
- TD0886 – Clarification to FAU_STG_EXT.1 Test 6
    - This TD adds an application note to the test EA and is applicable to the TOE.

- TD0899 – NIT Technical Decision: Correction of Renegotiation Test for TLS 1.2
    - This TD corrects a test activity in FCS_TLSC_EXT.1 and FCS_TLSS_EXT.1 that is applicable to the TOE.
- TD0900 – NIT Technical Decision: Clarification to Local Administrator Access in FIA_UIA_EXT.1.3
    - This TD updates FIA_UIA_EXT.1 and is applicable to the TOE.
- TD0921 – NIT Technical Decision: Addition of FIPS PUB 186-5 and Correction of Assignment
    - This TD updates FCS_CKM.1 and is applicable to the TOE.
- TD0923 – NIT Technical Decision: Auditable event for FAU_STG_EXT.1 in FAU_GEN.1.2
    - This TD is applicable to the TOE but does not affect the contents of this ST as it only affects the application note.

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

- TD0682 - Addressing Ambiguity in FCS_SSHS_EXT.1 Tests
    - This TD archives TD0666 and is applicable to the TOE but relates solely to evaluation activities so it does not affect the ST.
- TD0695 - Choice of 128 or 256 bit size in AES-CTR in SSH Functional Package
    - This TD modifies an explanation in the PP but does not change any requirements. This TD is applicable to the TOE.
- TD0732 - FCS_SSHS_EXT.1.3 Test 2 Update
    - This TD archives TD0694 and is applicable to the TOE but relates solely to the evaluation activities so it does not affect the ST.
- TD0777 - Clarification to Selections for Auditable Events for FCS_SSH_EXT.1
    - This TD is applicable to the TOE because it logs failure to establish SSH connection.
- TD0909 - Updates to FCS_SSH_EXT.1.1 App Note in SSH FP 1.0
    - This TD is applicable to the TOE, since it modifies a Package Application Note.

There were no observation reports submitted during this evaluation.

1.2 Evidence

[ST] RedSeal Server v10.5 Security Target, Version 1.0, 17 October 2025

[CCECG] RedSeal Server v10.5 Common Criteria Evaluated Configuration Guide (CCECG), Version 1.0, October 24, 2025

[ADMIN] RedSeal Installation and Administration Guide, Version 10.5.2

1.3 Conformance Claims

Common Criteria Versions

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

Common Evaluation Methodology Versions

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

Protection Profiles

- collaborative Protection Profile for Network Devices, Version 3.0e, 06 December 2023 [NDcPP], including the following optional and selection-based SFRs: FAU_STG.1; FAU_STG_EXT.3; FCS_HTTPS_EXT.1; FCS_NTP_EXT.1; FCS_TLSC_EXT.1; FCS_TLSS_EXT.1; FIA_AFL.1; FIA_PMG_EXT.1; FIA_X509_EXT.1/Rev; FIA_X509_EXT.2; FIA_X509_EXT.3; FMT_MOF.1/Functions; FMT_MTD.1/CryptoKeys; and FPT_APW_EXT.1,
- Functional Package for Secure Shell (SSH), Version 1.0, 13 May 2021 [SSHPKG], including the following selection-based SFRs: FCS_SSHS_EXT.1.

1.4 SAR Evaluation

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

SAR	Verdict
ASE_CCL.1	Pass
ASE_ECD.1	Pass
ASE_INT.1	Pass
ASE_OBJ.1	Pass
ASE_REQ.1	Pass
ASE_SPD.1	Pass
ASE_TSS.1	Pass
ADV_FSP.1	Pass

SAR	Verdict
AGD_OPE.1	Pass
AGD_PRE.1	Pass
ALC_CMC.1	Pass
ALC_CMS.1	Pass
ATE_IND.1	Pass
AVA_VAN.1	Pass

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

2 Security Functional Requirement Evaluation Activities

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

Evaluator notes, such as changes made as a result of NIAP Technical Decisions, are highlighted in bold text, as are changes made as a result of NIAP Technical Decisions. Bold text is also used within evaluation activities to identify when they are mapped to individual SFR elements rather than the component level.

2.1 Security Audit (FAU)

2.1.1 FAU_GEN.1 Audit Data Generation

2.1.1.1 TSS Activities

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

Section 6.1.1 of [ST] asserts under FAU_GEN.1 that generating/import of, changing, or deleting of cryptographic keys—the TOE logs the specific administrator action that was performed and identifies cryptographic keys either by identifying the certificate associated with the key, using the certificate subject identifier and certificate issuer identifier.

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

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

2.1.1.2 Guidance Activities

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

Section “Audit Records” of the [CCECG] provides a sample of each audit record type that the TOE must generate.

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.

All the administrative actions related to TSF data related to configuration changes have an audit record in the CCECG.

2.1.1.3 Test Activities

The evaluator shall test the TOE’s ability to correctly generate audit records by having the TOE generate audit records for the events listed in the table of audit events and administrative actions listed above. This should include all instances of an event: for instance, if there are several different identity and authentication (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 cPP_ND_v3.0e-SD, 06-Dec-2023 16 generated for the establishment and termination of a channel for each of the cryptographic protocols contained in the ST. If HTTPS is implemented, the test demonstrating the establishment and termination of a TLS session can be combined with the test for an HTTPS session. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the guidance documentation, and that the fields in each audit record have the proper entries.

The evaluator examined the audit records produced by the TOE during the testing process. All of the events required by FAU_GEN.1 as well as the ones listed in Table 4 of the [ST] were generated by the TOE.

Each audit record was checked for content, and was found to have a timestamp when it was recorded. Records also recorded IP addresses, the identity of users who took actions (when relevant), information on event type and the outcome of the audited event.

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.

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

2.1.2 FAU_GEN.2 User Identity Association

2.1.2.1 TSS Activities

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

2.1.2.2 Guidance Activities

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

2.1.2.3 Test Activities

This activity should be accomplished in conjunction with the testing of FAU_GEN.1.1.
For distributed TOEs the evaluator shall verify that where auditable events are instigated by another component, the component that records the event associates the event with the identity of the instigator. The evaluator shall perform at least one test on one component where another component instigates an auditable event. The evaluator shall verify that the event is recorded by the component as expected and the event is associated with the instigating component. It is assumed that an event instigated by another component can at least be generated for building up a secure channel between two TOE components. If for some reason (could be e.g. TSS or Guidance Documentation) the evaluator would come to the conclusion that the overall TOE does not generate any events instigated by other components, then this requirement shall be omitted.

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

2.1.3 FAU_STG_EXT.1 Protected Audit Event Storage

2.1.3.1 TSS Activities

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

Section 6.1.2 of [ST] states that the TOE is a single standalone appliance that is able to store generated audit records locally (i.e., on the appliance). The TOE can be configured to export audit records to an external audit server (i.e., external syslog server), over a trusted channel protected by TLS.

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.

Section 6.1.2 of [ST] states that the TOE is a single standalone appliance that is able to store generated audit records locally (i.e., on the appliance). The TOE can be configured to export audit records to an external audit server (i.e., external syslog server), over a trusted channel protected by TLS.

The evaluator shall examine the TSS to ensure that it details whether the transmission of audit data to an external IT entity can be done in real-time, periodically, or both. In the case where the TOE is capable of performing transmission periodically, the evaluator shall verify that the TSS provides details about what event stimulates the transmission to be made as well as the possible acceptable frequency for the transfer of audit data.

[ST] section 6.1.2 indicates that the TOE can be configured to export audit records to an external audit server over a trusted channel protected by TLS. In this circumstance, the TOE acts as a TLS client. The audit records are exported in real time (i.e., as they are generated). Additionally, the TOE provides a manual export mechanism that allows Administrators to download the locally stored audit records in order to view them.

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).

This evaluation activity is not applicable; the TOE is not distributed.

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

[ST] section 6.1.2 indicates that the TOE is a single standalone appliance that is able to store generated audit records locally (i.e., on the appliance). The TOE maintains the following logs that together constitute the audit trail:

- Audit—contains records of all configuration changes made in the CLI.

- System—contains records of system events, including server starts, stops, and restorations.
- Analyzer—contains records of data collection events, including date, time, name of the event, name of the credential used, and the communication method used. The analyzer log also contains records of the results of analysis events.
- Server—contains all log messages generated by TOE server and database processes, including records in the Audit, System, and Analyzer log files.

For each of these logs, the TOE maintains a configurable number of log files on the appliance—the current log file and the most recently created log files up to the configured number minus one. For example, if the TOE is configured to maintain five files (the default) for each log, the log will consist of the current file and the four most recently created log files.

Log files are rotated based on a configurable schedule that can be specified in terms of frequency or size (but not both—size is used by default). The size must be in the range of 1,000 Bytes to 1,000MB (default is 50MB). When the current log file (for each of the logs defined above) reaches its rotation threshold, it is closed and a new current log file is created. If the configured maximum number of files for that log already exists, the oldest one is deleted. The maximum amount of space available on the TOE for all logs is 2GB.

The evaluator shall examine the TSS to ensure it describes the method implemented for local logging, including format (e.g. buffer, log file, database) and whether the logs are persistent or non-persistent.

Section 6.1.2 of [ST] states that audit data

For each of audit logs, the TOE maintains a configurable number of log files on the appliance—the current log file and the most recently created log files up to the configured number minus one. For example, if the TOE is configured to maintain five files (the default) for each log, the log will consist of the current file and the four most recently created log files.

Log files are rotated based on a configurable schedule that can be specified in terms of frequency or size (but not both—size is used by default). The size must be in the range of 1,000 Bytes to 1,000MB (default is 50MB). When the current log file (for each of the logs defined above) reaches its rotation threshold, it is closed and a new current log file is created. If the configured maximum number of files for that log already exists, the oldest one is deleted. The maximum amount of space available on the TOE for all logs is 2GB.

The evaluator shall examine the TSS to ensure it describes the conditions that must be met for authorized deletion of audit records.

Section 6.1.2 of [ST] states that the TOE stores the logs comprising the audit trail are stored in the TOE's file system and protected from unauthorized modification and deletion by file system permissions. Log files are deleted when the oldest log is rotated out by the log rotation settings.

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

[ST] Section 6.1.2 of [ST] states that log files are rotated based on a configurable schedule that can be specified in terms of frequency or size (but not both—size is used by default):

- The TOE maintains a configurable number of log files on the appliance—the current log file and the most recently created log files up to the configured number minus one. For example, if the TOE is configured to maintain five files (the default) for each log, the log will consist of the current file and the four most recently created log files.
- The Administrator can configure logs to be rotated daily, weekly, or monthly, or when they reach a maximum size. The size must be in the range of 1,000 Bytes to 1,000MB (default is 50MB). When the current log file (for each of the logs defined above) reaches its rotation threshold, it is closed and a new current log file is created. If the configured maximum number of files for that log already exists, the oldest one is deleted. The maximum amount of space available on the TOE for all logs is 2GB.

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.

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

2.1.3.2 Guidance Activities

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

[CCECG] Section "Configure a syslog server" identifies the `set log` command to configure a syslog server.

Further the [CCECG} states: The remotehost parameter hostname must point to a syslog server. Configuring the hostname parameter with the FQDN will establish the reference identifier. You must specify either primary or secondary, followed by the hostname (FQDN) for the log server, or the reset command argument. As part of establishing the remote syslog connection, the FQDN of the configured syslog server is compared against the CN or SAN (if present) in the presented certificate to verify the identity of the host matches the configured value, and the X.509 properties of the certificate are checked to verify that the certificate itself is valid.

The only requirement for the audit server used by the TOE is that it is a syslog server, and that it supports TLS 1.2.

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

[CCECG] Section "Configure a syslog server" states that after the syslog server is configured, audit records will continue to be logged locally on the appliance.

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

N/A – no configuration is required for protection of audit data stored locally.

If the storage size is configurable, the evaluator shall review the Guidance Documentation to ensure it contains instructions on specifying the required parameters.

[CCECG] Section "Set audit log rotation parameters" identifies the `set log` command to set log rotation parameters.

Set <`num_logs`> to the number of files to be kept on the appliance at any one time, for the named log type. If `num_logs` is 5 (the default), the five most recently created log files are kept on the system, for each of the log types. The frequency and size parameters are mutually exclusive. Logs are rotated either on a set schedule or when the log file reaches a specified size. If you try to set both arguments in the same command, the command fails.

[CCECG] Section "set log command details" provides additional details and parameters for the `set log` command.

If more than one selection is made for FAU_STG_EXT.1.5, the evaluator shall review the Guidance Documentation to ensure it contains instructions on specifying which action is performed when the local storage space is full.

[CCECG] Section "Set audit log rotation parameters" identifies the `set log` command to set log rotation parameters.

Set <`num_logs`> to the number of files to be kept on the appliance at any one time, for the named log type. If num_logs is 5 (the default), the five most recently created log files are kept on the system, for each of the log types. The frequency and size parameters are mutually exclusive. Logs are rotated either on a set schedule or when the log file reaches a specified size. If you try to set both arguments in the same command, the command fails.

[CCECG] Section "set log command details" provides additional details and parameters for the `set log` command.

2.1.3.3 Test Activities

Testing of secure transmission of the audit data externally (FTP_ITC.1) and, where applicable, intercomponent (FPT_ITT.1 or FTP_ITC.1) shall be performed according to the assurance activities for the particular protocol(s).

The evaluator shall perform the following additional test for this requirement:

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

The evaluator configured the TOE to send audit records to an audit server protected by TLS. The evaluator confirmed that the audit server received the audit records and the packets were protected via TLS in transit.

Test 2: For distributed TOEs, Test 1 defined above shall be applicable to all TOE components that forward audit data to an external audit server.

This test is not applicable as the TOE is not a distributed TOE.

Test 3: The evaluator shall perform operations that generate audit data and verify that this data is stored locally. The evaluator shall then make note of whether the TSS claims persistent or non-persistent logging and perform one of the following actions:

If persistent logging is selected, the evaluator shall perform a power cycle of the TOE and ensure that following power on operations the log events generated are still maintained within the local audit storage.

If non-persistent logging is selected, the evaluator shall perform a power cycle of the TOE and ensure that following power on operations the log events generated are no longer present within the local audit storage.

The evaluator observed the oldest audit records present on the TOE, then rebooted the device and verified the oldest audit records were still present as the TOE implements a persistent log store.

Test 4: 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.1.5. Depending on the configuration this means that the evaluator shall check the content of the audit data when the audit data is just filled to the maximum and then verifies that:

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.1.5).

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.1.5)

The TOE behaves as specified (for the option 'other action' in FAU_STG_EXT.1.5).

The evaluator performed actions to generate audit events until the local audit trail was full and confirmed that the oldest events were overwritten when new events were generated.

Test 5: For distributed TOEs, for the local storage according to FAU_STG_EXT.1.4, Test 1 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, Test 2 specified above shall be applied. The evaluator shall verify that the transfer of audit data to an external audit server is implemented.

This test is not applicable as the TOE is not a distributed TOE.

TD0886

An Application Note is appended to Test 6 for FAU_STG_EXT.1 in CPP_ND_V3.0E-SD as follows:

Note: The intent of the test is to ensure that the local audit TSF (as specified by FAU_STG_EXT.1.3) operates independently from the ability to transmit the generated audit data to an external audit server (as specified in FAU_STG_EXT.1.1). There are no specific requirements on the interruption of the connection between the TOE and the external audit server (as for FTP_ITC.1).

Test 6 [Conditional]: In case manual export or ability to view locally is selected in FAU_STG_EXT.1.6, during interruption the evaluator shall perform a TSF-mediated action and verify the event is recorded in the audit trail.

The evaluator verified the provides the ability to view the audit records locally.

2.1.4 FAU_STG_EXT.3 Action in Case of Possible Audit Data Loss

2.1.4.1 TSS Activities

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.

Section 6.1.2 of [ST] states that the TOE will generate a warning message if the storage space for audit records reaches 75% capacity. The warning is logged to the audit trail.

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 is supported only by one of the components. In particular, the evaluator shall 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 shall verify that the TSS makes clear any situations in which audit records might be 'invisibly lost'.

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

2.1.4.2 Guidance Activities

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.

[CCECG] Section “set log command details” states that the maximum amount of space available on RedSeal for all logs is 2GB. RedSeal generates a warning message if the storage space for audit records reaches 75% capacity. The warning is logged to the audit trail.

The description in the guidance documentation corresponds to the description in the TSS.

2.1.4.3 Test Activities

Test 1: The evaluator shall verify that a warning is issued by the TOE before the local storage space for audit data is full.

The evaluator generated audit records on the TOE until more than 90% of storage space was used. Audit records generated by the TOE were then examined, and it was shown that an alert was thrown once 75% of storage space was used up.

Test 2: 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.

This portion of the test activity is not applicable as the TOE is not a distributed TOE.

2.1.5 FAU_STG.1 Protected Audit Trail Storage

2.1.5.1 TSS Activities

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.

Section 6.1.2 of [ST] states that the logs comprising the audit trail are stored in the TOE's file system and protected from unauthorized modification and deletion by file system permissions. Log files are deleted when the oldest log is rotated out by the log rotation settings. The size must be in the range of 1,000 Bytes to 1,000MB (default is 50MB). When the current log file (for each of the logs defined above) reaches its rotation threshold, it is closed and a new current log file is created. If the configured maximum number of files for that log already exists, the oldest one is deleted. The maximum amount of space available on the TOE for all logs is 2GB.

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).

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

2.1.5.2 Guidance Activities

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.

N/A – no configuration is required for protection of audit data stored locally.

2.1.5.3 Test Activities

Test 1: The evaluator shall attempt to 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.

The only way to access the TOE's SSH CLI is with an administrative account. The interface cannot be used without authenticating first, and the administrative account is the only valid type. Thus, it is not possible for a non-administrator to access the audit trail through the CLI without an administrative account.

It is possible to access the TOE's Java Client and Web GUI interfaces with a non-administrative account. When doing that though a user is given a minimal interface or *System Settings* page that lacks the ability to modify or delete audit records.

Test 2: The evaluator shall access the audit trail as an authenticated Security Administrator and attempt to delete the audit records (if supported by the TOE, and to the extent described in the TSS). The evaluator shall verify that these attempts succeed. The evaluator shall verify that only the records authorized for deletion are deleted.

The evaluator connected to the TOE's Java interface and used the *Clear All Logs* functionality in *System Settings* to delete audit logs. After doing this the evaluator examined the TOE's storage space, and verified that the deletion of records increased the amount of storage space available.

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.

This portion of the test activity is not applicable as the TOE is not a distributed TOE.

2.2 Cryptographic Support (FCS)

Section 6.2.1 of [ST] identifies the TOE's CAVP certificates with respect to its cryptographic claims that require them. This has been reproduced below, with the relevant SFRs added as per Labgram #108.

As per Labgram #108, the full justification and evidence for why these certificates are sufficient to meet the evaluation activity requirements for the SFRs in question has been included in the Certificate Reporting section in the ETR.

The TOE includes OpenSSL 3.2.2-6, which provides the cryptographic algorithms to support SSH connections to the CLI.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Asymmetric Key Generation (FCS_CKM.1)		
RSA (2048 bits)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1	A7460 RSA KeyGen (FIPS 186-5)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2	A7460 ECDSA KeyGen/KeyVer (FIPS 186-5)
FFC Schemes using 'safe-prime' groups: Diffie-Hellman MODP Groups 14, 16, 18	NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and RFC 3526	CCTL tested against known-good implementation
Key establishment (FCS_CKM.2)		
Elliptic curve-based scheme (P-256, P-384, P-521 curves)	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"	A7460 KAS-ECC-SSC SP800-56Ar3
FFC Schemes using "safe-prime" groups: Diffie-Hellman MODP Groups 14, 16, 18	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and groups listed in RFC 3526	A7460 KAS-FFC-SSC SP800-56Ar3
Data encryption (FCS_COP.1/DataEncryption)		

Functions	Standards	Certificates
AES in CBC mode (128, 256 bits) AES in CTR mode (128, 256 bits)	ISO 18033-3 (AES) ISO 10116 (CBC and CTR mode)	A7460 AES-CBC A7460 AES-CTR
Digital signature generation and verification (FCS_COP.1/SigGen)		
RSA Digital Signature Algorithm (2048 bit modulus)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSAPKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A7460 RSA SigGen/SigVer (FIPS 186-5)
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A7460 SHA-1 A7460 SHA2-256 A7460 SHA2-384 A7460 SHA2-512
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-1 (key size 512 bits, digest size 160 bits) HMAC-SHA-256 (key size 512 bits, digest size 256 bits) HMAC-SHA-512 (key size 1024 bits, digest size 512 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"	A7460 HMAC-SHA-1 A7460 HMAC-SHA2-256 A7460 HMAC-SHA2-512
Deterministic random bit generation (FCS_RBG_EXT.1)		
CTR_DRBG (AES-256)	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions"	A7460 Counter DRBG

The TOE through an integrated Dell BSAFE SSL-J v7.3.1 implementation of cryptographic algorithms support secure communication: between the TOE's browser-based Web Beta client and the remote administrative users (HTTPS); between the TOE and the Java client (TLS); between the TOE and the audit server (TLS); and between the TOE and legacy web clients (HTTPS).

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Asymmetric Key Generation (FCS_CKM.1)		
RSA (2048, 3072, 4096 bits)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1	A7459 RSA KeyGen (FIPS 186-5)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2	A7459 ECDSA KeyGen/ KeyVer (FIPS 186-5)
FFC Schemes using 'safe-prime' groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and groups listed in RFC 7919	CCTL tested against known-good implementation

Functions	Standards	Certificates
Key establishment (FCS_CKM.2)		
Elliptic curve-based scheme (P-256, P-384, P-521 curves)	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"	A7459 KAS-ECC-SSC SP800-56Ar3
FFC Schemes using 'safe-prime' groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192	NIST Special Publication 800-56A Revision 2, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and groups listed in RFC 7919	A7459 KAS-FFC-SSC SP800-56Ar3
Data encryption (FCS_COP.1/DataEncryption)		
AES in CBC mode (128, 256 bits) AES in GCM mode (128, 256 bits) AES in CTR mode (128, 256 bits)	ISO 18033-3 (AES) ISO 10116 (CBC and CTR mode) ISO 19772 (GCM mode)	A7459 AES-CBC A7459 AES-GCM A7459 AES-CTR
Digital signature generation and verification (FCS_COP.1/SigGen)		
RSA Digital Signature Algorithm (2048, 3072, 4096 bit modulus)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.4, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSAPKCS2v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A7459 RSA SigGen/SigVer (FIPS 186-5)
ECDSA Digital Signature Algorithm (NIST curves P-256, P-384, P-521)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"; ISO/IEC 14888-3, Section 6.4	A7459 ECDSA SigGen/SigVer (FIPS 186-5)
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A7459 SHA-1 A7459 SHA2-256 A7459 SHA2-384 A7459 SHA2-512
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-1 (key size 512 bits, digest size 160 bits) HMAC-SHA-256 (key size 512 bits, digest size 256 bits) HMAC-SHA-384 (key size 1024 bits, digest size 384 bits) HMAC-SHA-512 (key size 1024 bits, digest size 512 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"	A7459 HMAC-SHA-1 A7459 HMAC-SHA2-256 A7459 HMAC-SHA2-384 A7459 HMAC-SHA2-512

Functions	Standards	Certificates
Deterministic random bit generation (FCS_RBG_EXT.1)		
Hash_DRBG(SHA-256)	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions"	A7459 Hash DRBG

2.2.1 FCS_CKM.1 Cryptographic Key Generation

2.2.1.1 TSS Activities

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

Section 6.2.3 of [ST] states that the TOE generates RSA asymmetric key pairs with cryptographic key sizes (modulus) of 2048, 3072 or 4096 bits, in accordance with Appendix A.1 of FIPS PUB 186-5, "Digital Signature Standard (DSS)". The RSA keys are used in support of SSH public key authentication and TLS server authentication. All three key sizes are used by Dell BSAFE SSL-J in support of TLS server authentication; 2048-bit key sizes are used by OpenSSL in support of SSH public key authentication.

The TOE generates ECC asymmetric key pairs over NIST curves P-256, P-384, and P-521, in accordance with Appendix A.2 of FIPS PUB 186-5, "Digital Signature Standard (DSS)". The ECDSA keys are used in support of SSH and TLS key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using Diffie-Hellman MODP groups 14, 16, 18 in accordance with RFC 3526, Section 3. These keys are used in support of SSH key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using FFDHE groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192 in accordance with RFC7919. These keys are used in support of TLS key exchange.

2.2.1.2 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

As also described in Section 6, it is possible to use the `enable cipher-suites` and `disable cipher-suites` commands to manually enable and disable specific ciphers on the TOE's TLS server interface.

2.2.1.3 Test Activities

Modified by TD0921

Key Generation for FIPS PUB 186-4 or FIPS PUB 186-5 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. This test must be repeated for each supported RSA modulo and generation method.

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

a. Random Provable Primes (p and q shall be provable primes):
b. Random Probable primes (p and q shall be probable primes)
c. Provable Primes with Conditions (p1, p2, q1, q2, p and q shall all be provable primes):
d. Provable/probable primes with Conditions (p1, p2, q1, and q2 shall be provable primes and p and q shall be probable primes)
e. Probable primes with Conditions (p1, p2, q1, q2, p and q shall all be probable primes)

The Random Provable primes, and all the Primes with Conditions can be tested in the same manner because each of these begin with a starting random number and calculate the p and q values from this value. The test instructs the TSF to generate intermediate values and the p, q, n, and d values. The evaluator then validates the correctness of the values generated by the TSF.

To test the key generation method for the Random Provable primes method or 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. If the TSF provides the input to the key generation function, such input must be recorded and verified. For each RSA key length (modulo) claimed supported, the evaluator shall have the TSF generate 25 key pairs. The evaluator shall verify the correctness of the TSF's implementation by comparing Key Pair values generated by the TSF with those generated using the same set of input values using from a known good implementation.

The Random Probable primes must be tested in a different way because this test generates different random numbers, not related to each other, until the number satisfies the "probably prime" requirements. This validation method requires two tests for Random Probable primes. These include the Known Answer Test and the Miller-Rabin probabilistic primality test.

To test the key generation method for the Random Probable primes, the known answer test must be used. The evaluator shall compare the results of a known good implementation with the TSF results for a set (of a corresponding size depending on modulus) of supplied values containing private prime factor p, private prime factor q and confirm all public keys, e, match.

Then the evaluator shall use the TSF to generate prime p, q pairs for each modulus size and perform the Miller-Rabin tests.

Key Generation for Elliptic Curve Cryptography (ECC)

Key Generation for ECDSA Schemes

Key pairs for the ECDSA consist of pairs (d, Q), where the private key, d, is an integer, and the public key, Q, is an elliptic curve point.

The evaluator shall verify the implementation of ECC Key Generation by the TOE using the following components tests:

- *ECC Key Pair Generation*
- *ECC Public Key Validation (PKV)*

ECC Key Pair Generation Test

There are four methods by which these pairs may be generated:

- FIPS 186-4 Section B.4.1 Key Pair Generation Using Extra Random Bits

  Using this method, 64 more bits are requested from the RBG than are needed for d so that bias produced by the mod function is negligible.
- FIPS 186-4 Section B.4.2 Key Pair Generation by Testing Candidates

  Using this method, a random number is obtained and tested to determine that it will produce a value of d in the correct range. If d is out-of-range, another random number is obtained (i.e., the process is iterated until an acceptable value of d is obtained.
- FIPS 186-5 Section A.2.1 ECDSA Key Pair Generation using Extra Random Bits
- Using this method, more bits are requested from the DRBG than are needed for d so that the bias produced by the mod function is negligible.
- FIPS 186-5 Section A2.2 ECDSA Key Pair Generation by Rejection Sampling
- Using this method, a random number is obtained and tested to determine that it will produce a value of d in the correct range. If d is out of range, an ERROR is returned.

The evaluator shall test the ECC Key Pair Generation by having the TSF produce 10 key pairs (d, Q) for each implemented key generation method using each supported curve (i.e., P-256, P-384, and P-521). The steps are the same for each key generation method and curve. The private key, d, shall be generated using the output of an approved DRBG converted to an integer via modular reduction or the discard method. The known private key is then used by the TSF to compute the public key, Q'. To evaluator then validates the correctness by comparing the value Q' computed by the TSF to the public key, Q generated by a known good implementation.

CC Public Key Verification (PKV) Test

The evaluator shall generate 12 key pairs (d, Q) for each selected curve, with 6 valid public keys, Q, using a known good implementation and 6 modified, Q', and determine whether the TSF can accurately detect these modifications. Q' should be otherwise valid but include at least one of the following errors: a) point X or Y not on the curve, b) point X or Y is too large

for the field for the given curve. The evaluator encouraged to make sure that the modification does not inadvertently result in another valid public key (e.g., modifying Y and accidentally hitting a different point on the curve).

[ST] Section 6.2.1 identifies the following asymmetric key generation implementations

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Asymmetric Key Generation (FCS_CKM.1)		
RSA (2048 bits)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1	A7460 RSA KeyGen (FIPS 186-5)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2	A7460 ECDSA KeyGen/KeyVer (FIPS 186-5)
FFC Schemes using 'safe-prime' groups: Diffie-Hellman MODP Groups 14, 16, and 18	NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and RFC 3526	CCTL tested against known-good implementation

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Asymmetric Key Generation (FCS_CKM.1)		
RSA (2048, 3072, 4096 bits)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1	A7459 RSA KeyGen (FIPS 186-5)
ECDSA (P-256, P-384, P-521 curves)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2	A7459 ECDSA KeyGen/ KeyVer (FIPS 186-5)
FFC Schemes using 'safe-prime' groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and groups listed in RFC 7919	CCTL tested against known-good implementation

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

2.2.2 FCS_CKM.2 Cryptographic Key Establishment

2.2.2.1 TSS Activities

The evaluator shall ensure that the supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1. If the ST specifies more than one scheme,

the evaluator shall examine the TSS to verify that it identifies the usage for each scheme. It is sufficient to provide the scheme, SFR, and service in the TSS.

Section 6.2.3 of [ST] states that the TOE generates RSA asymmetric key pairs with cryptographic key sizes (modulus) of 2048 bits, 3072 bits, and 4096 bits, in accordance with Appendix A.1 of FIPS PUB 186-5, “Digital Signature Standard (DSS)”. The RSA keys are used in support of SSH public key authentication and TLS server authentication.

The TOE generates ECC asymmetric key pairs over NIST curves P-256, P-384, and P-521, in accordance with Appendix A.2 of FIPS PUB 186-5, “Digital Signature Standard (DSS)”. The ECDSA keys are used in support of SSH and TLS key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using Diffie-Hellman MODP groups 14, 16, 18 in accordance with RFC 3526, Section 3. These keys are used in support of SSH key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using FFDHE groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192 in accordance with RFC7919. These keys are used in support of TLS key exchange.

Key Establishment Scheme Usage by TOE

Scheme	**SFR**	**Service**
ECDH	FCS_TLSC_EXT.1	Audit server
ECDH	FCS_TLSS_EXT.1	Administration
DH (Groups ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192)	FCS_TLSC_EXT.1	Audit server
DH (Groups ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192)	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHS_EXT.1	Administration
DH (MODP Group 14, 16, 18)	FCS_SSHS_EXT.1	Administration

The supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1.

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 as shown in the table below. The information provided in this example 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.

Scheme	**SFR**	**Service**
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_IPSEC_EXT.1	Authentication

Section 6.2.3 of [ST] states that the TOE generates RSA asymmetric key pairs with cryptographic key sizes (modulus) of 2048 bits, in accordance with Appendix A.1 of FIPS PUB 186-5, "Digital Signature Standard (DSS)". The RSA keys are used in support of SSH public key authentication and TLS server authentication.

The TOE generates ECC asymmetric key pairs over NIST curves P-256, P-384, and P-521, in accordance with Appendix A.2 of FIPS PUB 186-5, "Digital Signature Standard (DSS)". The ECDSA keys are used in support of SSH and TLS key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using Diffie-Hellman MODP groups 14, 16, 18 in accordance with RFC 3526, Section 3. These keys are used in support of SSH key exchange.

The TOE generates FFC asymmetric safe-prime key pairs using FFDHE groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192 in accordance with RFC7919. These keys are used in support of TLS key exchange.

The supported key establishment schemes correspond to the key generation schemes identified in FCS_CKM.1.1.

2.2.2.2 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

2.2.2.3 Test Activities

Key Establishment Schemes The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the selected key establishment scheme(s). **ECC and FIPS 186-type SP800-56A Key Establishment Schemes** Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] identifies the CAVP certifications verifying SP 800-56A key establishment schemes, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Key establishment (FCS_CKM.2)		
Elliptic curve-based scheme (P-256, P-384, P-521 curves)	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"	A7460 KAS-ECC-SSC SP800-56Ar3

Functions	Standards	Certificates
FFC Schemes using “safe-prime” groups: Diffie-Hellman MODP Groups 14, 16, 18	NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and groups listed in RFC 3526	A7460 KAS-ECC-SSC SP800-56Ar3

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Key establishment (FCS_CKM.2)		
Elliptic curve-based scheme (P-256, P-384, P-521 curves)	NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”	A7459 KAS-ECC-SSC SP800-56Ar3
FFC Schemes using ‘safe-prime’ groups: ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192	NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and groups listed in RFC 7919	A7459 KAS-ECC-SSC SP800-56Ar3

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.

Refer to test activities for FTP_TRP.1/Admin and FTP_ITC.1.

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.

For each of the claimed safe-prime groups the evaluator used a known good implementation (e.g. OpenSSL s_client or OpenSSH ssh) to establish a connection with the TOE using each of the specified groups and verified that the connection completed successfully using each claimed group.

2.2.3 FCS_CKM.4 Cryptographic Key Destruction

2.2.3.1 TSS Activities

The evaluator shall examine 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 shall confirm 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[2]). In particular, if a TOE claims not to store plaintext keys in non-volatile memory then the evaluator shall check that this is consistent with the operation of the TOE.

Section 6.2.4 of [ST] lists all keys/CSPs used by the TOE by their function.

Key/CSP	Origin, Use and Storage
RSA private key	Generated by TOE. Used to authenticate the TOE in TLS and SSH sessions. Stored on disk in PKCS #12 keystore.
EC DH private key	Generated by TOE. Used in TLS and SSH key exchange. Stored in RAM.
DH private key	Generated by TOE. Used in TLS and SSH key exchange. Stored in RAM.
AES keys used for secure communication	Generated by TOE. Used to encrypt/decrypt data transmitted/received in TLS and SSH sessions. Stored in RAM.
AES key used to encrypt PKCS #12 keystore	Derived by TOE from password using PBKDF2. Exists ephemerally in RAM.
HMAC keys	Generated by TOE. Used to verify integrity of packets in TLS and SSH sessions. Stored in RAM.
NTP keys	Specified by administrator. Used to authenticate communications received from configured NTP servers. Stored in plaintext in TOE file system.
DRBG parameters (seed, entropy input)	Generated by TOE. Used to instantiate DRBG. Stored in RAM.

The keys in the above table that are stored in RAM are ephemeral keys that are destroyed by the cryptomodule manipulating the key. The OpenSSL cryptomodule destroys keys directly by overwriting them once with zeroes. The BSAFE cryptomodule provides the <object>.clearSensitiveData() method, which destroys the reference to the key and immediately issues a request for garbage collection to destroy keys it holds in RAM.

The keys in the above table stored in plaintext in the TOE's file system are long-term persistent keys that are required for correct continuing operation of the product. They are destroyed when no longer required in one of two ways:

- The key is overwritten by a new value for the key, e.g., when the **cliadmin** changes the value of an NTP key using the `set ntp authentication symmetric add-key` CLI command
- The **cliadmin** executes the `reset all` CLI command, which resets the appliance to its factory defaults. The reset all command instructs a part of the TSF to destroy the abstraction that represents the key, using the Linux `rm –rf <filename>` command.

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).

Section 6.2.4 of [ST] states that the keys stored in plaintext in the TOE's file system are long-term persistent keys that are required for correct continuing operation of the product. They are destroyed when no longer required in one of two ways:

- The key is overwritten by a new value for the key, e.g., when the **cliadmin** changes the value of an NTP key using the `set ntp authentication symmetric add-key` CLI command
- The **cliadmin** executes the `reset all` CLI command, which resets the appliance to its factory defaults. The reset all command instructs a part of the TSF to destroy the abstraction that represents the key, using the Linux `rm –rf <filename>` command.

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.

Section 6.2.4 of [ST] states that the keys that are stored in RAM are ephemeral keys that are destroyed by the cryptomodule manipulating the key. The OpenSSL cryptomodule destroys keys directly by overwriting them once with zeroes. The BSAFE cryptomodule provides the `<object>.clearSensitiveData()` method, which destroys the reference to the key and immediately issues a request for garbage collection to destroy keys it holds in RAM.

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.

Section 6.2.4 of [ST] identifies that the only key stored in a non-plaintext form is the RSA private key, which is protected by AES with the key derived from PBKDF2 and is itself stored in a volatile memory.

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.

The evaluators examined [ST] and observed that no exceptions to the behavior described by FCS_CKM.4.1 have been identified, so it can be assumed that this behavior is followed in all cases.

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.

Section 6.2.4 of [ST] states that the OpenSSL cryptomodule destroys keys directly by overwriting them once with zeroes.

2.2.3.2 Guidance Activities

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

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

[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).]

[CCECG] Section "Key Destruction" states that RedSeal uses some secret keys, private keys, and other critical security parameters. When these keys are no longer required, they are destroyed.

There are no circumstances or configuration where key destruction is delayed. Key destruction is never delayed at the physical layer.

2.2.3.3 Test Activities

None

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

2.2.4.1 TSS Activities

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

[ST] section 6.2.1 identifies that the TOE supports AES with key sizes 128 bits and 256 bits across CBC, CTR, and GCM modes.

2.2.4.2 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

2.2.4.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2 of [ST] identifies the CAVP certifications verifying AES encryption and decryption, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Data encryption (FCS_COP.1/DataEncryption)		
AES in CBC mode (128, 256 bits) AES in CTR mode (128, 256 bits)	ISO 18033-3 (AES) ISO 10116 (CBC and CTR mode)	A7460 AES-CBC A7460 AES-CTR

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Data encryption (FCS_COP.1/DataEncryption)		

Functions	Standards	Certificates
AES in CBC mode (128, 256 bits) AES in GCM mode (128, 256 bits) AES in CTR mode (128, 256 bits)	ISO 18033-3 (AES) ISO 10116 (CBC and CTR mode) ISO 19772 (GCM mode)	A7459 AES-CBC A7459 AES-GCM A7459 AES-CTR

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

2.2.5.1 TSS Activities

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

[ST] section 6.2 states that RSA (2048/3072/4096 bit modulus) and ECDSA (P-256/P-384/P-521 curves) are supported for digital signatures.

2.2.5.2 Guidance Activities

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

[CCECG] Section “Evaluated configuration” states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

This process will restrict the TLS version and cipher suites to the approved ones claimed in the ST, including the allowed cryptographic algorithms and key sizes used by the TOE for signature services.

2.2.5.3 Test Activities

Modified by TD0921

ECDSA Signature Algorithm Tests

The evaluator shall verify the implementation of ECDSA Signature generation by the TOE using the Signature Generation Test. This test validates the TSF generation of the (r, s) pair that represent the digital signature. The digital signature shall be verified using the same domain parameters and hash function that were used during signature generation. An approved hash function or an XOF shall be used for this purpose.

Signature Generation Test

The purpose of this test is to verify the ability of the TSF to produce correct signatures.

To test signature generation, the evaluator supplies 10 pseudorandom messages to the TSF and a key pair, (d, Q), generated by a known good implementation. Exercising each applicable

curve (i.e., P-256, P 384, or P-521) and hash algorithm or extendable-output function combination, the TSF generates signatures for each supplied message and returns the corresponding signatures. Using a known-good implementation, the evaluator validates the signatures by using the associated public key, Q, to verify the signature.

Signature Verification Test

The purpose of this test is to verify the ability of the TSF to accept valid signatures and reject invalid signatures.

For each curve/hash algorithm or extendable-output function combination supported by the TSF, the evaluator shall use a known good implementation to generate a key pair, (d, Q), and use known good implementation with the private key, d, to sign 15 pseudorandom messages of 1024 bits. The evaluator shall alter some of the messages or signatures so that signature verification should fail.

To test signature verification, the evaluator supplies the public key, Q, and 15 pseudorandom messages, including altered messaged, to the TSF for verification of signatures. The evaluator shall then verify that the TSF validates correct signatures on the original messages and flags or rejects the altered messages.

RSA Signature Algorithm Tests

Signature Generation Test

The evaluator shall verify the implementation of RSA Signature generation by the TOE using the Signature Generation Test. This test verifies the ability of the TSF to produce correct signatures.

There are 2 different RSA Signature algorithms that can be implemented. These include:

a. RSASSA-PKCS1-v1.5

b. RSASSA-PSS

To test signature generation, the evaluator generates or obtains 10 messages for each modulus size/hash or extendable-output function combination supported by the TOE. Using a key generated by a known good implementation, the TSF generates and returns the corresponding signatures to a known good implementation that validates the signatures by using the associated public key to verify the signature.

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

The evaluator shall verify the implementation of RSA Signature verification by the TOE using the Signature Verification Test. This test verifies the ability of the TSF to recognize valid and invalid signatures.

There are 2 different RSA Signature algorithms that can be implemented. These include:

RSASSA-PKCS1-v1.5

RSASSA-PSS

For each modulus size/hash or extendable-output function combination supported by the TOE, the evaluator shall use a known good implementation to generate a modulus and three associated key pairs, (d, e). Each private key d is used to sign six pseudorandom messages each of 1024 bits. Some of the public keys, e, messages, IR format, or signatures must be altered so that signature verification should fail. The modifications must cover distinct “key modified”, “message modified”, “signature modified”, “IR moved”, and “trailer moved” tests. The modulus, hash or extendable-output algorithm, public key e values, messages, and signatures are forwarded to the TSF. The TSF then attempts to verify the signatures and returns the results. The evaluator then compares the received results with the stored results from a known good implementation.

The evaluator shall verify that the TSF validates correct signatures on the original messages and flags or rejects the altered messages.

Section 6.2.1 of [ST] identifies the CAVP certifications verifying digital signature services, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Digital signature generation and verification (FCS_COP.1/SigGen)		
RSA Digital Signature Algorithm (2048 bit modulus)	FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Section 5.4, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSAPKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A7460 RSA SigGen/SigVer (FIPS 186-5)

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Digital signature generation and verification (FCS_COP.1/SigGen)		
RSA Digital Signature Algorithm (2048, 3072, 4096 bit modulus)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.4, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSAPKCS2v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A7459 RSA SigGen/SigVer (FIPS 186-5)
ECDSA Digital Signature Algorithm (NIST curves P-256, P-384, P-521)	FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"; ISO/IEC 14888-3, Section 6.4	A7459 ECDSA SigGen/SigVer (FIPS 186-5)

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

2.2.6.1 TSS Activities

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

Sections 6.2.1 [ST] indicate the hash function is associated with following cryptographic services:

"The TOE uses the SHA hash algorithms as follows:

- as part of the HMAC algorithms that provide data integrity for SSH and TLS
- as part of RSA digital signature generation and verification
- as part of the conditioning used to protect stored passwords (refer to Section **Error! Reference source not found.**)
- for NTP authentication (SHA1)."

2.2.6.2 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

This process will configure cryptographic parameters to ensure that only the required hash algorithms are used for trusted channel communications.

2.2.6.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2.1 of [ST] identifies the CAVP certifications verifying cryptographic hashing, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A7460 SHA-1 A7460 SHA2-256 A7460 SHA2-384 A7460 SHA2-512

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A7459 SHA-1 A7459 SHA2-256 A7459 SHA2-384 A7459 SHA2-512

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

2.2.7.1 TSS Activities

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

Table 6.2.1 of [ST] includes a list of the HMAC functions that lists the key size, block size, digest size, and output MAC length).

Hash Function	Key Length	Block Size	Output MAC Length
SHA-1	512 bits	512 bits	160 bits
SHA-256	512 bits	512 bits	256 bits
SHA-384	1024 bits	1024 bits	384 bits
SHA-512	1024 bits	1024 bits	512 bits

2.2.7.2 Guidance Activities

The evaluator shall verify that the AGD guidance instructs the administrator how to configure the TOE to use the values used by the HMAC function: key length, hash function used, block

size, and output MAC length used defined in the Security Target supported by the TOE for keyed hash function.

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

This process will configure cryptographic parameters to ensure that only the required keyed-hash algorithms are used for trusted channel communications.

2.2.7.3 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2.1 of [ST] identifies the CAVP certifications verifying cryptographic keyed hashing, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-1 (key size 512 bits, digest size 160 bits) HMAC-SHA-256 (key size 512 bits, digest size 256 bits) HMAC-SHA-512 (key size 1024 bits, digest size 512 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"	A7460 HMAC-SHA-1 A7460 HMAC-SHA2-256 A7460 HMAC-SHA2-512

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-1 (key size 512 bits, digest size 160 bits) HMAC-SHA-256 (key size 512 bits, digest size 256 bits) HMAC-SHA-384 (key size 1024 bits, digest size 384 bits) HMAC-SHA-512 (key size 1024 bits, digest size 512 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"	A7459 HMAC-SHA-1 A7459 HMAC-SHA2-256 A7459 HMAC-SHA2-384 A7459 HMAC-SHA2-512

2.2.8 FCS_HTTPS_EXT.1 HTTPS Protocol

2.2.8.1 TSS Activities

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

[ST] Section 6.2.5.3 states that the TOE's HTTPS protocol complies with RFC 2818 and is implemented using TLS 1.2 (RFC 5246). HTTPS is used for the browser-based web connections, therefore, only the server requirements in RFC 2818 are applicable. RFC 2818 (section 2.1: complies as specified, section 2.2: complies as specified, section 2.2.1: not applicable, section 2.2.2: complies as specified, section 2.3: default port is 443, section 2.4: Use 'https' as specified, section 3.1: not applicable, and section 3.2: client identity checking is not performed/not applicable).The TOE uses HTTPS to protect communications between itself and remote users accessing the browser-based web interfaces. The TOE implements the server side of the HTTPS protocol according to RFC 2818 by using a TLS session to secure the HTTP connection.

2.2.8.2 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

This process will configure the DRBG to use the algorithm claimed in [ST] and enforce TLS and HTTPS certificate validation.

2.2.8.3 Test Activities

This test is now performed as part of FIA_X509_EXT.1/Rev testing. Tests are performed in conjunction with the TLS evaluation activities. If the TOE is an HTTPS client or an HTTPS server utilizing X.509 client authentication, then the certificate validity shall be tested in accordance with testing performed for FIA_X509_EXT.1.

2.2.9 FCS_NTP_EXT.1 NTP Protocol

2.2.9.1 TSS Activities

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.

Section 6.2.5.5 of [ST] states that the TOE can synchronize its system clock with an NTP server. The TOE supports NTP v4 as defined in RFC 5905 and uses SHA-1 as its means for authenticating the NTP timestamps it receives from configured NTP servers.

FCS_NTP_EXT.1.1

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

Section 6.2.5.5 of [ST] states that the TOE can synchronize its system clock with an NTP server. The TOE supports NTP v4 as defined in RFC 5905 and uses SHA-1 as its means for authenticating the NTP timestamps it receives from configured NTP servers.

FCS_NTP_EXT.1.2, FCS_NTP_EXT.1.3, and FCS_NTP_EXT.1.4

None

2.2.9.2 Guidance Activities

FCS_NTP_EXT.1.1

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.

[CCECG] Section "Set NTP servers" states that the TOE only supports NTP v4 (RFC 5905) and this is not configurable.

The `set ntp` command is used to configure up to 5 NTP servers. Each server is identified using a comma-separated list within the command.

The `set ntp authentication symmetric add-key` sets up the SHA-1 authentication key.

The `set ntp authentication symmetric configure-key trusted <KEY_ID> <NTP_SERVER>` is used to configure the trusted key for NTP servers using the `set ntp authentication` command.

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).

[CCECG] Section “Set NTP servers” states that the TOE only supports NTP v4 (RFC 5905) and this is not configurable.

The `set ntp` command is used to configure up to 5 NTP servers. Each server is identified using a comma-separated list within the command.

The `set ntp authentication symmetric add-key` sets up the SHA-1 authentication key.

The `set ntp authentication symmetric configure-key trusted <KEY_ID> <NTP_SERVER>` is used to configure the trusted key for NTP servers using the `set ntp authentication` command.

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.

[CCECG] Section “Set NTP servers” states that the TOE will not accept broadcast or multicast NTP packets by default.

FCS_NTP_EXT.1.4
None.

2.2.9.3 Test Activities

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.

As part of NTP testing the evaluator had the TOE receive time updates from an NTP server. NTP traffic was captured during that test, and an examination of NTP packets showed the use of NTPv4 as claimed in the [ST].

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.

During NTP testing the evaluator successfully configured the TOE with a SHA-1 key.

FCS_NTP_EXT.1.2

[conditional] If the message digest algorithm is claimed in element 1.2, the evaluator shall 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.

After performing the above test the evaluator altered the SHA-1 key that was used for it. After having the key modified the TOE would no longer accept time updates from the configured NTP server.

FCS_NTP_EXT.1.2

The evaluator shall use a packet sniffer to capture the network traffic between the TOE and the NTP server. The evaluator shall use 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.

As described above the evaluator had the TOE receive time updates from an NTP server. Wire captures showed that NTPv4 with SHA-1 was used, and audit records showed the TOE accepting time updates from the server.

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.

The evaluator demonstrated that the TOE would not accept time updates from broadcast or multicast NTP server.

FCS_NTP_EXT.1.4

The evaluator shall perform the following tests:

a. 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 multisource update of the time information is appropriate and consistentwith the behaviour prescribed by the RFC 1305 for NTPv3 and RFC 5905 for NTPv4.

b. 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 response 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. 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.

The evaluator configured the TOE for three separate NTP servers. The TOE was able to successfully receive time updates.

The evaluator configured an NTP test script to send unsolicited NTP time updates. The TOE did not accept time changes from those unsolicited messages.

2.2.10 FCS_RBG_EXT.1 Cryptographic Operation (Random Bit Generation)

2.2.10.1 Evaluation Activity

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

The vendor produced a proprietary Entropy Analysis Report (EAR) that the evaluators determined was suitable to meet the requirements specified in Appendix D of [NDcPP].

2.2.10.2 TSS Activities

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

Section 6.2.2 of [ST] states that the TOE instantiates the SHA-256 Hash DRBG provided by its BSAFE module to generate random bits for use with other BSAFE algorithms and the AES-256 Counter DRBG provided by OpenSSL to generate random bits for use with other OpenSSL algorithms.

The TOE seeds these DRBGs with 256 bits minimum assumed entropy obtained from the hardware-based Intel RDRAND function. The TOE uses Java Native Instructions (JNI) to interface with the native C code used to interface with RDRAND to obtain seed data for the BSAFE cryptographic module.

2.2.10.3 Guidance Activities

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

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

This process will instantiate a Hash DRGB and a Counter DRBG (AES) to generate random bits; these settings are not configurable.

2.2.10.4 Test Activities

Performed in accordance with NIAP Policy Letter #5.

Section 6.2.1 of [ST] identifies the CAVP certification verifying deterministic random bit generation, as follows.

Cryptographic Functions Implemented by OpenSSL

Functions	Standards	Certificates
Deterministic random bit generation (FCS_RBG_EXT.1)		
CTR_DRBG (AES-256)	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions"	A7460 Counter DRBG

Cryptographic Functions Implemented by RedSeal Server

Functions	Standards	Certificates
Deterministic random bit generation (FCS_RBG_EXT.1)		
Hash_DRBG(SHA-256)	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions"	A7459 Hash DRBG

2.2.11 FCS_SSH_EXT.1 SSH Protocol

2.2.11.1 TSS Evaluation Activity

FCS_SSH_EXT.1.1
The evaluator shall ensure that the selections indicated in the ST are consistent with selections in this and subsequent components. Otherwise, this SFR is evaluated by activities for other SFRs.

[ST] Section 6.2.5.4 states that the TOE acts as an SSH server when supporting inbound remote administration via the CLI. The TOE's implementation of SSH-2 complies with RFCs 4251, 4252, 4253, 4254, 4344, 5656, 6668, 8268, 8332, and 8731.

The SFR and TSS are consistent with the TOE being identified as a SSHv2 server.

FCS_SSH_EXT.1.2
The evaluator shall check to ensure that the authentication methods listed in the TSS are identical to those listed in this SFR component; and, ensure if password-based authentication methods have been selected in the ST then these are also described; and, ensure that if keyboard-interactive is selected, it describes the multifactor authentication mechanisms provided by the TOE.

[ST] Section 6.2.5.4 states that the TOE supports both public key and password-based authentication as described in RFC 4252.

The public key authentication algorithms are identified as:

- Public key authentication algorithm
    - for authenticating users
        - ssh-rsa

The user authentication algorithm is consistent with what is defined in FCS_SSH_EXT.1.2.

FCS_SSH_EXT.1.3
The evaluator shall check that the TSS describes how “large packets” are detected and handled.

[ST] Section 6.2.5.4 states that the TOE ensures that packets greater than 1 gigabyte in an SSH transport connection are dropped—i.e., such a packet is not processed further when this size limit is reached and the buffer containing the packet is freed.

FCS_SSH_EXT.1.4
The evaluator will check the description of the implementation of SSH in the TSS to ensure the encryption algorithms supported are specified. The evaluator will check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

[ST] Section 6.2.5.4 states that the TOE supports the AES encryption/decryption algorithm (in CBC or CTR mode) with key sizes of 128 or 256 bits. The encryption algorithms specified in the TSS are identical to those listed in the SFR.

FCS_SSH_EXT.1.5
The evaluator will check the description of the implementation of SSH in the TSS to ensure the hashing algorithms supported are specified. The evaluator will check the TSS to ensure that the hashing algorithms specified are identical to those listed for this component.

[ST] Section 6.2.5.4 states that the TOE supports HMAC-SHA-256 and HMAC-SHA-512 hashing algorithms for integrity. The hashing algorithms identified in the TSS are identical to those identified in the SFR component.

FCS_SSH_EXT.1.6

The evaluator will check the description of the implementation of SSH in the TSS to ensure the shared secret establishment algorithms supported are specified. The evaluator will check the TSS to ensure that the shared secret establishment algorithms specified are identical to those listed for this component.

[ST] Section 6.2.5.4 states that the TOE implements the following key exchange algorithms for the server:

- diffie-hellman-group14-sha256 (RFC 8268),
- diffie-hellman-group16-sha512 (RFC 8268),
- diffie-hellman-group18-sha512 (RFC 8268)
- ecdh-sha2-nistp256 (RFC 5656),
- ecdh-sha2-nistp384 (RFC 5656),
- ecdh-sha2-nistp521 (RFC 5656)

The shared secret establishment algorithms specified in the TSS are identical to those listed in the SFR.

FCS_SSH_EXT.1.7

The evaluator will check the description of the implementation of SSH in the TSS to ensure the KDFs supported are specified. The evaluator will check the TSS to ensure that the KDFs specified are identical to those listed for this component.

[ST] Section 6.2.5.4 states that KDF as defined in RFC 4253 (Section 7.2) and RFC 5656 (Section 4) are to derive session keys from a shared secret for the server.

FCS_SSH_EXT.1.8

The evaluator shall check the TSS to ensure that if the TOE enforces connection rekey or termination limits lower than the maximum values that these lower limits are identified.

In cases where hardware limitation will prevent reaching data transfer threshold in less than one hour, the evaluator shall check the TSS to ensure it contains:

a. An argument describing this hardware-based limitation and
b. Identification of the hardware components that form the basis of such argument.

For example, if specific Ethernet Controller or Wi-Fi radio chip is the root cause of such limitation, these subsystems shall be identified.

[ST] Section 6.2.5.4 states that the TOE ensures that within SSH connections the same session keys are used for a threshold of no longer than one hour, and each encryption key is used to protect no more than one gigabyte of data (transmitted or received). The TOE initiates a rekey when any of the thresholds is reached, whichever is hit first.

2.2.11.2 Guidance Evaluation Activity

FCS_SSH_EXT.1.1
There are no guidance evaluation activities for this component. This SFR is evaluated by activities for other SFRs.

There are no guidance evaluation activities for this component.

FCS_SSH_EXT.1.2
The evaluator shall check the guidance documentation to ensure the configuration options, if any, for authentication mechanisms provided by the TOE are described.

[CCECG] Section "Access to TOE" states:

Remote connection to the CLI via SSH v2, using username and password or username and public key.

[CCECG] Section "Logging in to the command line interface" states: "RedSeal provides a Command Line Interface (CLI) for management and administration. The CLI is accessible remotely via SSHv2."

Connect to RedSeal using a remote connection. For remote access, use SSH client software (for example, PuTTY) to connect to the RedSeal server's IP address on port 22, and use the SSH connection type.

[CCECG] Section "Unlocking administrator accounts"

For any password-based accounts that are locked out, the cliadmin can use the enable user command to unlock the locked user account in SSH, or navigating to EDIT -> System Settings -> Users -> Right-click user and click Edit on the Java client/Web GUI. The user account will also unlock automatically after the configured lockout period has elapsed.

FCS_SSH_EXT.1.3
None listed.

There are no guidance evaluation activities for this component.

FCS_SSH_EXT.1.4
The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

Section 6 states that the Encryption algorithms—aes128-cbc, aes256-cbc, aes128-ctr, and aes256-ctr are enabled by the (`enable common criteria`) command.

FCS_SSH_EXT.1.5
The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

[CCECG] Section “Evaluated configuration” states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

The command sets RedSeal's data integrity MAC algorithms to use hmac-sha2-256, and hmac-sha2-512 for the SSH connections.

FCS_SSH_EXT.1.6
The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

[CCECG] Section “Evaluated configuration” states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

The command sets RedSeal's key exchange algorithms.

FCS_SSH_EXT.1.7
None listed.

There are no guidance evaluation activities for this component.

FCS_SSH_EXT.1.8
The evaluator shall check the guidance documentation to ensure that if the connection rekey or termination limits are configurable, it contains instructions to the administrator on how to configure the relevant connection rekey or termination limits for the TOE.

[CCECG] Section “Evaluated configuration” states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (enable common criteria). No other configuration is necessary.

The SSH re-key thresholds are not configurable.

2.2.11.3 Test Evaluation Activity

FCS_SSH_EXT.1.1
There are no test evaluation activities for this component. This SFR is evaluated by activities for other SFRs.

Not applicable.

FCS_SSH_EXT.1.2

Test 1: [conditional] If the TOE is acting as SSH Server:

a. The evaluator shall use a suitable SSH Client to connect to the TOE, enable debug messages in the SSH Client, and examine the debug messages to determine that only the configured authentication methods for the TOE were offered by the server.

b. [conditional] If the SSH server supports X509 based Client authentication options:

   a. The evaluator shall initiate an SSH session from a client where the username is associated with the X509 certificate. The evaluator shall verify the session is successfully established.

   b. Next the evaluator shall use the same X509 certificate as above but include a username not associated with the certificate. The evaluator shall verify that the session does not establish.

   c. Finally, the evaluator shall use the correct username (from step a above) but use a different X509 certificate which is not associated with the username. The evaluator shall verify that the session does not establish.

The evaluator used an SSH client in debug mode to verify that the TOE supports SSH password and public key authentication.

The TOE does not support X.509 based SSH authentication.

FCS_SSH_EXT.1.2

Test 2: [conditional] If the TOE is acting as SSH Client, the evaluator shall test for a successful configuration setting of each authentication method as follows:

a. The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established.

b. Next, the evaluator shall use bad authentication data (e.g. incorrectly generated certificate or incorrect password) and ensure that the connection is rejected.

Steps a-b shall be repeated for each independently configurable authentication method supported by the server.

N/A, the TOE is not an SSH client.

FCS_SSH_EXT.1.2

Test 3: [conditional] If the TOE is acting as SSH Client, the evaluator shall verify that the connection fails upon configuration mismatch as follows:

a. The evaluator shall configure the Client with an authentication method not supported by the Server.

b. The evaluator shall verify that the connection fails.

N/A, the TOE is not an SSH client.

FCS_SSH_EXT.1.2

If the Client supports only one authentication method, the evaluator can test this failure of connection by configuring the Server with an authentication method not supported by the

Client. In order to facilitate this test, it is acceptable for the evaluator to configure an authentication method that is outside of the selections in the SFR.

N/A, the TOE is not an SSH client.

FCS_SSH_EXT.1.3

Test 1: The evaluator shall demonstrate that the TOE accepts the maximum allowed packet size.

The evaluator used an SSH test tool to send a packet just below the TOE's maximum size limit. The TOE was shown to accept this packet.

Modified by TD0732

FCS_SSH_EXT.1.3

Test 2: This test is performed to verify that the TOE drops packets that are larger than size specified in the component.

a. The evaluator shall establish a successful SSH connection with the peer.
b. Next the evaluator shall craft a packet that is ~~**one byte**~~ **slightly** larger than the maximum size specified in this component and send it through the established SSH connection to the TOE. **The packet should not be greater than the maximum packet size + 16 bytes. If the packet is larger, the evaluator shall justify the need to send a larger packet.**
c. **The e**valuator shall verify that the packet was dropped by the TOE**. The method of verification will vary by the TOE. Examples include ~~by~~** reviewing the TOE audit log for a dropped packet audit **or observing the TOE terminates the connection**.

The evaluator used an SSH test tool to send a packet just above the TOE's maximum size limit. The TOE was shown to terminate the connection when it received this packet.

FCS_SSH_EXT.1.4

If the TOE can be both a client and a server, these tests must be performed for both roles.

Test 1: The evaluator must ensure that only claimed algorithms and cryptographic primitives are used to establish an SSH connection. To verify this, the evaluator shall establish an SSH connection with a remote endpoint. 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 only the algorithms defined in the ST for the TOE for SSH connections. The evaluator shall perform one successful negotiation of an SSH connection and verify that the negotiated algorithms were included in the advertised set. If the evaluator detects that not all algorithms defined in the ST for SSH are advertised by the TOE or the TOE advertises additional algorithms not defined in the ST for SSH, the test shall be regarded as failed.

The data collected from the connection above shall be used for verification of the advertised hashing and shared secret establishment algorithms in FCS_SSH_EXT.1.5 and FCS_SSH_EXT.1.6 respectively.

As described in the [ST], aes128-cbc, aes256-cbc, aes128-ctr, and aes256-ctr are the encryption algorithms supported by the TOE. The evaluator opened SSH connections with the TOE and examined the Key Exchange Init packets. These showed that the claimed ciphers aligned with those that were advertised.

FCS_SSH_EXT.1.4

If the TOE can be both a client and a server, these tests must be performed for both roles.

Test 2: For the connection established in Test 1, the evaluator shall terminate the connection and observe that the TOE terminates the connection.

When the TOE is receiving an SSH connection and the client sends the Exit command the TOE terminates the connection as expected.

FCS_SSH_EXT.1.4

If the TOE can be both a client and a server, these tests must be performed for both roles.

Test 3: The evaluator shall configure the remote endpoint to only allow a mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the attempt fails.

The evaluator demonstrated that the TOE would not accept an AES192-CBC cipher as either a client or server.

FCS_SSH_EXT.1.5

Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.

The evaluator inspected the test data collected in FCS_SSH_EXT.1.4 Test 1 and verified that only the hmac-sha2-256 and hmac-sha2-512 MAC algorithms were present, as claimed in the ST.

FCS_SSH_EXT.1.5

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

The evaluator configured an SSH client to allow only the non-selected hmac-sha1 MAC. The evaluator attempted to connect to the TOE and verified that the connection failed.

FCS_SSH_EXT.1.6

Test 1: The evaluator shall use the test data collected in FCS_SSH_EXT.1.4, Test 1 to verify that appropriate mechanisms are advertised.

The evaluator inspected the data captured in the evidence for FCS_SSH_EXT.1.4 Test 1 and verified that only the diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp256, ecdh-sha2-nistp384, and ecdh-sha2-nistp521 kex algorithms were advertised, as claimed in the ST for the SSH Server interface.

FCS_SSH_EXT.1.6

Test 2: The evaluator shall configure an SSH peer to allow only a key exchange method that is not included in the ST selection. The evaluator shall attempt to establish an SSH connection and observe that the connection is rejected.

The evaluator configured an SSH peer to allow only the non-claimed diffie-hellman-group1-sha1 key exchange method. The evaluator verified that the connection failed.

FCS_SSH_EXT.1.7

None listed.

N/A

FCS_SSH_EXT.1.8

The test harness needs to be configured so that its connection rekey or termination limits are greater than the limits supported by the TOE -- it is expected that the test harness should not be initiating the connection rekey or termination.

FCS_SSH_EXT.1.8

Test 1: Establish an SSH connection. Wait until the identified connection rekey limit is met. Observed that a connection rekey or termination is initiated. This may require traffic to periodically be sent, or connection keep alive to be set, to ensure that the connection is not closed due to an idle timeout.

The evaluator used a proprietary test tool to connect to the TOE and start a timer awaiting a rekey while capturing packets. The evaluator observed that after 59 minutes and 55 seconds had passed, the tool indicated a rekey had occurred. The evaluator inspected the packet capture log and observed one large packet traveling from the server at the 3600 seconds mark(1hr). These packets indicate a rekey occurred and confirm the tool's output.

FCS_SSH_EXT.1.8

Test 2: Establish an SSH connection. Transmit data from the TOE until the identified connection rekey or termination limit is met. Observe that a connection rekey or termination is initiated.

The evaluator connected to the TOE with a proprietary SSH test tool that does not initiate any rekeys but will honor rekeys initiated by the peer. The evaluator induced the TOE to generate a large amount of output in the SSH session established, by instructing the TOE to repeatedly

print the log data present on the device and verified the TOE initiated a rekey once the 1 GB limit was elapsed.

FCS_SSH_EXT.1.8

Test 3: Establish an SSH connection. Send data to the TOE until the identified connection rekey limit or termination is met. Observe that a connection rekey or termination is initiated.

The evaluator used a proprietary tool to execute a command repeatedly in an SSH session to send data to the TOE. The evaluator observed that when the client had sent 1 GB of data, a rekey occurred.

2.2.12 FCS_SSHS_EXT.1 SSH Server Protocol

2.2.12.1 TSS Activities

No activities.

2.2.12.2 Guidance Activities

FCS_SSHS_EXT.1

The evaluator shall check the guidance documentation to ensure that it contains instructions to the administrator on how to ensure that only the allowed mechanisms are used in SSH connections with the TOE.

[CCECG] Section "Evaluated configuration" states that all configuration actions required to conform with Common Criteria requirements are carried out with a single master command (`enable common criteria`). No other configuration is necessary.

The (`enable common criteria`) command configures the following as part of the "SSH configuration" values set:

"Sets supported public key algorithms to rsa-sha2-256 and rsa-sha2-512 for server host key authentication to a remote SSH client and ssh-rsa for remote user authentication"

2.2.12.3 Test Activities

Modified by TD0682

FCS_SSHS_EXT.1

The evaluator shall ~~repeat Test 1 and Test 2 from FCS_SSH_EXT.1.4 for each of the authentication mechanisms supported by the TOE.~~ perform the following tests:

Test 1: The evaluator shall use a suitable SSH Client to connect to the TOE and examine the list of server host key algorithms in the SSH_MSG_KEXINIT packet sent from the server to the client to determine that only the configured server authentication methods for the TOE were offered by the server.

The evaluator used a nmap script to enumerate the host key algorithm that the TOE presents. The evaluator observed that only the rsa-sha2-256 and rsa-sha2-512 key algorithms were present, consistent with the ST.

Modified by TD0682

FCS_SSHS_EXT.1

Test 2: The evaluator shall test for a successful configuration setting of each server authentication method as follows. The evaluator shall initiate a SSH session using the authentication method configured and verify that the session is successfully established. Repeat this process for each independently configurable server authentication method supported by the server.

The evaluator configured an SSH client to allow only one supported server key algorithm and verified that the TOE supported it.

Modified by TD0682

FCS_SSHS_EXT.1

Test 3: ~~Next~~ The evaluator shall configure the **~~remote~~** peer to only allow an authentication mechanism that is not included in the ST selection. The evaluator shall attempt to connect to the TOE and observe that the **~~attempt fails~~ TOE sends a disconnect message.**

The evaluator configured an SSH client to allow only the unsupported ssh-rsa server key algorithm. The evaluator verified that an attempted connection to the TOE failed.

2.2.13 FCS_TLSC_EXT.1 TLS Client Protocol

2.2.13.1 TSS Activities

FCS_TLSC_EXT.1.1

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

Section 6.2.5.1 of [ST] lists the supported TLS cipher suites for the TOE's TLS client implementation, which are consistent with those that are claimed in the SFR.

FCS_TLSC_EXT.1.2

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

Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the administrator are relaxed and the identifier may also be established through a "Gatekeeper" discovery process. The TSS shall describe the discovery process and highlight how the reference

identifier is supplied to the "joining" component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

Section 6.2.5.1 of [ST] states that the TOE establishes the reference identifier for the external audit server based on the hostname (fully qualified domain name – FQDN; IP addresses are not supported) of the audit server configured by the **cliadmin** using the set log CLI command. The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125 section 6, and establishes a trusted channel only if the server certificate is valid. The TOE verifies the external server's presented identifier by comparing it to the configured reference identifier, matching the server's FQDN. The TOE does not support wildcards for peer authentication. Certificate pinning is not supported.

FCS_TLSC_EXT.1.2
If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC 5952 for IPv6, RFC 3986 for IPv4) is enforced.

N/A – IP addresses are not supported.

FCS_TLSC_EXT.1.3
None.

N/A

FCS_TLSC_EXT.1.4
If "present the Supported Groups Extension" is selected, the evaluator shall verify that TSS describes the Supported Groups Extension and whether the required behaviour is performed by default or may be configured. If TLS 1.2 is claimed and DHE ciphers are claimed, then the TSS must also specify whether the TOE is capable of negotiating DHE ciphers and whether the TOE client will terminate if an unsupported DHE parameter set is returned in the Server Key Exchange or whether all valid server-generated DHE parameters are accepted.

Section 6.2.5.1 of [ST] states that the TOE presents the Supported Elliptic Curves Extension in its Client Hello message, specifying the following curves/groups: secp256r1; secp384r1; secp521r1; ffdhe2048, ffdhe3072; ffdhe4096; ffdhe6144; and ffdhe8192. This is done by default and is not configurable.

FCS_TLSC_EXT.1.5

[Conditional]: The evaluator shall verify that TSS describes the signature_algorithms extension and whether the required behavior is performed by default or may be configured.

[Conditional]: The evaluator shall verify that TSS describes the signature_algorithms_cert extension and whether the required behavior is performed by default or may be configured.

Section 6.2.5.1 of [ST] states that the TOE presents the signature_algorithms extension with support for the following algorithms by default (this is not configurable):

- rsa_pkcs1 with sha256(0x0401)
- rsa_pkcs1with sha384(0x0501)
- rsa_pkcs1 with sha512(0x0601)
- ecdsa_secp256r1 with sha256(0x0403)
- ecdsa_secp384r1 with sha384(0x0503)
- ecdsa_secp521r1 with sha512(0x0603)
- rsa_pss_rsae with sha256(0x0804)
- rsa_pss_rsae with sha384(0x0805)
- rsa_pss_rsae with sha512(0x0806)
- rsa_pss_pss with sha256(0x0809)
- rsa_pss_pss with sha384(0x080a)
- rsa_pss_pss with sha512(0x080b)

FCS_TLSC_EXT.1.6

The evaluator shall verify that TSS describes whether the list of supported ciphersuites can be configured or not.

Section 6.2.5.1 of [ST] states that in in addition to the `enable common criteria` CLI command, the TOE provides the ability to configure the list of supported ciphersuites for the TOE's TLS client using the command: `<enable|disable> ciphersuites`.

FCS_TLSC_EXT.1.7

None

N/A

FCS_TLSC_EXT.1.8

The evaluator shall verify in the TSS that, for TLS 1.3, the TOE shall not permit out-of-band provisioning of pre-shared keys (PSKs) in the evaluated configuration.

Section 6.2.5.1 of [ST] states that in its evaluated configuration (after the **cliadmin** has executed the `enable common criteria` CLI command), the TOE supports TLS v1.2.

The TLS client does not support TLS 1.3.

FCS_TLSC_EXT.1.9

None

N/A

2.2.13.2 Guidance Activities

FCS_TLSC_EXT.1.1
The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS.

Section "Evaluated configuration" of [CCECG] describes how to `enable common criteria` mode and states that this is sufficient to ensure that the supported TLS versions and cipher suites are limited to those claimed in [ST].

[CCECG] Section "Evaluated configuration" explicitly states that the TOE supports TLS 1.2 using the following ciphersuites for the TLS client:

- TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288
- TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289.

FCS_TLSC_EXT.1.2
The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE includes support for IP addresses, the evaluator shall ensure that the operational guidance provides a set of warnings and/or CA policy recommendations that would result in secure TOE use.

[CCECG] Section "Configure a syslog server" provides the guidance to configure the reference identifier.

The `remotehost` parameter `hostname` must point to a syslog server. Configuring the hostname parameter with the FQDN will establish the reference identifier. You must specify either primary or secondary, followed by the hostname (FQDN) for the log server, or the reset command argument. The FQDN is compared against the CN or SAN (if present) in the presented certificate to verify the identity of the host matches the configured value.

Note: the use of IPv4 addresses in SAN or CN is not identified in the ST.

FCS_TLSC_EXT.1.2
Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1, the SFR selects attributes from RFC 5280, and FCO_CPC_EXT.1.2 selects "no

channel"; the evaluator shall verify the guidance provides instructions for establishing unique reference identifiers based on RFC 5280 attributes.

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

FCS_TLSC_EXT.1.3
None

N/A

FCS_TLSC_EXT.1.4
If the TSS indicates that the Supported Groups Extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the Supported Groups Extension.

Section "Evaluated configuration" of [CCECG] describes how to enable common criteria mode and states that this is sufficient to ensure that the Supported Groups Extension are limited to those claimed in [ST].

FCS_TLSC_EXT.1.5
If the TSS indicates that the signature_algorithms extension must be configured to meet the requirement, the evaluator shall verify that AGD guidance includes configuration of the signature_algorithms extension.

N/A – the TLS client ciphersuites are not configurable once the TOE has been placed into CC mode.

FCS_TLSC_EXT.1.6
If the TSF provides the ability of configuring the list of supported ciphersuites, the evaluator shall verify that AGD guidance includes configuration of the list of supported ciphersuites.

N/A – the TLS client ciphersuites are not configurable once the TOE has been placed into CC mode.

FCS_TLSC_EXT.1.7
None

N/A

FCS_TLSC_EXT.1.8
The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

The TOE does not support TLS 1.3. This evaluation activity is not applicable.

FCS_TLSC_EXT.1.9
None

N/A

2.2.13.3 Test Activities

FCS_TLSC_EXT.1.1
Test 1: The evaluator shall establish a TLS connection using each of the ciphersuites specified by the requirement. This connection may be established as part of the establishment of a higher-level protocol, e.g., as part of an HTTPS session. It is sufficient to observe the successful negotiation of a ciphersuite to satisfy the intent of the test; it is not necessary to examine the characteristics of the encrypted traffic to discern the ciphersuite being used (for example, that the cryptographic algorithm is 128-bit AES and not 256-bit AES).

The evaluator caused the TOE to initiate connections to a TLS server using each of the cipher suites claimed in the Security Target and confirmed that each connection succeeded.

FCS_TLSC_EXT.1.1
The goal of the following test is to verify that the TOE accepts only certificates with appropriate values in the extendedKeyUsage extension, and implicitly that the TOE correctly parses the extendedKeyUsage extension as part of X.509v3 server certificate validation.
Test 2: The evaluator shall establish the connection with a server presenting a certificate that contains the serverAuth (OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage extension and verify that the connection successfully negotiated. The evaluator shall then verify that when the same server presents an otherwise valid server certificate that contains the extendedKeyUsage extension without serverAuth the client rejects the connection. Ideally, the two certificates should be identical except for the OID values.

The evaluator confirmed that a TLS server presenting a certificate with the Server Authentication purpose in the extendedKeyUsage field resulted in a successful connection. The evaluator then configured the TLS server with a certificate without the Server Authentication purpose in the extendedKeyUsage field and attempted a connection and verified that the TOE rejected the connection.

FCS_TLSC_EXT.1.1
Test 3: [conditional]: Perform this test only if support of TLS 1.2 is claimed. The evaluator shall send a server certificate in the TLS connection that does not match the server-selected ciphersuite (for example, send an ECDSA certificate while using the TLS_RSA_WITH_AES_128_CBC_SHA ciphersuite). The evaluator shall verify that the TOE disconnects after receiving the server's Certificate handshake message.

The evaluator configured the TLS server to present an RSA server certificate with an ECDSA ciphersuite and attempted a connection from the TOE. The evaluator confirmed that the TOE rejected the connection.

FCS_TLSC_EXT.1.1
Test 4: The evaluator shall perform the following 'negative tests':

i. [conditional]: Perform this test only if support of TLS 1.2 is claimed. The evaluator shall configure the server to select the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the TOE TLS client denies the connection.

ii. Modify the server's selected ciphersuite in the Server Hello handshake message to be a ciphersuite (compatible with the server-selected version of TLS) not presented in the Client Hello handshake message. The evaluator shall verify that the TOE TLS client rejects the connection after receiving the Server Hello.

iii. The evaluator shall attempt to establish a TLS connection using each valid TLS/SSL version (i.e. TLS 1.3, TLS 1.2, TLS 1.1, TLS 1.0, SSL 3.0, SSL 2.0). The evaluator shall verify that the version(s) specified in FCS_TLSC_EXT.1.1 are successfully established and all other versions are rejected by the TOE TLS client. If a supported_versions extension is not sent by the TOE in the ClientHello, then the evaluator must ensure the test server responds with a ServerHello that is valid for the TLS version being negotiated. If the TOE includes the Supported Versions extension in its ClientHello, the evaluator shall also ensure the version(s) specified in the extension match the version(s) in FCS_TLSC_EXT.1.1. NOTE: For TLS 1.3 aware test servers, it is appropriate for the test server to issue a TLS Alert. The TOE client must not attempt to continue the connection.

For part i: The evaluator configured the TLS server to present the TLS_NULL_WITH_NULL_NULL ciphersuite in the Server Hello message and attempted a connection from the TOE. The evaluator confirmed that the TOE rejected the connection.

For part ii: the evaluator configured the TLS server to present a ciphersuite which is not present in the ClientHello message in the ServerHello message and attempted a connection from the TOE. The evaluator confirmed the TOE rejected the connection.

For part iii: The evaluator verified the TOE rejected connections when the server attempted to select any non-claimed protocol versions and only accepted connections when a claimed protocol version is selected by the server.

Test 5: The evaluator shall perform the following modifications to the traffic (i.e. Man-in-the-middle modifications that result in invalid signatures and MACs):

i. [conditional]: Perform this test only if support of TLS 1.2 is claimed. If using DHE or ECDH ciphersuites, modify the signature block in the Server's Key Exchange handshake message, and verify that the client denies the connection and no application data flows. The handshake shall be valid (e.g. the Finished message is calculated using the modified signature), with the exception of the invalid signature. This test does not apply to ciphersuites using RSA key exchange. If a TOE only supports RSA key exchange in conjunction with TLS, then this test shall be omitted.

ii. [conditional]: Perform this test only if support of TLS 1.3 is claimed. Modify the signature block in the Server's Certificate Verify handshake message, and verify that the client denies the connection and no application data flows. The handshake shall

be valid (e.g. the Finished message is calculated using the modified signature), with the exception of the invalid signature.

For part i: The evaluator configured a TLS server to modify the signature of the TLS server's ServerKeyExchange record and verified the TOE rejected the connection.

For part ii: The TOE does not claim TLS 1.3 support.

Test 6: The evaluator shall perform the following 'scrambled message tests':

i. Modify a byte in the Server Finished handshake message and verify that the handshake does not finish successfully and no application data flows. (Note: This modification must be performed prior to the contents of the Finished message being encrypted.)
ii. [conditional]: Perform this test only if support of TLS 1.2 is claimed. Send a garbled message from the server after the server has issued the ChangeCipherSpec message and verify that the handshake is not finished successfully and no application data flows. (Note: TLS 1.3 provides for a dummy ChangeCipherSpec message to aid in middlebox compatibility if such an option is enabled in the specific implementation [see Section D.4 in RFC 8446]. If TLS 1.3 middlebox compatibility mode is enabled a ChangeCipherSpec message may appear in packet traces, but it does not influence the protocol. To be clear: for TLS 1.3, this test does not need to be performed.)
iii. [conditional]: Perform this test only if support of TLS 1.3 is claimed. Send a plaintext EncryptedExtensions message from the server and verify that the handshake is not finished successfully and no application data flows. (Note: Under TLS 1.3, the EncryptedExtensions message is the first message to be encrypted with the handshake traffic secret.)
iv. [conditional]: Perform this test only if support of TLS 1.2 is claimed. Modify at least one byte in the server's nonce in the Server Hello handshake message and verify that the client rejects the Server Key Exchange handshake message (if using a DHE or ECDHE ciphersuite) or that the server denies the client's Finished handshake message.

For part i: The evaluator modified a byte in the Server Finished record before encrypting and sending it to the client and observed the TOE terminating the connection after receiving the Server Finished message.

For part ii: The evaluator configured a TLS server to send a garbled application data message instead of a Finished record after the ChangeCipherSpec message and confirmed the TOE rejected the connection.

For part iii: The TOE does not claim TLS 1.3 support.

For part iv: The evaluator configured a proprietary testing tool to modify a byte of the server's nonce value after it was sent to the TOE and use that modified value to calculate further Server Key Exchange/Finished data. The evaluator had the TOE attempt to connect to the server and verified that the connection failed.

FCS_TLSC_EXT.1.2

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

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

or

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

or

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

Note that for some tests additional conditions apply.

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

- IPv4: The CN contains a single address that is represented a 32-bit numeric address (IPv4) is written in decimal as four numbers that range from 0-255 separated by periods as specified in RFC 3986.
- IPv6: The CN contains a single IPv6 address that is represented as eight colon separated groups of four lowercase hexadecimal digits, each group representing 16 bits as specified in RFC 4291. Note: Shortened addresses, suppressed zeros, and embedded IPv4 addresses are not tested.

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

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

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

The evaluator configured the TLS server to present a certificate with a CN that does match the reference identifier and no SAN extension. The evaluator attempted a connection from the TOE and verified that the TOE did not accept the connection. For this test, the evaluator generated a certificate with an invalid FQDN.

FCS_TLSC_EXT.1.2

Test 2 [conditional]: The evaluator shall present a server certificate that contains a CN that matches the reference identifier, contains the SAN extension, but does not contain an

identifier in the SAN that matches the reference identifier. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported SAN type (e.g. IPv4, IPv6, FQDN, URI). When testing IPv4 or IPv6 addresses, the evaluator shall modify a single decimal or hexadecimal digit in the SAN.

The evaluator configured the TLS server to present a certificate with a CN that matched the reference identifier and a SAN extension with a value that did not match the reference identifier. The evaluator attempted a connection from the TOE and verified that the TOE did not accept the connection. The evaluator tested a bad FQDN in the SAN.

FCS_TLSC_EXT.1.2

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

The evaluator configured the TLS server to present a certificate with a valid CN and no SAN extension. The evaluator attempted a connection from the TOE and verified that the TOE accepted the connection. The evaluator tested FQDN identifiers in the CN field.

FCS_TLSC_EXT.1.2

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

The evaluator configured the TLS server to present a certificate with a CN that does not match the reference identifier and a SAN extension with a value that does match the reference identifier. The evaluator attempted a connection from the TOE and verified that the TOE accepted the connection. The evaluator tested FQDN identifiers in the SAN extension.

FCS_TLSC_EXT.1.2

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

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

ii. [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds, if wildcards are supported, or fails if wildcards are not supported. The evaluator shall configure the reference identifier without a left-most

label as in the certificate (e.g. example.com) and verify that the connection fails. The evaluator shall configure the reference identifier with two left-most labels (e.g. bar.foo.example.com) and verify that the connection fails. (Remark: Support for wildcards was always intended to be optional. It is sufficient to state that the TOE does not support wildcards and observe rejected connection attempts to satisfy corresponding assurance activities.)

The TOE does not claim to support wildcards.

The evaluator presented the TOE with a server certificate with a wildcard in the leftmost part of the CN. The TOE was shown to reject the connection.

FCS_TLSC_EXT.1.2

Test 6: Objective: The objective of this test is to ensure the TOE is able to differentiate between IP address identifiers that are not allowed to contain wildcards and other types of identifiers that may contain wildcards.

[conditional]: If IP address identifiers are supported in the SAN or CN, the evaluator shall present a server certificate that contains a CN that matches the reference identifier, except one of the groups has been replaced with a wildcard asterisk (*) (e.g. CN=*.168.0.1 when connecting to 192.168.0.1, CN=2001:0DB8:0000:0000:0008:0800:200C:* when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6).

This test is not applicable as the TOE does not utilize IP reference identifiers.

FCS_TLSC_EXT.1.2

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

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

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

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

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

This test is not applicable because the TOE is not distributed and does not claim FPT_ITT.1.

FCS_TLSC_EXT.1.3

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

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

The evaluator verified the TOE accepts connections when the server presents a valid certificate that chains to a CA trusted by the TOE.

FCS_TLSC_EXT.1.3

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

Test 2 [conditional]: If "except with the following administrator override" is selected, the evaluator shall change the presented certificate(s) or modify the operational environment, so that certificate validation fails due to the TSF's inability to determine revocation status. The evaluator shall verify that the certificate is not accepted by the TSF until the Security Administrator authorizes the TSF to establish the connection and this action results in the Trusted Channel being successfully established.

This test is not applicable as the TOE does not make the "except with the following administrator override" selection in the ST.

FCS_TLSC_EXT.1.3

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

Test 3: While performing testing of invalid TLS Client Reference Identifiers, expired X.509 certificates, and invalid X.509 trust chains; the evaluator shall ensure the TSF does not present an administrator override option, with the exception of failure to determine revocation status (if selected). Note: This should be a review of behavior observed while performing other tests.

While conducting the invalid identifiers test case for the TLSC interface in FCS_TLSC_EXT.1.2 test 1 and 2, the evaluator observed that no optional override function was presented to the administrator. A similar observation was made when conducting the expired certificate test(FIA_X509_EXT.1.1 Test 2) and an invalid chain certificate chain test(FCS_TLSC_EXT.1.3 Test 1).

FCS_TLSC_EXT.1.4
Test 1 [conditional]: If “not present the Supported Groups Extension” is selected, the evaluator shall examine the Client Hello message and verify it does not contain the Supported Groups extension.

This test is not applicable as the TOE makes the selection to present the Supported Groups Extension.

`FCS_TLSC_EXT.1.4
Test 2 [conditional]: If “present the Supported Groups Extension” is selected, the evaluator shall configure the server to perform ECDHE or DHE (as applicable) key exchange using each of the TOE’s supported groups. The evaluator shall verify that the connection succeeds. This test shall be repeated for each type of key exchange message/extension supported (i.e. Key Share extension for TLS 1.3 and Server Key Exchange Message for TLS 1.2).

The evaluator verified that the TOE could successfully establish a TLS client connection using each of the specified ECDHE curve values for the key exchange algorithm.

FCS_TLSC_EXT.1.4
Test 3 [conditional]: If secp curves are selected, the evaluator shall configure the server to perform an ECDHE key exchange in the TLS connection using a non-supported curve and shall verify that the connection fails and no application data flows. The non-supported curve shall be as similar to the selected curve(s) as possible (i.e. a non-selected curve when not all curves are selected or P-224). This test shall be repeated for each type of key exchange message/extension supported (i.e. Key Share extension for TLS 1.3 and Server Key Exchange Message for TLS 1.2).

The evaluator configured the TLS server to select a key exchange that is not supported by the TOE, secp192r1, and verified that the TOE rejected the connection.

FCS_TLSC_EXT.1.4
Test 4a [conditional, for TLS 1.3 only]: If ffdhe curves are selected, the evaluator shall configure the server to perform a DHE key exchange in the TLS connection using a non-supported group and shall verify that the connection fails and no application data flows. The non-supported group shall be as similar to the selected group(s) as possible (i.e. a non-selected group when not all groups are selected or undefined Codepoint 0x0105 (ffdhe8192 + 1)).

This test is not applicable as the TOE does not claim TLS 1.3 support.

FCS_TLSC_EXT.1.4
Test 4b [conditional, for TLS 1.2 only]: If ffdhe curves are selected, the evaluator shall configure the server to return DHE parameters in the Server Key Exchange in the TLS connection that do not meet the construction for any claimed ffdhe group. The evaluator shall

verify that the connection fails and no application data flows. If the TOE client supports any server-returned DHE parameter set, then this test is not applicable.

This test is not applicable as the TOE supports any server-returned DHE parameter set.

FCS_TLSC_EXT.1.5

Test 1 [conditional]: The evaluator shall perform the following tests if "present the signature_algorithms extension" is selected:

i. The evaluator shall examine the Client Hello message and verify it contains the signature_algorithms extension and the SignatureSchemes match the SignatureSchemes specified in the requirement.

ii. The evaluator shall establish a TLS connection using each of the SignatureSchemes specified by the requirement and observes the session is successfully completed. The evaluator shall ensure the test server sends a leaf Certificate that has a public key algorithm that is consistent with the SignatureScheme being tested. For TLS 1.2 and if the ciphersuite is DHE or ECDHE, the evaluator shall ensure that the server sends Server Key Exchange messages consistent with the SignatureScheme being tested. For TLS 1.3, the evaluator shall ensure that the server sends Certificate Verify messages consistent with the SignatureScheme being tested.

For part i: The evaluator examined the ClientHello sent by the TOE and verified that the SignatureAlgorithms offered by the TOE is consistent with the claim in the ST.

For part ii: The evaluator verified the TOE could successfully complete a connection using each of the claimed signature algorithms as the algorithm for the ServerkeyExchange signature algorithm.

FCS_TLSC_EXT.1.5

Test 2 [conditional]: The evaluator shall perform the following tests if "present the signature_algorithms_cert extension" is selected:

i. The evaluator shall examine the Client Hello message and verify it contains the signature_algorithms_cert extension and the SignatureSchemes match the SignatureSchemes specified in the requirement.

ii. The evaluator shall establish a TLS connection using a certificate chain using each of the SignatureSchemes specified by the requirement. The evaluator shall ensure the signatures used in the certificate chain are consistent with the SignatureScheme being tested.

This test is not applicable as the TOE does not make the "present the signature_algorithms_cert extension".

FCS_TLSC_EXT.1.6

[conditional]: If the TSF provides the ability of configuring the list of supported ciphersuites, the evaluator shall establish a TLS connection using one of the possible configurations of the list of supported ciphersuites. The evaluator shall then change the configuration and repeat

the test. The evaluator shall verify that the behavior of the TOE has changed according to the modification of the list of ciphers. This test shall be repeated for all supported TLS versions. If the TSF does not provide the ability of configuring the list of supported ciphersuites, this test shall be omitted.

This test is not applicable as the TOE does not provide the ability to configure the list of supported ciphersuites.

FCS_TLSC_EXT.1.7
The evaluator shall establish a TLS connection with a server and observe that the early data extension and the post-handshake client authentication extension according to RFC 8446 Section 4.2 are not advertised in the Client Hello Message. This test shall be executed for all TLS versions supported by the TOE.

The evaluator verified the TOE does not send the EarlyData or PostHandshake ClientAuthentication extensions in the ClientHello message.

FCS_TLSC_EXT.1.8
None

N/A

FCS_TLSC_EXT.1.9
Test 1 [conditional]: If "support TLS 1.2 secure renegotiation…" is selected, the evaluator shall use a network packet analyzer/sniffer to capture a TLS 1.2 handshake between the two TLS endpoints. The evaluator shall verify that either the “renegotiation_info” field or the SCSV ciphersuite is included in the ClientHello message during the initial handshake.

This test is not applicable as the TOE does not make the “support TLS 1.2 secure renegotiation” selection.

FCS_TLSC_EXT.1.9
Test 2 [conditional]: If "support TLS 1.2 secure renegotiation…" is selected, the evaluator shall perform a TLS 1.2 handshake and verify the TOE TLS Client’s handling of ServerHello messages received during the initial handshake that include the “renegotiation_info” extension. The evaluator shall modify the length portion of this field in the ServerHello message to be non-zero and verify that the TOE TLS client sends a failure and terminates the connection. The evaluator shall verify that a properly formatted field results in a successful TLS connection.

This test is not applicable as the TOE does not make the “support TLS 1.2 secure renegotiation” selection.

FCS_TLSC_EXT.1.9
Test 3 [conditional]: If "support TLS 1.2 secure renegotiation…" is selected, the evaluator shall perform a TLS 1.2 handshake and verify that ServerHello messages received during secure renegotiation contain the “renegotiation_info” extension. The evaluator shall modify either

the “client_verify_data” or “server_verify_data” value and verify that the TOE TLS client terminates the connection.

This test is not applicable as the TOE does not make the “support TLS 1.2 secure renegotiation” selection.

FCS_TLSC_EXT.1.9

Modified by TD0899

Test 4a [conditional, if the TOE supports TLS 1.3]: The evaluator shall initiate a TLS session between the TSF and a test TLS 1.3 server that completes a compliant TLS 1.3 handshake, followed by a hello request message. The evaluator shall observe that the TSF completes the initial TLS 1.3 handshake successfully, but terminates the session on receiving the hello request message.

It is preferred that the TSF sends a fatal error alert message (e.g., unexpected message) in response to this, but it is acceptable that the TSF terminates the connection silently (i.e., without sending a fatal error alert).

Test 4b [conditional]: I, if the TOE supports TLS 1.2 and "rejects TLS 1.2 …renegotiation attempts" is selected, then for each selected TLS version, t]: The evaluator shall initiate a TLS session between the so-configured TSF and a test TLS 1.2 server that is configured to perform a compliant handshake, followed by a hello reset request. The evaluator shall confirm that the TSF completes the initial handshake successfully but terminates the TLS sessiondoes not initiate renegotiation after receiving the hello reset request. Note: It is preferred that the TSF sends a fatal error alert message (e.g., unexpected message) in response to this, but it is acceptable that the TSF terminates the connection silently (i.e., without sending a fatal error alert).

Test 4a: N/A, the TOE does not support TLS 1.3.

Test 4b: The evaluator configured a proprietary testing tool to establish a TLS handshake and attempt to renegotiate. The evaluator observed that the TOE refused to renegotiate the connection and remained on the existing negotiated parameters & session.

2.2.14 FCS_TLSS_EXT.1 TLS Server Protocol

2.2.14.1 TSS Activities

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.

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

Section 6.2.5.2 of [ST] lists the TLS ciphersuites that are supported by the TOE when it acts as a TLS server. This list is consistent with what is claimed in FCS_TLSS_EXT.1.1.

In its evaluated configuration, the TOE will accept ClientHello messages that specify support for TLS v1.2. It will reject ClientHello messages that specify support only for TLS v1.1 or older versions of TLS/SSL, or for TLS v1.3.

FCS_TLSS_EXT.1.2
The evaluator shall verify that the TSS describes the algorithms and key sizes the TSF supports for authenticating itself to TLS clients. The evaluator shall ensure these algorithms are consistent with the selected ciphersuites.

Section 6.2.5.2 of [ST] states that the TOE authenticates itself using X.509 certificates using 2048, 3072, 4096 bits RSA. These algorithms are consistent with the selected ciphersuites.

FCS_TLSS_EXT.1.3
The evaluator shall verify that the TSS lists all EC Diffie-Hellman curves and/or Diffie-Hellman groups used in the key establishment by the TOE when acting as a TLS Server. The evaluator shall ensure these algorithms are consistent with the selected ciphersuites.

Section 6.2.5.2 of [ST] states that when the TOE negotiates a cipher suite that uses DHE as its key exchange algorithm, it sends a Server Key Exchange message that specifies the supported Diffie-Hellman parameters (ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192).

When the TOE negotiates a cipher suite that uses ECDHE as its key exchange algorithm, it sends a Server Key Exchange message that specifies the supported NIST curves (secp256r1, secp384r1, and secp521r1), the ECDH public key, and the associated domain parameters.

These algorithms are consistent with the selected ciphersuites.

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), if session resumption based on session tickets is supported (RFC 5077) and/or if session resumption according to RFC 8446 is supported.

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

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

[ST] section 6.2 5.2 states the TOE rejects all renegotiation attempts for its TLS interfaces and does not support session resumption.

FCS_TLSS_EXT.1.4
If the TOE claims a TLS server capable of session resumption (as a single context, or across multiple contexts), the evaluator shall verify that the TSS describes how session resumption operates (i.e. what would trigger a full handshake, e.g. checking session status, checking Session ID, etc.). If multiple contexts are used, the TSS describes how session resumption is coordinated across those contexts. In case session establishment and session resumption are always using a separate context, the TSS shall describe how the contexts interact with respect to session resumption (in particular regarding the session ID). It is acceptable for sessions established in one context to be resumable in another context.

[ST] section 6.2 5.2 states the TOE rejects all renegotiation attempts for its TLS interfaces and does not support session resumption.

FCS_TLSS_EXT.1.5
The evaluator shall verify that TSS describes whether the list of supported ciphersuites can be configured or not.

[ST] section 6.2 5.2 states that in addition to the `enable common criteria` CLI command, the TOE provides the ability to configure the list of supported ciphersuites for the TOE's TLS client using: <`enable|disable cipher-suites`>.

FCS_TLSS_EXT.1.6
None

N/A

FCS_TLSS_EXT.1.7
The evaluator shall verify in the TSS that, for TLS 1.3, the TOE shall not permit out-of-band provisioning of pre-shared keys (PSKs) in the evaluated configuration.

The TOE does not support TLS 1.3.

FCS_TLSS_EXT.1.8
None

N/A

2.2.14.2 Guidance Activities

FCS_TLSS_EXT.1.1
The evaluator shall check the guidance documentation to ensure that it contains instructions on configuring the TOE so that TLS conforms to the description in the TSS (for instance, the set of ciphersuites advertised by the TOE or TLS version supported by the TOE may have to be restricted to meet the requirements).

Section "Evaluated configuration" of [CCECG] describes the set of ciphersuites advertised by the TOE and the TLS version supported by the TOE.

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

[CCECG] Section "Master command to enable all required settings" states that the `enable common criteria` command enables all settings required for Common Criteria compliance.

FCS_TLSS_EXT.1.3
The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

[CCECG] Section "Master command to enable all required settings" states that the `enable common criteria command` enables all settings required for Common Criteria compliance.

FCS_TLSS_EXT.1.4
The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

The ST states that session resumption is not supported. Therefore, this evaluation activity is not applicable.

FCS_TLSS_EXT.1.5
If the TSF provides the ability of configuring the list of supported ciphersuites, the evaluator shall verify that AGD guidance includes configuration of the list of supported ciphersuites.

[CCECG] Section "Master command to enable all required settings" states that the enable common criteria command enables all settings required for Common Criteria compliance.

FCS_TLSS_EXT.1.6
None

N/A

FCS_TLSS_EXT.1.7
The evaluator shall verify that any configuration necessary to meet the requirement must be contained in the AGD guidance.

The ST states that the TOE does not support the use of PSKs. Therefore, this evaluation activity is not applicable.

FCS_TLSS_EXT.1.8
None

N/A

2.2.14.3 Test Activities

FCS_TLSS_EXT.1.1

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

The evaluator used the sslyze tool to test the ciphersuites supported by the TOE. The evaluator confirmed that negotiation of each claimed ciphersuite was successful and the connection was successfully established.

FCS_TLSS_EXT.1.1

Test 2: The evaluator shall perform the following tests:

i. 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.

ii. [conditional]: Perform this test only if support of TLS 1.2 is claimed. 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.

For part i: The evaluator configured a TLS client to only offer ciphersuites in the ClientHello which are not claimed by the TOE and verified the TOE rejected the connection.

For part ii: The evaluator configured a TLS client to only offer the TLS_NULL_WITH_NULL_NULL ciphersuite in its ClientHello and verified the TOE rejected the connection.

FCS_TLSS_EXT.1.1

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

i. [conditional]: Perform this test only if support of TLS 1.2 is claimed. Modify a byte in the Client Finished handshake message, and verify that the server rejects the connection and does not send any application data.

(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 TLS Finished message and b) Encrypt every TLS message after session keys are negotiated.)

ii. [conditional]: Perform this test only if support of TLS 1.2 is claimed. 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'.

iii. [conditional]: Perform this test only if support of TLS 1.3 is claimed. The evaluator shall use a client to send a Client Hello message containing a single curve in the Supported Groups extension. The curve that is selected to be presented in this extension should not be supported by the TOE. The evaluator shall verify that the TOE disconnects after receiving the Client Hello message.

iv. [conditional]: Perform this test only if support of TLS 1.3 is claimed. The evaluator shall use a client to send a Client Hello message containing multiple curves in the Supported Groups extension. These curves should be chosen such that only one of these curves is supported by the TOE. The evaluator shall verify that the TOE responds with a Hello Retry Request message selecting the supported curve. This shall be reflected in the Key Share extension of the Hello Retry Request message.

For part i: The evaluator configured a TLS client to modify the Client Finished handshake message. The evaluator observed the TOE rejected the connection and did not transmit any application data.

For part ii: The evaluator examined a TLS finished record sent by the TOE to the client and verified that the Finished record was truly encrypted and not readable in plaintext by the evaluator.

For part iii: This is not applicable as the TOE does not support TLS 1.3.

For part iv: This is not applicable as the TOE does not support TLS 1.3.

FCS_TLSS_EXT.1.1

Test 4: The evaluator shall attempt to establish a TLS/SSL connection using each of the supported TLS/SSL versions (i.e., TLS 1.3, TLS 1.2, TLS 1.1, TLS 1.0, SSL 3.0, SSL 2.0). The client shall be configured so it only supports the version being tested. The evaluator shall verify that the versions specified in FCS_TLSS_EXT.1.1 are successfully established and all other versions not successfully established. If the TOE attempts to downgrade the version, it is acceptable for the test client to terminate the connection; however, the version selected by the TOE shall always be a version specified in FCS_TLSS_EXT.1.1.

The evaluator verified the TOE did not accept connections from clients attempting to negotiate a protocol version which is not supported by the TOE (TLS 1.3, TLS 1.1, TLS 1.0, SSL 3.0, SSL 2.0) and only accepts connections from clients attempting to negotiate the supported protocol version (TLS 1.2).

FCS_TLSS_EXT.1.2 and FCS_TLSS_EXT.1.3

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

The evaluator shall repeat this test for each supported elliptic curve. The evaluator shall attempt a connection using a supported ECDHE ciphersuite (TLS 1.2) or group (TLS 1.3) and a single supported elliptic curve specified in the supported groups extension. The Evaluator shall verify (through a packet capture or instrumented client) that the TOE selects the same curve in the Server Key Exchange (TLS 1.2) or Server Hello (key_share, for TLS 1.3) message and successfully establishes the connection.

For TLS 1.2, the evaluator shall attempt a connection using a supported ECDHE ciphersuite and a single unsupported elliptic curve (e.g., secp192r1 (0x13)) specified in RFC 4492, Section 5.1.1. The evaluator shall verify that the TOE does not send a Server Hello message and the connection is not successfully established.

For TLS 1.3, the evaluator shall attempt a connection using a supported ciphersuite and a single unsupported group. Both the key_share and supported_groups extensions must be set to the same unsupported group. The evaluator shall verify that the TOE does not send a Server Hello message and the connection is not successfully established.

The evaluator configured a TLS client to specify only one curve supported by the TOE in the Elliptic Curves extension. The evaluator observed that the TOE selected the same curve and

established a connection successfully. This was repeated for each claimed curve secp256r1, secp384r1, secp521r1.

The evaluator configured a TLS client to specify a curve not supported by the TOE in the Elliptic Curves extension (specifically, secp192k1). The evaluator verified the TOE did not establish a connection or return a Server Hello record.

FCS_TLSS_EXT.1.2 and FCS_TLSS_EXT.1.3

Test 2 [conditional]: If DHE ciphersuites are supported, the evaluator shall repeat the following test for each supported parameter size. If any configuration is necessary, the evaluator shall configure the TOE to use a supported Diffie-Hellman parameter size. The evaluator shall attempt a connection using a supported DHE ciphersuite.

For TLS 1.2, 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).

For TLS 1.3, the evaluator shall verify (through a packet capture or instrumented client) that the TOE sends a Server Key Share Extension Message where the KeyShareServerHello structure contains a KeyShareEntry structure with an opaque key_exchange value whose Length is consistent with the configured Diffie-Hellman parameter size(s).

The evaluator connected to the TOE using a TLS client tool. The evaluator observed that the TOE would accept connections using each of the DHE parameters specified in the [ST].

FCS_TLSS_EXT.1.2 and FCS_TLSS_EXT.1.3

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

This is not applicable as the TOE does not use any RSA key establishment ciphersuites.

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 RFC 5246 (TLS 1.2) or session tickets according to RFC 5077 (TLS 1.2) or session resumption according to RFC 8446 (TLS 1.3), the evaluator shall perform the following test:

i. For all supported TLS versions the client shall send a Client Hello with a zero-length session identifier and with a SessionTicket extension containing a zero-length ticket. A non-zero length session identifier for TLS 1.3 would result in testing compatibility

mode which is not the objective of this test. For TLS 1.3, the evaluator shall ensure that a 'psk_key_exchange_modes' extension is included in the Client Hello.

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

iii. The client verifies the Server Hello message contains a zerolength session identifier. For TLS 1.2 the client could alternatively pass the following steps (not applicable for TLS 1.3):

Note: The following steps are only performed if the ServerHello message contains a non-zero length SessionID.

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

v. 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).

vi. 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.

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

The evaluator demonstrated that the TOE does not support session resumption. The TOE did not send a NewSessionTicket message when opening a connection. The TOE did send a Session ID in its Server Hello message, but using that Session ID in a Client Hello did not result in the old connection being resumed.

FCS_TLSS_EXT.1.4

Test 2 [conditional]: If the TOE supports session resumption using session IDs according to RFC 5246 (TLS 1.2), 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):

i. 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). When the session is resumed, the

evaluator shall verify on the TLS Client used for performing this test, that the TOE (TLS Server) has not advertised support for the early data extension.

ii. 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.

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

This test is not applicable as the TOE does not support resumption based on RFC 5246.

FCS_TLSS_EXT.1.4

Test 3 [conditional]: If the TOE supports session tickets according to RFC 5077 (supported only by TLS 1.2), the evaluator shall carry out the following steps:

i. 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. When the session is resumed, the evaluator shall verify on the TLS Client used for performing this test, that the TOE (TLS Server) has not advertised support for the early data extension.

ii. 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 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.

Remark: If multiple contexts are supported for session resumption, for each of the above test cases, the session ticket may be obtained in one context for resumption in another context.

There is no requirement that the session ticket be obtained and replayed within the same context subject to the description provided in the TSS. All contexts that can reuse a session ticket constructed in another context must be tested. It is not mandated that the session establishment and session resumption share context. For example, it is acceptable for a control channel to establish and application channel to resume the session.

This test is not applicable as the TOE does not support resumption based on RFC 5077.

FCS_TLSS_EXT.1.4

Test 4 [conditional]: If the TOE supports session resumption according to RFC 8446 (supported only by TLS 1.3), the evaluator shall carry out the following steps:

i. 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 pre-shared key in the ClientHello. The evaluator shall confirm that the TOE responds similarly to figure 3 of RFC 8446 after successfully reusing the pre-shared-key to resume the session. Specifically, the server must not send back a Certificate message if the session is correctly resumed. When the session is resumed, the evaluator shall verify on the TLS Client used for performing this test, that the TOE (TLS Server) has not advertised support for the early data extension.

ii. 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 modify the pre-shared key 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, or (2) terminates the connection in some way that prevents the flow of application data.

iii. 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 force the non-TOE client to attempt to establish a new connection using the previous session ticket material as a pre-shared key, but set psk_key_exchange_modes with a value of psk_ke in the Client Hello message and omit the psk_ke_dhe. The evaluator shall confirm that the TOE either (1) implicitly rejects the session ticket by performing a full handshake, or (2) terminates the connection in some way that prevents the flow of application data.

This test is not applicable as the TOE does not support session resumption based on RFC 8446.

FCS_TLSS_EXT.1.5

Test 1[conditional]: If the TSF provides the ability of configuring the list of supported ciphersuites, the evaluator shall establish a TLS connection using one of the possible configurations of the list of supported ciphersuites. The evaluator shall then change the configuration and repeat the test. The evaluator shall verify that the behavior of the TOE has changed according to the modification of the list of ciphers. This test shall be repeated for all

supported TLS versions. If the TSF does not provide the ability of configuring the list of supported ciphersuites, this test shall be omitted.

The evaluator configured the TOE to accept a specific ciphersuite and then verified it would accept a TLS connection using that ciphersuite. The ciphersuite was then removed and the TOE would no longer open a connection using it.

FCS_TLSS_EXT.1.6
According to RFC 8446 Section 4.2.10, a PSK is required to use the early data extension. As NDcPP only allows the use of PSK in conjunction with session resumption, a NDcPP conformant TOE which acts as TLS Server cannot use the early data extension if session resumption is not supported. For TOEs that do not support session resumption, execution of test FCS_TLSS_EXT.1.4 Test 1 is regarded as sufficient that the TOE does not support the early data extension. For TOEs that support session resumption, FCS_TLSS_EXT.1.4 Test 2(i), 3(i) or 4(i) (depending on the supported TLS versions and the way session resumption is implemented) ensure that the TOE does not support the early data extension.

The TOE does not support session resumption and therefore does not have any scenario which uses a PSK to secure a TLS connection.

FCS_TLSS_EXT.1.7
None

N/A, no activity defined.

FCS_TLSS_EXT.1.8
Test 1 [conditional]: If "support secure renegotiation…" is selected, the evaluator shall use a network packet analyzer/sniffer to capture a TLS 1.2 handshake between the two TLS endpoints. The evaluator shall verify that the “renegotiation_info” extension is included in the ServerHello message.

This test is not applicable as the TOE does not make the “support secure renegotiation” selection.

FCS_TLSS_EXT.1.8
Test 2 [conditional]: If "support secure renegotiation…" is selected, the evaluator shall perform a TLS 1.2 handshake and modify the length portion of the field in the ClientHello message in the initial handshake to be non-zero. The evaluator shall verify that the TOE TLS server sends a failure and terminates the connection. The evaluator shall verify that a properly formatted field results in a successful TLS connection.

This test is not applicable as the TOE does not make the “support secure renegotiation” selection.

FCS_TLSS_EXT.1.8

Test 3 [conditional]: If "support secure renegotiation…" is selected, the evaluator shall perform a TLS 1.2 handshake and modify the "client_verify_data" or "server_verify_data" value in the ClientHello message received during secure renegotiation. The evaluator shall verify that the TOE TLS server terminates the connection.

This test is not applicable as the TOE does not make the “support secure renegotiation” selection.

FCS_TLSS_EXT.1.8

Modified by TD0899

Test 4a [conditional, if the TOE supports TLS 1.3]: The evaluator shall follow the operational guidance as necessary to configure the TSF to negotiate TLS 1.3. The evaluator shall initiate a valid initial TLS 1.3 session, send a valid client hello on the non-renegotiable TLS channel, and observe that the TSF terminates the session.

Note: It is preferred that the TSF sends a fatal error alert message (e.g., unexpected message) in response to this, but it is acceptable that the TSF terminates the connection silently (i.e., without sending a fatal error alert).

Test 4b [conditional, if the TOE supports TLS 1.2 and]: If "rejects TLS 1.2 …renegotiation attempts" is selected, then for each selected TLS version, t]: The evaluator shall follow the operational guidance as necessary to configure the TSF to negotiate the versionTLS 1.2 and reject renegotiation. The evaluator shall initiate a valid initial TLS 1.2 session for the specified version, send a valid ClientHello on the non-renegotiable TLS channel, and observe that the TSF does not perform renegotiation of the TLS channel. terminates the session. Note: It is preferred that the TSF sends a fatal error alert message (e.g., unexpected message) in response to this, but it is acceptable that the TSF terminates the connection silently (i.e., without sending a fatal error alert).

The TOE terminated a TLS 1.2 connection after receiving a session renegotiation attempt.

2.3 Identification and Authentication (FIA)

2.3.1 FIA_AFL.1 Authentication Failure Management

2.3.1.1 TSS Activities

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

Section 6.3.2 of [ST] states the TOE implements a mechanism to respond to consecutive failures to authenticate a remote login attempt using a password, within a specified time window (failure window). By default, the TOE allows three attempts to enter a valid password within a 15-minute failure window. If the configured number of failed attempts is reached within the

failure window, the TOE locks the user account and prevents the user from logging in. The mechanism applies to all local and remote password-based interfaces, including the browser-based Web Beta client, the Java client, the remote client, the Web UI, and the CLI.

The mechanism applies to all remote password-based interfaces, including the browser-based Web Beta client, the Java client, the remote client, the Web UI, and the CLI. The default lockout period is 600 seconds (10 minutes).

In the event that a user becomes locked out due to the consecutive authentication failure lockout function, the **cliadmin** can use the enable user command to unlock the user or the **uiadmin** can use the Java Client, remote client or Web Beta client to unlock the user: `EDIT -> System Settings -> Users ->` Right click user and click Edit. Alternatively, the user can wait for the configured lockout period to pass.

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

Section 6.3.2 of [ST] states that in the evaluated configuration the **cliadmin** must be configured to login using SSH public key in order to ensure continuity of administrator access in the event that all other administrators using a password become locked-out due to the lockout function.

2.3.1.2 Guidance Activities

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

[CCECG] Section “Authentication failure lockouts” provides the guidance to configure the authentication failure parameters.

The enable common criteria command automatically sets authentication failure parameters. For more information, see the Evaluated configuration section in this guide. However, the user lockout threshold and duration are also manually configurable. To set these parameters use the following commands:

```
> set property server redseal.srm.authentication.lockout,

> set property server redseal.srm.authentication.max_failure_count,

> set property server
redseal.srm.authentication.lockout_duration_seconds
```

The authentication failure lockout settings are applied to all password-based authentication and all administrative interfaces (i.e. this is not configured on a per-interface or per-account basis).

Authentication failure lockout does not apply to SSH public key authentication. For more information on setting these parameters manually, see the Installation and Administration Guide section on the Command Line Interface.

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

[CCECG] Section “Session inactivity lockout best practice” provides the instructions so that remote admins are not permanently locked out. Set the timeout to a value in minutes to make sure that remote admins are not permanently locked out.

[CCECG] “Section “Unlocking administrator accounts” further indicates that “it is strongly recommended that the cliadmin account be configured to log in using SSH public key”

2.3.1.3 Test Activities

The evaluator shall perform the following tests for each method by which remote administrators access the TOE (e.g. any passwords entered as part of establishing the connection protocol or the remote administrator application): **Test 1:** The evaluator shall use the operational guidance to configure the number of successive unsuccessful authentication attempts allowed by the TOE (and, if the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall also use the operational guidance to configure the time period after which access is re-enabled). The evaluator shall test that once the authentication attempts limit is reached, authentication attempts with valid credentials are no longer successful.

The following evidence can be seen in FIA_AFL.1 Test 2. In the first part of that test, the evaluator configures the lockout parameters, uses an invalid password to meet the lockout requirement and attempts to login with a valid password which is then rejected by the TOE.

Test 2: After reaching the limit for unsuccessful authentication attempts as in Test 1 above, the evaluator shall proceed as follows. If the administrator action selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall confirm by testing that following the operational guidance and performing each action specified in the ST to re-enable the remote administrator’s access results in successful access (when using valid credentials for that administrator). If the time period selection in FIA_AFL.1.2 is included in the ST, then the evaluator shall wait for just less than the time period configured in Test 1 and show that an authorisation attempt using valid credentials does not result in successful access. The evaluator shall then wait until just after the time period configured in Test 1 and show that an authorisation attempt using valid credentials results in successful access.

The evaluator configured a threshold of 3 unsuccessful attempts for account lockout and a lockout time of 2 minutes. 3 login attempts with a bad password were then made. After that the

TOE was shown to block connection attempts even with good credentials for 2 minutes. Once the lockout time period expired the valid credentials were attempted again and found to successfully authenticate to the TOE establishing a session for the identified user. This was repeated for each interface.

The evaluator verified that a locked account could be unlocked by an administrative user restoring the ability for the account to log in with valid credentials after the unlock action was performed.

2.3.2 FIA_PMG_EXT.1 Password Management

2.3.2.1 TSS Activities

The evaluator shall check that the TSS:

a. lists the supported special character(s) for the composition of administrator passwords.

b. to ensure that the minimum_password_length parameter is configurable by a Security Administrator.

c. lists the range of values supported for the minimum_password_length parameter. The listed range shall include the value of 15.

[ST] section 6.3.3 states that passwords for the **cliadmin** and **uiadmin** accounts can be composed of any combination of upper and lower case letters, numbers, and the following special characters: !@#$%^&*()_-+={}|[]\:";'<>?,./.

The TOE implements two password policies, controlled by the `redseal.srm.strictPasswordCheck` server property. When this property is set to false (the default), the TOE enforces a minimum password length of seven characters. When the property is set to true, the TOE enforces a minimum password length of 15 characters. The property is also set to true when the **cliadmin** executes the `enable common criteria` CLI command.

2.3.2.2 Guidance Activities

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

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

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

[CCECG] Section "Set Admin Passwords" identifies the characters that may be used in passwords and guidance for the composition of strong passwords. Common Criteria compliance enforces minimum 15-character length.

2.3.2.3 Test Activities

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

The evaluator composed a set of passwords that met the TOE's requirements and contained all of the types of characters supported by the TOE (uppercase letters, lowercase letters, numbers, and selected symbols). The TOE was shown to accept those passwords.

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

The evaluator created multiple passwords which did not meet length or complexity requirements. The TOE was shown to reject all of them.

2.3.3 FIA_UIA_EXT.1 User Identification and Authentication

2.3.3.1 TSS Activities

The evaluator shall examine the TSS to determine that it describes the logon process for remote authentication mechanism (e.g. SSH public key, Web GUI password, etc.) and optional local authentication mechanisms supported by the TOE. This description shall contain information pertaining to the credentials allowed/used, any protocol transactions that take place, and what constitutes a "successful logon".

Section 6.3.1 of [ST] states that in order to log in, the user must provide an identity and also authentication data that matches that identity configured on the TOE. Users logging on to **cliadmin** remotely via SSH can be configured to authenticate with either a password-based mechanism or a public key. Users accessing the Web UI can be defined by the Subject DN associated with their smart card certificate for certificate-based authentication or can be configured to authenticate with a password-based mechanism.

The TOE supports the following logon methods:

- Remote connection to the CLI via SSH v2, using username and password
- Remote connection to the CLI via SSH v2, using username and public key
- Remote connection via the Java client, using username and password—the user launches the Java application on their local workstation and provides the following information:

- Host Address—IP address or host name of the RedSeal server to connect to
- Port—access port for the RedSeal server (3825 by default)
- User name—RedSeal user account ID Password—password for the entered RedSeal account

- Remote connection to the TOE's Remote client (i.e. the browser-based Java implementation of the Java client) , using username and password—the user navigates to the /rc/ui subdirectory of the TOE's web server root in a web browser. The user is presented with the same login prompt as with the Java client.
- Remote connection to the TOE's Web Beta client, using username and password—the user navigates to the /redseal/a/login subdirectory of the TOE's web server root in a web browser. The user is presented with a login prompt to enter their username and password.

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

Section 6.3.1 of [ST] identifies that the only functionality the TOE will perform without authentication is to display the warning banner or respond to an ICMP request.

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, 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.

This evaluation activity is not applicable to the TOE because the TOE is not distributed.

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

This evaluation activity is not applicable to the TOE because the TOE is not distributed.

2.3.3.2 Guidance Activities

The evaluator shall examine the guidance documentation to determine that any necessary preparatory steps (e.g., establishing credential material such as pre- shared keys, tunnels, certificates, etc.) to logging in are described. For each supported the login method, the evaluator shall ensure the guidance documentation provides clear instructions for

successfully logging on. If configuration is necessary to ensure the services provided before login are limited, the evaluator shall determine that the guidance documentation provides sufficient instruction on limiting the allowed services.

[CCECG] Section "Master command to enable all required settings" states that the `enable common criteria` command enables all settings required for Common Criteria compliance.

[CCECG] The CCECG Section "Logging in to the Java client/Remote client" provides the instructions to logon to the Java Client.

- The CCECG points to [ADMIN] for information on how to install and log in to the Java client, see the [ADMIN] RedSeal Installation and Administration Guide, RedSeal Clients -> Java client.

[CCECG] Section "Logging in to the Web Beta UI" provides the instructions to log into the Web UI.

[CCECG] Section "Logging in to the command line interface" provides the instructions for logging into Command Line Interface (CLI) using the administrator password of public key.

2.3.3.3 Test Activities

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

Test 1: The evaluator shall use the guidance documentation to configure the appropriate credential supported for the login method. For all combinations of supported credentials and login methods, 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.

Password based authentication was demonstrated in FIA_AFL.1 tests 1 and 2.

Public key authentication was demonstrated by attempting to open two connections with valid key pairs. One key pair was trusted by the TOE while the other was not. The TOE accepted the trusted one and rejected the untrusted one.

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

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

The evaluator demonstrated that the TOE would either ignore or respond to pings depending on how they were configured.

An NMAP scan was performed to demonstrate that only ports corresponding to claimed services were open.

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.

N/A. The TOE does not support local access.

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

N/A. The TOE is not distributed.

2.3.4 FIA_X509_EXT.1/Rev X.509 Certificate Validation

2.3.4.1 TSS Activities

The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected).

Section 6.3.4 of [ST] states that the TOE performs RFC 5280 certificate validation and certificate path validation on all X.509 certificates presented to it for the purpose of TLS server authentication. It also performs validation on the chain of certificates that are loaded into the TOE's trust store. The TOE supports a path length of at least three certificates.

The TOE validates a certification path by ensuring the presence of the basicConstraints extension with the CA flag set to TRUE for all CA certificates. The TOE will not treat a certificate as a CA certificate if the basicConstraints extension is not present or the CA flag is not set to TRUE. The certification path terminates with a trusted CA certificate designated as the Root CA.

The TOE validates X.509 certificates using the path validation algorithm defined in RFC 5280.

The TOE uses the following rules for validating the extendedKeyUsage field:

- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

Certificates are not used for trusted updates or executable code integrity, nor does the TOE support TLS mutual authentication. Therefore, the TOE does not support the rules for validating certificates with the Code Signing or Client Authentication purpose in the extendedKeyUsage field, and this part of the requirement is trivially satisfied.

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

[ST] section 6.3.4 states that the certificate chain is validated to the root, and a revocation check is performed on each certificate (except the root certificate) using OCSP.

The TOE downloads and caches OCSP status information for every CA listed in the trusted CA list of the TOE. The OCSP status is cached for the 'next update time' that is configured on the OCSP responder. The TOE uses this received value as the cache time. OCSP responders can also be configured for other external devices if someone decides to use it. The TOE uses a hard coded 1 hour as next update time (cached time) in this case. Caching only applies to validated certificates; if a TOE never validated a certificate, the TOE cache does not store the OCSP information for the issuing CA. To use OCSP for verifying the revocation status of certificates, you must configure the TOE to access an OCSP responder (server). The entity that manages the OCSP responder can be a third-party certificate authority (CA) or, if your enterprise has its own PKI, the TOE itself.

2.3.4.2 Guidance Activities

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

Section "Configure a syslog server" of the [CCECG] states that the X.509 certificate of the remote syslog server is checked when the certificate is presented to the TOE.

Section "Certificate Management Process" of the [CCECG] states that the chain validity of certificates imported to the TOE is verified when the certificate is attempted to be imported.

Section "Evaluated configuration" of the [CCECG] describes how the certificate chain of presented certificates is validated.

Section "Evaluated configuration" of the [CCECG] describes how OCSP is used for certificate validity checking.

Section "Evaluated configuration" of the [CCECG] also states that server and client certificates are checked for the appropriate value in the extendedKeyUsage field.

2.3.4.3 Test Activities

The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step or when performing trusted updates (if FPT_TUD_EXT.2 is selected). It is expected that either OCSP or CRL revocation checking is performed when a certificate is presented to the TOE (e.g. during authentication). The

evaluator shall perform the following tests for FIA_X509_EXT.1/Rev. These tests must be repeated for each distinct security function that utilizes X.509v3 certificates. For example, if the TOE implements certificate-based authentication with IPSEC and TLS, then it shall be tested with each of these protocols:

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

The evaluator configured a certificate chain consisting of a root CA, a top intermediate CA signed by the root CA, a bottom intermediate CA signed by the top intermediate CA, and a leaf certificate signed by the intermediate CA representing a test TLS server. The evaluator used the TOE to establish a connection with the TLS server and observed that the connection was successfully completed.

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

The evaluator configured the test TLS server to present the same certificate chain as configured in Test 1a, but omitted the "top" intermediate CA, breaking the chain between the leaf and root CAs. The evaluator verified that the TOE failed to validate the certificate and rejected the connection.

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

The evaluator configured a test TLS server to present the TOE with a certificate which had expired, along with the necessary intermediate certificates to complete the chain to the trusted root CA. The evaluator verified that the TOE failed to validate the certificate and rejected the connection.

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 shall then attempt the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the

certificate is no longer valid that the validation function fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore, the revoked certificate(s) used for testing shall not be a trust anchor.

The evaluator configured the OCSP responder to provide a status response of good for both the leaf and intermediate CA certificates. The evaluator caused the TOE to initiate a connection to the syslog server which was configured to present a certificate carrying OCSP revocation information. The evaluator verified the TOE accepted the connection.

Next the evaluator configured the OCSP responder to provide a status response of good for the leaf certificate and revoked for the intermediate CA certificate. The evaluator caused the TOE to initiate a connection to the syslog server which was configured to present a certificate carrying OCSP revocation information. The evaluator verified the TOE rejected the connection.

Finally the evaluator configured the OCSP responder to provide a status response of revoked for the leaf certificate and good for the intermediate CA certificate. The evaluator caused the TOE to initiate a connection to the syslog server which was configured to present a certificate carrying OCSP revocation information. The evaluator verified the TOE rejected the connection.

Test 4a: [conditional] If OCSP is selected, the evaluator shall configure an authorized responder or use a man-in-the-middle tool to use a delegated OCSP signing authority to respond to the TOE's OCSP request. The resulting positive OCSP response (certStatus: good (0)) shall be signed by an otherwise valid and trusted certificate with the extendedKeyUsage extension that does not contain the OCSPSigning (OID 1.3.6.1.5.5.7.3.9). The evaluator shall verify that the TSF does not successfully complete the revocation check.

Note: Per RFC 6960 Section 4.2.2.2, the OCSP signature authority is delegated when the CA who issued the certificate in question is NOT used to sign OCSP responses.

The evaluator configured one of the OCSP responders to sign responses with a delegated certificate for OCSP signing that lacks the OCSP Signing EKU and verified the TOE rejected responses which are signed by the certificate that lacks the OCSP Signing EKU.

Test 4b: [conditional] If CRL is selected, the evaluator shall present an otherwise valid CRL signed by a trusted certificate that does not have the cRLsign key usage bit set and verify that validation of the CRL fails.

The TOE does not claim CRL support. Therefore, this test is not applicable.

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

The evaluator configured a proprietary testing tool to modify one of the first eight bytes of the certificate and to start a TLS server utilizing this modified certificate.

The evaluator had the TOE attempt to connect to this server and verified that the connection failed.

Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC 5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

The evaluator configured a proprietary testing tool to modify the last byte of the certificate and to start a TLS server utilizing this modified certificate. The last byte is normally part of the certificate Signature value.

The evaluator had the TOE attempt to connect to this server and verified that the connection failed.

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

The evaluator configured a proprietary testing tool to modify the public key of the certificate and to start a TLS server utilizing this modified certificate.

The evaluator had the TOE attempt to connect to this server and verified that the connection failed.

Test 8: (Conditional on support for EC certificates as indicated in FCS_COP.1/SigGen). The following tests are run when a minimum certificate path length of three certificates is implemented:

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

The evaluator attempted to load an ECDSA chain into the TOE's trust store for syslog functionality. The evaluator observed that the TOE did not permit any ECDSA certificates to be imported to the TOE's trust store for the syslog functionality. The evaluator examined the ST claims and found that the only TLS ciphers used by the TOEs client functionality are RSA ciphers. The evaluator then examined the ST to determine where ECDSA signatures would be used by the TOE and concluded that they are only used in the context of SSH connections which are not using X.509 Certificates. Therefore, the evaluator has concluded that this test is passing.

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

a trusted CA certificate), where the intermediate certificate in the certificate chain uses an explicit format version of the Elliptic Curve parameters in the public key information field, and is signed by the trusted EC root CA, but having no other changes. The evaluator shall confirm the TOE treats the certificate as invalid.

The evaluator attempted to load an ECDSA chain into the TOE's trust store for syslog functionality. The evaluator observed that the TOE did not permit any ECDSA certificates to be imported to the TOE's trust store for the syslog functionality. The evaluator examined the ST claims and found that the only TLS ciphers used by the TOEs client functionality are RSA ciphers. The evaluator then examined the ST to determine where ECDSA signatures would be used by the TOE and concluded that they are only used in the context of SSH connections which are not using X.509 Certificates. Therefore, the evaluator has concluded that this test is passing.

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

The evaluator attempted to load an ECDSA chain into the TOE's trust store for syslog functionality. The evaluator observed that the TOE did not permit any ECDSA certificates to be imported to the TOE's trust store for the syslog functionality. The evaluator examined the ST claims and found that the only TLS ciphers used by the TOEs client functionality are RSA ciphers. The evaluator then examined the ST to determine where ECDSA signatures would be used by the TOE and concluded that they are only used in the context of SSH connections which are not using X.509 Certificates. Therefore, the evaluator has concluded that this test is passing.

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

The goal of the following tests is to verify that the TOE accepts a certificate as a CA certificate only if it has been marked as a CA certificate by using basicConstraints with the CA flag set to True (and implicitly tests that the TOE correctly parses the basicConstraints extension as part of X509v3 certificate chain validation).

For each of the following tests the evaluator shall create a chain of at least three certificates: a self-signed root CA certificate, an intermediate CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

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

The evaluator configured a certificate chain that consisted of a trusted root CA, a top intermediate ca, and a bottom intermediate CA that lacked the basicConstraints extension signed by the top intermediate CA, and the leaf certificate signed by the invalid bottom CA. The evaluator configured a test TLS server to present this certificate chain.

The evaluator had the TOE attempt to connect to this server and verified that the connection failed.

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 shall confirm 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 configured a certificate chain that consisted of a trusted root CA, a top intermediate CA, and a bottom intermediate CA that had a false basicConstraints extension signed by the top intermediate CA, and the leaf certificate signed by the invalid bottom CA. The evaluator configured a test TLS server to present this certificate chain.

The evaluator had the TOE attempt to connect to this server and verified that the connection failed.

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).

The TOE only uses certificates for TLS connections.

2.3.5 FIA_X509_EXT.2 X.509 Certificate Authentication

2.3.5.1 TSS Activities

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

Section 6.3.5 of [ST] describes the use of X.509v3 certificates as defined by RFC 5280 to support authentication of external TLS syslog servers.

The TOE uses the X.509 certificate presented in the TLS server Certificate message to authenticate the TLS server. Root CA certificates and intermediate CA certificates that may be needed as part of the validation are stored in the TOE's trust store. The administrative guidance provides instructions for uploading root and intermediate CA certificates to the TOE's trust store.

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

Section 6.3.5 of [ST] states that the TOE uses OCSP to determine the revocation status of X.509 certificates. If the TOE is unable to establish a connection to the OCSP responder in order to determine the validity of a certificate, it will drop the connection.

2.3.5.2 Guidance Activities

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

[CCECG] Section "Certificate Management" provides the instructions to configure and provide TLS communication with client applications and web browsers.

[CCECG} Section "Evaluated configuration" states that the execution of the `enable common criteria` command will enforce TLS and HTTPS certificate validation.

2.3.5.3 Test Activities

The evaluator shall perform the following test for each trusted channel: a. Test 1: The evaluator shall demonstrate that using a valid certificate that requires certificate validation checking to be performed in at least some part by communicating with a non-TOE IT entity. The evaluator shall then manipulate the environment so that the TOE is unable to verify the validity of the certificate and observe that the action selected in FIA_X509_EXT.2.2 is performed. If the selected action is administrator-configurable, then the evaluator shall follow the guidance documentation to determine that all supported administrator-configurable options behave in their documented manner.

The evaluator turned one of the revocation providers off and verified the TOE did not accept the connections to the peer when the revocation status of the provided by the non-available provider could not be procured.

2.3.6 FIA_X509_EXT.3 X.509 Certificate Requests

2.3.6.1 TSS Activities

If the ST author selects "device-specific information", the evaluator shall verify that the TSS contains a description of the device-specific fields used in certificate requests.

[ST] does not select “device-specific information” so this evaluation activity is N/A to the TOE.

2.3.6.2 Guidance Activities

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

[CCECG] Section “Certificate Management” provides the instructions to configure and provide TLS communication with client applications and web browsers.

[CCECG] Section “Certificate Management”, subsection “Additional Configuration” provides the instructions for “Change the Default Common Name" for CN configuration”. The section points to [ADMIN] section “Change the default common name”.

2.3.6.3 Test Activities

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.

The evaluator had the TOE generate a certificate request and verified that the CSR message was formatted correctly.

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

The evaluator demonstrated that the TOE would accept a CSR signed by a CA in the TOE’s trust store. A CSR signed by an untrusted CA was rejected.

2.4 Security Management (FMT)

General requirements for distributed TOEs

TSS

For distributed TOEs, the evaluator shall verify that the TSS describes how every function related to security management is realized for every TOE component and shared between different TOE components. The evaluator shall confirm that all relevant aspects of each TOE component are covered by the FMT SFRs.

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

General requirements for distributed TOEs

Guidance Documentation

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

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

General requirements for distributed TOEs

Tests

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

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

2.4.1 FMT_MOF.1/Functions Management of Security Functions Behaviour

2.4.1.1 TSS Activities

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

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

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

Section 6.4.3 of [ST] states that the ability to determine and modify the audit behavior is restricted to cliadmin and to uiadmin and that the ability to perform TOE updates is restricted to cliadmin only; this function cannot be performed via any interface besides the CLI.

2.4.1.2 Guidance Activities

For distributed TOEs see chapter 2.4.1.2. There are no specific requirements for non-distributed TOEs.

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

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.

[CCECG] Section "Communication with External Entities" provides the instructions to configure the TOE to export audit records to an external syslog server over TLS. RedSeal acts as a TLS client and initiates the connection to the syslog server. RedSeal identifies and authenticates the syslog server by validating the syslog server's X.509 certificate. If the connection is unintentionally broken, RedSeal restarts the connection.

[CCECG] Section "Configure a syslog server" provides the instructions to configure the syslog server.

1. Log in to the command line interface.
2. Use the `set log` command to set a syslog server.

The `remotehost` parameter `hostname` must point to a syslog server. Configuring the `hostname` parameter with the FQDN will establish the reference identifier. Either primary or secondary must be specified, followed by the hostname (FQDN) for the log server, or the reset command argument.

[CCECG] Section "set log command details" indicates that messages continue to be logged locally on the appliance.

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

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

2.4.1.3 Test Activities

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 perform the following tests:

Test 1: 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.

This is not applicable as the TOE does not make the 'transmission of audit data to external IT entity' in the second selection.

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 perform the following tests:

Test 2: 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.

This is not applicable as the TOE does not make the 'transmission of audit data to external IT entity' in the second selection.

If 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection

The evaluator shall perform the following tests:

Test 1: 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.1.4, FAU_STG_EXT.1.5 and FAU_STG_EXT.2.

The evaluator verified that users without authentication as a security administrator cannot modify the configuration for the handling of audit data by the TOE as they cannot access the correct location for the configuration setting.

If 'handling of audit data' is selected from the second selection together with 'modify the behaviour of' in the first selection

The evaluator shall perform the following tests

Test 2: 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.1.4, FAU_STG_EXT.1.5 and FAU_STG_EXT.2.

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.

The evaluator verified that the TOE permits security administrators to modify the handling of audit data settings configured on the TOE.

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 perform the following tests:

Test 1: 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.

This is not applicable as the TOE does not make the 'audit functionality when Local Audit Storage Space is full' selection.

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 perform the following tests:

Test 2: 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.

This is not applicable as the TOE does not make the 'audit functionality when Local Audit Storage Space is full' selection.

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 perform the following tests:

Test 1: 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.

The evaluator verified that users without authentication as a security administrator cannot determine the configuration for the handling of audit data by the TOE as they cannot access the correct location for the configuration setting.

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 perform the following tests:

Test 2: 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.

The evaluator verified that the TOE permits security administrators to determine the behavior of the handling of audit behavior.

2.4.2 FMT_MOF.1/ManualUpdate Management of Functions Behavior

2.4.2.1 TSS Activities

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

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

2.4.2.2 Guidance Activities

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

[CCECG] Section "Updating RedSeal software" states that the update is verified by a digital signature.

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

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

2.4.2.3 Test Activities

Test 1: 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 observed that there is no functionality provided by the TOE prior to providing authentication and authorization information.

The evaluator attempted to use the update command as an authentication factor and verified the attempt failed.

Evidence for the Java client not being able to access the functions required for update can be seen in the evidence provided for FMT_MTD.1/CryptoKeys Test 1

Test 2: 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.

Evidence for this can be seen in FPT_TUD_EXT.1, as indicated in the test activity.

2.4.3 FMT_MTD.1/CoreData Management of TSF Data

2.4.3.1 TSS Activities

For each administrative function identified in the guidance documentation that is accessible through an interface prior to administrator log-in, the evaluator shall confirm that the TSS

details how the ability to manipulate the TSF data through these interfaces is disallowed for non-administrative users.

Section 6.4.4 of [ST] states that the ability to manage TSF data is restricted to the **cliadmin** and **uiadmin**, since the commands for managing TSF data are provided either by the CLI, which only **cliadmin** can access, and the Java client/Remote client and the Web Beta client where such commands are restricted to **uiadmin**.

Management of the SSH public keys used by the SSH Server connections are restricted to the **cliadmin**.

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

Section 6.4.4 of [ST] states that only authorized administrators have the ability to manage the TOE's trust store by uploading X.509 v3 certificates, CA certificates, and cryptographic keys (see above for commands). The **cliadmin** is able to use the create cert-request CLI command to generate a Certificate Request and upload the certificate returned by the CA in response to the certificate request. Management of the SSH public keys used by the SSH Server connections are restricted to the **cliadmin**.

2.4.3.2 Guidance Activities

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

The TOE provides the following security management functions, remotely via the CLI over SSH:

- Configure the access banner (`set banner`) [CCECG Section: "Configure Login Banners"]
- Configure the remote session inactivity time before session termination (`set session-timeout`) [CCECG Section: "Set Session inactivity lockout"]
- Update the TOE and verify TOE updates prior to installation using digital signature verification (`upload image`) [CCECG Section "Updating RedSeal software"]
- Configure local audit behavior (i.e. changes to behaviour when local audit storage space is full; changes to local audit storage size) (`set log`); [CCECG Section "Set audit log rotation parameters"]
- Configure/modify the transmission of audit data to an external IT entity (`set log`); [CCECG Section "Configure a syslog server"]

- Configure the list of TOE-provided services available before an entity is identified and authenticated, per FIA_UIA_EXT.1 (`set respond-to-ping`) [CCECG Section "Ping"]
- Manage the cryptographic keys (`upload certificates, set ntp authentication symmetric add-key`) [CCECG Section "Certificate Management"], [CCECG Section "Set NTP servers"]
- Configure the list of supported TLS ciphers (`enable cipher-suites, disable cipher-suites`)
- Re-enable an administrator account (`enable user`) [CCECG Section "Unlocking administrator accounts"]
- Set the time used for time stamps (`set date`) [CCECG Section "Set a Timestamp"]
- Configure NTP (`set ntp, set ntp authentication`) [CCECG Section "Set NTP servers"]
- Manage the TOE's trust store and designate X.509 v3 certificates as trust anchors (`upload certificate, upload ca-certificate`) [CCECG Section "Certificate Management"]
- Generate Certificate Signing Request (CSR) and process CA certificate response (`create cert-request, upload ca-certificate`) [CCECG Section "Certificate Management"]
- Configure the authentication failure parameters for FIA_AFL.1 (`set property server`) [CCECG Section "Authentication failure lockouts"]
- Manage the trusted public keys database (`add credential cliadmin`) [CCECG Section "Set Admin Passwords"]

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.

[CCECG] Section "Certificate Management" provides the commands for "Certification Configuration" to configure a trust store.

[CCECG] Section "Certificate Management" provides the commands for "Certificate Management Process" to use a CA-signed certificate.

2.4.3.3 Test Activities

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

2.4.4 FMT_MTD.1/CryptoKeys Management of TSF Data

2.4.4.1 TSS Activities

For distributed TOEs see Section 2.4.1.1.

The TOE is not a distributed TOE, therefore this portion is not applicable.

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

The evaluator examined [ST] Section 6.4.4 and found that the administrator can "use the create cert-request CLI command to generate a Certificate Request and upload the certificate returned by the CA in response to the certificate request". The evaluator has concluded that the specific action referenced is generating new keys.

2.4.4.2 Guidance Activities

For distributed TOEs see Section 2.4.1.2.

The TOE is not a distributed TOE, therefore this portion is not applicable.

For non-distributed TOEs, the evaluator shall also ensure the Guidance Documentation describes how the operations are performed on the keys the Security Administrator is able to manage.

[CCECG] section "Certificate Management" provides the instructions for how the administrator can manage the specific certificates installed on the TOE.

2.4.4.3 Test Activities

The evaluator shall perform the following tests:

Test 1: 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 attempted to manage the cryptographic keys of the TOE without prior authentication as a Security Administrator and verified the attempt fails.

Test 2: The evaluator shall try to perform at least one of the related actions with prior authentication as Security Administrator. This attempt should be successful.

The evaluator authenticated to the TOE as a Security Administrator and verified the Security Administrator could perform the management activities for the cryptographic keys.

2.4.5 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.

2.4.5.1 TSS Activities (containing also requirements on Guidance Documentation and Tests)

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

Section 6.4.2 of [ST] lists the supported management functions that can be remotely administered via the CLI over SSH. The evaluation activities for the relevant SFRs demonstrate the proper implementation of these functions.

The TOE provides the following security management functions, remotely via the CLI over SSH, :

- Configure the access banner (`set banner`)
- Configure the remote session inactivity time before session termination (`set session-timeout`)
- Update the TOE and verify TOE updates prior to installation using digital signature verification (`upload image`)

- Configure local audit behavior (i.e. changes to behaviour when local audit storage space is full; changes to local audit storage size) (`set log`);
- Configure/modify the transmission of audit data to an external IT entity (`set log`);
- Configure the list of TOE-provided services available before an entity is identified and authenticated, per FIA_UIA_EXT.1 (`set respond-to-ping`)
- Manage the cryptographic keys (`upload certificates`, `set ntp authentication symmetric add-key`)
- Configure the list of supported TLS ciphers (`enable cipher-suites`, `disable cipher-suites`)
- Re-enable an administrator account (`enable user`)
- Set the time used for time stamps (`set date`)
- Configure NTP (`set ntp`, `set ntp authentication`)
- Manage the TOE's trust store and designate X.509 v3 certificates as trust anchors (`upload certificate`, `upload ca-certificate`)
- Generate Certificate Signing Request (CSR) and process CA certificate response (`create cert-request`, `upload ca-certificate`)
- Configure the authentication failure parameters for FIA_AFL.1 (`set property server`)
- Manage the trusted public keys database (`add credential cliadmin`)

The TOE also provides the `enable common criteria` CLI command, which performs the following actions to configure the TOE consistent with the evaluated configuration:

- Enforces minimum password length of 15 characters
- Enables lock out for all users after three unsuccessful login attempts. The lockout interval for all users is set to 600 seconds (10 minutes)
- Restricts TLS to TLS v1.2 and disables all other TLS and SSL versions
- Disables disallowed cryptographic algorithms and methods for TLS and SSH communications.

Additionally, the following security management functions can also be performed by the TOE for the **uiadmin** user using the RedSeal Java client or the browser-based Remote client:

- Update the TOE and verify TOE updates prior to installation using digital signature verification (**`Admin > Server`**, select `Upload Image`)
- Configure audit behavior (**`System Settings > Logging`**)
- Manage the TOE's trust store and designate X.509 v3 certificates as trust anchors (**`System Settings > Server Certificate`**)

Lastly, the **uiadmin** user is able to manage the warning banner of the Web Beta client (**`Service Admin > Server Settings > Banne`**r). The Web Beta client is otherwise used for non-management functionality.

The evaluation team observed during testing that the TOE provides all the management functions specified in FMT_SMF.1.

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.

Section 2.2 of [ST] states the TOE provides a Command Line Interface (CLI) for management and administration. The CLI is accessible remotely via SSH only (no local access is provided by the TOE).

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.

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

(If configure local audit is selected) The evaluator shall examine the TSS and Guidance Documentation to ensure that a description of the logging implementation is described in enough detail to determine how log files are maintained on the TOE.

FMT_SMF.1 selects “Ability to configure local audit behavior (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full; changes to local audit storage size”.

Section 6.4.2 of [ST] states that the configuration of local audit behavior (i.e. changes to behaviour when local audit storage space is full; changes to local audit storage size) is configured using the `set log` command.

2.4.5.2 Guidance Activities

See Section 2.4.4.1

2.4.5.3 Test Activities

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

The evaluator verified that all of the management functions identified for the TOE have been executed during the course of the evaluation.

2.4.6 FMT_SMR.2 Restrictions on Security Roles

2.4.6.1 TSS Activities

The evaluator shall examine the TSS to determine that it details the TOE supported roles and any restrictions of the roles involving administration of the TOE (e.g. if local administrators

and remote administrators have different privileges or if several types of administrators with different privileges are supported by the TOE).

[ST] section 6.4.1 identifies two default administrator roles that together provide the capabilities of the Security Administrator role:

- **cliadmin**—performs administrative tasks using the CLI, remotely via SSH
- **uiadmin**— performs administrative tasks, remotely, using the Java client/Remote client (TLS) or Web Beta client (via HTTPS/TLS).

The **cliadmin** has full permissions, while the **uiadmin** have a subset of administrative permissions as identified by the identified management functions specified in FMT_SMF.1 [ST] section 6.4.2.

2.4.6.2 Guidance Activities

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

The ST states that the TOE does not provide local administration. The guidance for remote administration is located in [CCECG] Section “Logging in to the command line interface”, “Logging in to the Java client/Remote client” and “Logging in to the Web Beta UI”.

RedSeal provides a Command Line Interface (CLI) for management and administration. The CLI is accessible remotely via SSHv2. For remote access, the `cliadmin` initiates communication to the TSF, using SSH client software on their local workstation.

RedSeal provides a Java Client or remote client for management and administration. The Java Client is installed on a remote system and connects to the TOE over TLS. The remote client is accessed over a web browser and connects to the TOE over TLS. For remote access, the `uiadmin` initiates communication to the TSF, using the Java Client or web browser on their local workstation.

The TSF provides a remote management interface, the Web Beta Client, that a remote administrator can use to access a TSF hosted instance of the UI provided by the Java Client.

2.4.6.3 Test Activities

In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH, if the TSF shall be validated against the Functional Package for Secure Shell referenced in Section 2.2 of the cPP; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team’s test activities.

The TOE possesses three interfaces by which the TOE can be administered: SSH, remote Java client, Web Beta Client.

The SSH interface is the primary interface by which the TOE is administered.

The remote Java client has two different instances of the same interface, one is an application that is installed on a workstation in the environment, and the other is the same application served by a webpage hosted by the TOE. Both versions communicate to the TOE over a TLS channel. The evaluator verified that this interface could be used to administer the TOE.

The Web Beta Client is a web browser interface that can be accessed to perform operational use of the TOE that is outside the bounds of the evaluation. This interface is only used to set an access banner for the interface in the evaluated configuration, which the evaluator verified could be managed by the administrator.

2.5 Protection of the TSF (FPT)

2.5.1 FPT_APW_EXT.1 Protection of Administrator Passwords

2.5.1.1 TSS Activities

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

Section 6.5.1 of [ST] states that the TOE protects the passwords for the **cliadmin** and **uiadmin** by generating a hash of these passwords using PBKDF2 with SHA-512 and storing the hashed password, rather than storing the password itself or encrypting the password prior to storage. The TOE does not offer any functions that will disclose to any users a plaintext administrative password.

2.5.1.2 Guidance Activities

None

2.5.1.3 Test Activities

None

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

2.5.2.1 TSS Activities

The evaluator shall examine the TSS to determine that it details how any preshared keys, symmetric keys, and private keys are stored and that they are unable to be viewed through

any interface designed specifically for that purpose, by any enabled role, as outlined in the application note. If these values are not stored in plaintext, the TSS shall describe how they are protected/obscured.

Section 6.5.2 of [ST] states that the TOE does not offer any functions that will disclose to any users a stored cryptographic key.

Section 6.2.4 of [ST] specifically Table 11 "Private Keys, Symmetric Keys and CSP's" indicates how each of the specific keys are stored on the file system. This table indicates that the RSA Private Key is stored in an encrypted PKCS #12 keystore on the system disk, the NTP authentication keys are stored plaintext on the TOE file system, and all other keys are stored ephemerally in RAM of the system.

2.5.2.2 Guidance Activities

None

2.5.2.3 Test Activities

None

2.5.3 FPT_STM_EXT.1 Reliable Time Stamps

2.5.3.1 TSS Activities

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

Section 6.5.5 of [ST] The TOE comprises the RedSeal Server hardware appliance that includes a hardware-based real-time clock able to provide reliable time stamps for the use of the TOE. The clock is used for audit record time stamps, measuring session activity for termination, measuring the time an administrator account is locked following consecutive failed authentication attempts, and for cryptographic operations based on time/date, such as checking certificate expiry.

The cliadmin can use the show date CLI command to display the current system date, time and time zone, and can use the set date CLI command to set the system date and time to a specified value. The cliadmin can use the set timezone CLI command to set the time zone of the appliance. The default time zone is Coordinated Universal Time (UTC).

The TOE can also be configured to synchronize its time with an NTP server in the operational environment. The **cliadmin** uses the `set ntp` CLI command to configure the NTP servers to be used by the TOE.

If "obtain time from the underlying virtualization system" is selected, the evaluator shall examine the TSS to ensure that it identifies the VS interface the TOE uses to obtain time. If

there is a delay between updates to the time on the VS and updating the time on the TOE, the TSS shall identify the maximum possible delay.

The evaluation activity is not applicable. The TOE is a hardware appliance and not a virtual system.

2.5.3.2 Guidance Activities

The evaluator shall examine 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.

Section “Set a Timestamp” of [CCECG] provides the instructions to the administrator for how to set the time of the local system.

Section “Set NTP servers” of [CCECG] provides the instructions to the administrator for how configure the NTP servers. Examples are provided to set up the NTP servers using the SHA-1 authentication.

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

The ST states that “The product can be provisioned as a virtual appliance image and deployed on a virtual platform; however these virtual deployments are not being considered for evaluation.” Therefore, no guidance has been provided in the CCECG.

2.5.3.3 Test Activities

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

The evaluator queried the current time on the TOE, then attempted to change the time. The evaluator queried the time again and verified that the time successfully changed to the value specified by the evaluator.

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 shall 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.

The following evidence is from the first part of FCS_NTP_EXT.1.2. In this test, the evaluator views the current time, prior to configuring the NTP source time. After configuring the NTP and its key, the TOE updated its time

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

This test is not applicable because the TOE is not a virtualized device, thus there is no underlying VS.

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

This test is not applicable because the TOE consists of a single standalone device and therefore only has time source.

2.5.4 FPT_TST_EXT.1 TSF Testing

2.5.4.1 TSS Activities

Modified by TD0836
The evaluator shall examine the TSS to ensure that it details **each of** the self-tests that are **identified by the SFR** ~~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. If more than one failure response is listed in FPT_TST_EXT.1.2, the evaluator shall examine the TSS to ensure it clarifies which response is associated with which type of failure.

Section 6.5.3 of [ST] states that the TOE performs all self-tests on start-up. These consist of a test to verify the integrity of the TOE firmware/software and cryptographic self-tests that confirm the correct functionality of the cryptographic algorithms implemented by the OpenSSL and BSAFE cryptomodules included in the TOE.

Upon start-up, the TOE will trigger the integrity test to validate the integrity of the main firmware files and RedSeal production code. The test checks the parameters of the Java files against a SHA-512 of the file, the permission of the file, and the ownership of the file. If any of

the parameters changed, the system integrity test will fail and the administrative services (Admin server, web server and ssh server) will not startup.

Each cryptomodule performs the following cryptographic self-tests during module initialization:

- Cryptographic known answer tests—for symmetric and one-way cryptographic operations, the module will process known input data and compare it to the pre-computed output for each algorithm (AES KATs, SHA KATs, DRBG KAT, RSA Sign/Verify KAT, ECDSA Sign/Verify KAT) to ensure results are consistent with known answers.
- Pairwise consistency tests—for public key cryptographic operations, the module will perform a cryptographic operation followed by its reverse (e.g., encrypt/decrypt; sign/verify) to ensure that the result of the calculation is the same as the initially-supplied value.

The self-tests are sufficient to ensure the correct functionality of the TSF as they encompass the cryptographic functionality and the integrity of the entire TOE software/firmware executable code.

For distributed TOEs the evaluator shall examine the TSS to ensure that it details which TOE component performs which self-tests and when these selftests are run. The evaluator shall also examine the TSS to ensure it describes how the TOE reacts if one or more TOE components fail self-testing (e.g. halting and displaying an error message; failover behaviour).

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

2.5.4.2 Guidance Activities

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

[CCECG] Section “Start-up self-tests” states that the failure of any self-test will put the TOE into a maintenance mode where the Admin server, web server and ssh server will not start and either an error message is displayed (for integrity failure) or the result is logged (for cryptographic tests). If the self-tests pass the services will start normally.

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

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

2.5.4.3 Test Activities

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

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

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

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

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

The evaluator verified the TOE reported that self-tests were executed successfully for the cryptographic functions utilized by the TOE and the integrity of the TOE.

Modified by TD0836

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~~ **according to the SFR and in agreement with the descriptions in the TSS**.

The evaluator verified the TOE executed self-tests during the start-up sequence.

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

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

2.5.5 FPT_TUD_EXT.1 Trusted Update

2.5.5.1 TSS Activities

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

Section 6.5.4 of [ST] identifies the CLI command that is used to show the current software version.

The TOE can hold up to three executable images in its file system. One image is designated the “Current” image and one image (which can be the same as the Current image) is designated the “Next” image. The Current image is the image that is currently executing on the TOE, while the Next image is the image that will be loaded for execution at the next reboot of the TOE.

The **cliadmin** can use the show images CLI command to display the currently active version of the TOE, and can use the `set next image` CLI command to specify which image will be loaded for execution (and therefore become the Current image) at the next reboot. The

cliadmin uses the `upload image` CLI command to upload an image to the TOE. An uploaded image is designated the Next image (though this can be changed by the **cliadmin** using the set next image CLI command).

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. The evaluator shall verify that the TSS describes the method by which the digital signature is verified to include how the candidate updates are obtained, the processing associated with verifying the digital signature of the update, and the actions that take place for both successful and unsuccessful signature verification.

Section 6.5.4 of [ST] states that the entire update package is hashed and signed by RedSeal using SHA256 algorithm and RSA 4096 bit. Customers download images from the support portal and store them on their local filesystem or an HTTPS or SFTP server accessible from the appliance to be upgraded. The **cliadmin** uses the `upload image` CLI command to upload the new image to the appliance. The `upload image` CLI command copies the image and verifies the digital signature of the image after the upload is complete. If the verification is successful, the TOE denotes the uploaded image as the Next image. The Next image becomes the Current image at the next reboot. That is, a reboot initiates the update process. If verification fails, the TOE rejects the image and does not install it. The **cliadmin** is then able to `exit` the command and the TOE will automatically delete the unverified image.

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.

[ST] does not claim automatic checking or application of updates so this activity is N/A.

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 shall examine the guidance documentation instead.

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

2.5.5.2 Guidance Activities

The evaluator shall verify that the guidance documentation describes how to query the currently active version. If a trusted update can be installed on the TOE with a delayed

activation, the guidance documentation shall describe how to query the loaded but inactive version.

[CCECG] Section “Updating RedSeal software” how to query the currently active TOE version. The execution of the `show images` command will display all of the stored images and find out which image is currently installed. The `show images` command will display 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). The description shall include the procedures for successful and unsuccessful verification. The description shall correspond to the description in the TSS.

[CCECG] section “Updating RedSeal software” states that the TOE implements the use of a digital signature for verification of the trusted update. If the TOE successfully verifies the digital signature of the provided update the update is applied to the TOE, if the TOE does not successfully verify the digital signature the installation attempt is terminated and the previous executing version will remain the current version.

It was observed that the [CCECG] section “Updating RedSeal software” under the specific procedures for the update indicates in step 4 that the TOE will prompt for a published hash. This functionality is an optional additional validation and published hash functionality is not claimed; therefore, the functionality of the published hash validation was not examined during the evaluation.

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

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

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.

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

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 shall also ensure that the Guidance Documentation describes how the certificates are installed/updated/selected, if necessary.

[ST] does not claim the use of a certificate-based mechanism for software updates so this evaluation activity is N/A.

2.5.5.3 Test Activities

Test 1: The evaluator shall perform 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 shall verify 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 shall perform the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

The evaluator observed the TOE's current version, executed an update of the TOE, and then observed the current version again prior to performing a reboot and verified the reported the currently executing version of the TOE and the next version which would be applied upon reboot of the TOE. The evaluator rebooted the TOE and waited for the reboot to be completed. Once the TOE completed the reboot the evaluator verified that the current version indicated as the new version which was installed.

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 shall first confirm that no updates are pending and then perform 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 shall obtain or produce illegitimate updates as defined below and attempt to install them on the TOE. The evaluator shall verify that the TOE rejects all of the illegitimate updates. The evaluator shall perform this test using all of the following forms of illegitimate updates:

i. A modified version (e.g. using a hex editor) of a legitimately signed update
ii. An image that has not been signed
iii. 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)
iv. 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 the current version and most recently installed version, reflect the same version information as prior to the update attempt.

The evaluator attempted to update the TOE with software images that were modified, unsigned, and had an invalid/modified signature, and confirmed that the TOE did not successfully update.

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

The TOE only supports one method of update; thus this portion of the activity has been implicitly met.

For distributed TOEs the evaluator shall perform Test 1 and Test 2 for all TOE components.

This portion is not applicable as the TOE is not a distributed TOE.

2.6 TOE Access (FTA)

2.6.1 FTA_SSL.3 TSF-initiated Termination

2.6.1.1 TSS Activities

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

[ST] section 6.6.2 describes the session termination behavior. The TOE can be configured to terminate remote interactive sessions after a period of inactivity. The **cliadmin** can use the set `session-timeout` CLI command to configure the session idle timeout value for the remote CLI sessions as a number of minutes. The default value of the CLI session idle timeout is `infinite`, meaning the TSF-initiated termination of inactive remote interactive CLI sessions is disabled by default This must be configured in the evaluated configuration to specify a maximum value for session timeout . The **cliadmin** can enable the session idle timeout mechanism for the Remote client, and Web Beta client by using the set property command to set the `redseal.srm.https.sessionTimeout` property to true. When this property is set to true, Remote/Web Beta client sessions will be terminated after a period of inactivity as configured by the `set session-timeout` command. Session idle timeout for the Java client/remote client is configured by **uiadmin** under System Settings > Timeout.

2.6.1.2 Guidance Activities

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

[CCECG] Section “Set Session inactivity lockout” provides the guidance to configure the inactivity time period for remote administrative session termination. Specifically the instructions for this are presented in the sections “Access RedSeal through the CLI” and “Access RedSeal through the Java Client”.

2.6.1.3 Test Activities

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

Test 1: The evaluator shall follow the guidance documentation to configure several different values for the inactivity time period referenced in the component. For each period configured, the evaluator shall establish a remote interactive session with the TOE. The evaluator shall then observe that the session is terminated after the configured time period.

The evaluator configured the TOE to have various inactivity timeout values. The evaluator then authenticated to the TOE and left the session inactive. The evaluator verified when the configured inactivity period elapsed the user was logged out and had to re-authenticate to the TOE. This test was performed for all of the TOE's interfaces.

2.6.2 FTA_SSL.4 User-initiated Termination

2.6.2.1 TSS Activities

The evaluator shall examine the TSS to determine that it details how the remote administrative session (and if applicable the local administrative session) are terminated.

[ST] section 6.6.2 states that Administrators are able to terminate their own interactive sessions by logging out of the TOE. The **cliadmin** logs out of an interactive CLI session by entering the `exit` command. The **uiadmin** logs out of a Java client session by clicking on **File > Exit** or by closing the Java client application. The **uiadmin** logs out of a Legacy/Web Beta client session by clicking on "Log out".

2.6.2.2 Guidance Activities

The evaluator shall confirm that the guidance documentation states how to terminate a remote interactive session (and if applicable the local administrative session).

[CCECG] Section "Logging in to the command line interface" states the `exit` command is used to terminate an interactive CLI session.

[CCECG] Section "Exiting from the Web UI" states to log out of the Web UI, click your username in the upper-right corner of the interface and select `Log Out` from the menu.

[CCECG] Section "Exiting from the Java client" states to exit the Java client, select `Exit` from the `File` menu, or click the X in the upper right corner of the user interface.

2.6.2.3 Test Activities

Test 1 [conditional]: If the TOE supports local administration, the evaluator shall initiate an interactive local session with the TOE. The evaluator shall then follow the guidance documentation to exit or log off the session and observes that the session has been terminated.

This test is not applicable as the TOE does not support local administration of the TOE.

Test 2: For each method of remote administration, the evaluator shall initiate an interactive remote session with the TOE. The evaluator shall then follow the guidance documentation to exit or log off the session and observes that the session has been terminated.

The evaluator logged in using the SSH connection channel, then used the "exit" command to log out of the TOE.

The evaluator logged in to the TOE using the Java client, then used the "exit" in the file menu or simply closed the Java client application itself to log out of the TOE.

The evaluator logged in to the TOE using the web UI, then used the "log out" from the menu to log out of the TOE.

In each scenario the evaluator confirmed the user was logged out and needed to re-authenticate to regain access to the TOE.

2.6.3 FTA_TAB.1 Default TOE Access Banners

2.6.3.1 TSS Activities

The evaluator shall check the TSS to ensure that it details each administrative method of access (local and/or remote) available to the Security Administrator (e.g. serial port, SSH, HTTPS). The evaluator shall check the TSS to ensure that all administrative methods of access available to the Security Administrator are listed and that the TSS states that the TOE is displaying an advisory notice and a consent warning message for each administrative method of access. The advisory notice and the consent warning message might be different for different administrative methods of access and might be configured during initial configuration (e.g. via configuration file).

Section 6.6 of [ST] states that The following methods of administrative access to the TOE are available to the security administrator:

- Remote access to the CLI via SSH v2, restricted to **cliadmin**
- Remote access to the Web Beta client via browser, restricted to **uiadmin** and Web Beta client users with Admin permissions.
- Remote access via the Java client (via standalone Java application or via browser aka "Remote client"), restricted to **uiadmin**

Section 6.6.1 of [ST] states that the **cliadmin** can use the `set banner` CLI command to configure an advisory notice and consent warning message to be displayed to the user prior to establishment of a CLI session. The `pre-authentication` option ensures the message is displayed after the user enters a username but before the user is prompted to enter a password. On the Java client/Remote client, **uiadmin** can configure the banner for that interface under **`System Settings > Show/Hide`**. On the Web Beta client, **uiadmin** can configure the banner for that interface under **`Service Administration > Server Settings > Banner`**.

2.6.3.2 Guidance Activities

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

[CCECG] Section "Configure Login Banners" provides the guidance to configure the Web UI, the Java Client and the CLI login banner message.

2.6.3.3 Test Activities

Test 1: The evaluator shall follow the guidance documentation to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

The evaluator configured the TOE access banner and verified the configured banner was displayed prior to authentication. This was repeated for all interfaces.

2.7 Trusted Path/Channels (FTP)

2.7.1 FTP_ITC.1 Inter-TSF Trusted Channel

2.7.1.1 TSS Activities

The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

Section 6.7 of [ST] identifies the trusted channels used by the TOE. The TOE communicates with the following authorized IT entities:

- Audit server—the TOE can be configured to export its audit records to an external syslog server over TLS. In this case, the TOE acts as a TLS client and initiates the connection to the syslog server. The TOE identifies and authenticates the syslog server by validating the syslog server's X.509 certificate that is presented during the TLS negotiation.

2.7.1.2 Guidance Activities

The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

[CCECG] Section "Evaluated configuration" identifies the protocols and parameters that are used in secure communications as configured by the `enable common criteria` command.

[CCECG] Section “Communication with External Entities” identifies the following external entities during operations. Allowed protocols for communication are set using the `enable common criteria` master command.

- Audit server—RedSeal can be configured to export audit records to an external syslog server over TLS. RedSeal acts as a TLS client and initiates the connection to the syslog server. RedSeal identifies and authenticates the syslog server by validating the syslog server’s X.509 certificate. If the connection is unintentionally broken, RedSeal restarts the connection.

    Guidance to configure the syslog server is identified in [CCECG] Section “Configure a syslog server” provides the instructions using the CLI `set log` command to configure the syslog server.

2.7.1.3 Test Activities

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

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

The evaluator used the guidance to set up a TLS syslog connection from the TOE to the TLS server. The result of that led to a successful encrypted tunnel.

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

The following evidence can be seen in FTP_ITC.1 Test 4. In that test, the evaluator reviewed the packet capture for the connection for rsyslog tunneling and observed that the communications channels can be initiated from the TOE.

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

The evidence for this can be seen in FTP_ITC.1 Test 4. In that test, the evaluator reviewed the packet capture for the TLSC connection for rsyslog tunneling and observed that the Application Data were not sent in plaintext text.

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, 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 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 TOE external interruption (such as a cable being physically removed or a virtual connection being disabled), another network device shall be used to interrupt the connection between the TOE and the distinct IT entity. The interruption shall be external to the TOE (i.e., by manipulating the test environment and not by TOE configuration change).

The evaluator interrupted the connection between the TOE and the remote syslog server at a core switch for a short period of time, ~10 seconds, and then reconnected the connection. The evaluator observed that after the connection was re-established the traffic from the TOE to the remote syslog server was sent in a protected and encrypted manner.

Next, the evaluator interrupted the connection between the TOE and the remote syslog server at a core switch for a longer period of time, ~30 minutes, and then reconnected the connection. The evaluator observed that after the connection was re-established the traffic from the TOE to the remote syslog server was sent in a protected and encrypted manner

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

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.

2.7.2 FTP_TRP.1/Admin Trusted Path

2.7.2.1 TSS Activities

The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Section 6.7 of [ST] states that the TOE supports the following remote access methods:

- Remote access to the CLI via SSH v2, with the TOE acting as the SSH server. This is available to the single CLI user account, **cliadmin**. The **cliadmin** initiates communication to the TSF, using SSH client software on their local workstation. The TSF authenticates the **cliadmin** (either using password or public key) and maintains the trusted path for all remote administration actions until the **cliadmin** terminates the session by exiting the CLI.
- RedSeal Java client/Remote client—the TOE communicates with the RedSeal Java client or Remote client, which provide user-level access to the data modeling and analysis functions of the RedSeal Server, as well as some administrative capabilities. Communication between the Java client and the TOE is via RMI over TLS, with the TOE acting as the TLS server. The TOE identifies and authenticates the Java client by identifying and authenticating the user that attempts to login to the TOE via the Java client. The Remote client is identical in function and presentation to the Java client and uses the same logical interface to access the TOE, except that it is spawned in a web browser process rather than being run as a standalone executable.
- Remote access to the browser-based Web Beta client—**uiadmin** use a browser to communicate with the TOE's Web Beta client. Communication between the Web Beta client and the user's browser is via HTTPS, with the TOE acting as the HTTPS/TLS server. The TOE authenticates the user using password and maintains the trusted path for all remote administration actions until the **uiadmin** terminates the session by exiting the Web Beta client.

2.7.2.2 Guidance Activities

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

The [CCECG] contains instructions for establishing the remote administrative sessions.

- Remote access to the CLI via SSH v2, with the TOE acting as the SSH server.

  [CCECG] Section "Logging in to the command line interface" provides instructions for logging into the CLI using SSH.

- RedSeal Java client—the TOE communicates with the RedSeal Java client, which provides user-level access to the data modeling and analysis functions of the RedSeal Server, as well as some administrative capabilities. Communications between the Java client and RedSeal are protected by TLS.

  [CCECG] Section "Logging in to the Java client/Remote client" provides instructions for logging into the Java Client. [CCECG] Section "Master command to enable all required settings" identifies the `enable common criteria` command to enable all settings required for Common Criteria compliance.

- Remote access to the browser-based Web UI - The Web UI is within the TOE boundary and is the primary interface for administrative and operational tasks. Communications between the Web UI and RedSeal are protected by TLS.

[CCECG] Section “Logging in to the Web Beta UI” provides the instructions for logging onto the Web UI. [CCECG] Section “Master command to enable all required settings” identifies the `enable common criteria` command to enable all settings required for Common Criteria compliance.

2.7.2.3 Test Activities

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.

Throughout the course of the testing effort, the evaluator used each interface to initiate a connection to the management interface of the TOE to carry out a particular task.

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

The evaluator accessed each of the administrative interfaces and verified that the communication was encrypted across both HTTP/TLS and SSH interface.

Further assurance activities are associated with the specific protocols. For distributed TOEs the evaluator shall perform tests on all TOE components according to the mapping of trusted paths to TOE components in the Security Target.

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

3 Security Assurance Requirements

3.1 Class ASE: Security Target Evaluation

General ASE

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

3.1.1 ASE_TSS.1 TOE Summary Specification for Distributed TOEs

This section is N/A for this evaluation because the TOE is not distributed.

3.2 Class ADV: Development

3.2.1 ADV_FSP.1 Basic Functional Specification

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

3.2.1.1 ADV_FSP.1 Evaluation Activity

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

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or

performing updates). Explicitly labeling TSFI as security relevant or non-security relevant is not necessary. A TSFI is implicitly security relevant if it is used to satisfy an evaluation activity, or if it is identified in the ST or guidance documentation as adhering to the security policies (as presented in the SFRs). The intent is that these interfaces will be adequately tested and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied. According to the description above 'security relevant' corresponds to the combination of 'SFR-enforcing' and 'SFR-supporting' as defined in CC Part 3, paragraph 224 and 225.

The set of TSFI that are provided as evaluation evidence are contained in the Security Target and the guidance documentation.

Section 2.2.1 of [ST] identifies the security relevant TSFIs as remote syslog server, Web UI, NTP Server, and workstation (SSHv2 client). The TSS describes these logical interfaces as TLS trusted channels and SSH trusted path and defines their operation in terms of the relevant FCS and FTP requirements. Section 2.3 of [ST] also defines the external physical interfaces of the TOE in sufficient detail to determine their security relevance. The power supply interface is obviously non-TSFI based on its intended use).

3.2.1.2 ADV_FSP.1 Evaluation Activity

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

The vendor developed [CCECG] specifically to address the security functionality as identified in [ST]. The Guidance activities for the individual SFRs demonstrate that the guidance documentation includes sufficiently detailed instructions to configure and use the TSFIs in the manner required by [ST]. This is demonstrated by the evaluation activities for FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSS_EXT.1, and FMT_MTD.1/CoreData.

3.2.1.3 ADV_FSP.1 Evaluation Activity

The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

The evaluator shall use the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 2, including the EAs associated with testing of the interfaces.

It should be noted that there may be some SFRs that do not have a TSFI that is explicitly “mapped” to invoke the desired functionality. For example, generating a random bit string or destroying a cryptographic key that is no longer needed are capabilities that may be specified in SFRs, but are not invoked by an interface.

The required EAs define the design and interface information required to meet ADV_FSP.1. If the evaluator is unable to perform some EA, then the evaluator shall conclude that an adequate functional specification has not been provided.

Section 6.7 of [ST] associates the TOE's remote logical interfaces with the FTP_ITC.1 and FTP_TRP.1/Admin SFRs. Additionally, this section identifies the trusted channel protocol and which end of the connection the TOE is (client or server) as needed to specifically associate each logical interface further with FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, or FCS_TLSS_EXT.1 as appropriate.

Additionally, the intended usage of each logical interface can be inferred from the TSS to the extent that their applicability to other SFRs can be determined. Specifically, the syslog interface is also used in support of FAU_STG_EXT.1 and the management interface is used to enforce the various FMT requirements and supports the enforcement of the various FIA and FTA requirements through its usage.

3.3 Class AGD: Guidance Documents

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

Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in Appendix B.4.2.1.

3.3.1 AGD_OPE.1 Operational User Guidance

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR. For the related evaluation activities, the evaluation evidence documents Security Target, AGD documentation (user guidance) and functional specification documentation (if provided) shall be used as input documents. Each input document is subject to ALC_CMS.1-2 requirements.

In addition, the evaluator performs the EAs specified below.

3.3.1.1 AGD_OPE.1 Evaluation Activity

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

The [CCECG] is published with the Security Target at the https://www.niap-ccevs.org/ website. Section "Documentation References" of [CCECG] lists other documentation comprising [ADMIN] referenced from [CCECG] that the vendor publishes on its web site. This includes URLs for each document.

3.3.1.2 AGD_OPE.1 Evaluation Activity

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

According to [ST], the TOE solely refers to the RedSeal Server v10.5 appliance. [CCECG] (in section "Documentation References") and [Admin] refer to the RedSeal Server.

[ST] section 2.3 lists the supported Operational Environment components (syslog server, SSH client, RedSeal Client, Remote Administrator Workstation, NTP Server, and OCSP Responder). [CCECG] describes support for these components and configuration of their interactions with the TOE, and does not describe any external components or interfaces outside the scope of what [ST] claims.

3.3.1.3 AGD_OPE.1 Evaluation Activity

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

Section "Master command to enable all required settings" of [CCECG] describes how to `enable common criteria` mode and explicitly states this is required by the TOE.

3.3.1.4 AGD_OPE.1 Evaluation Activity

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

The introduction of the [CCECG], the guidance documentation, includes a warning to the reader that only the security functionality claimed by the ST has been addressed by the evaluated configuration.

3.3.1.5 AGD_OPE.1 Evaluation Activity

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

a. The guidance documentation shall contain instructions for configuring any cryptographic implementation associated with the evaluated configuration of the TOE. It shall provide a warning to the administrator that use of other cryptographic implementations was not evaluated nor tested during the CC evaluation of the TOE.
b. The evaluator shall verify that this process includes 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).
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.

Section 3.3.1.3 above addresses part a).

For part b), Section "Updating RedSeal software" of [CCECG] describes the update process. Specifically, administrators use the TOE to check for updates made available on the support portal at (https://www.redseal.net/services/customer-support) site. Upload the RedSeal image file from the location where it was saved: upload image sftp://username:password//home/image<version>.enc

Sections "Updating RedSeal software", and "Procedure" of [CCECG] states that If the integrity validation fails, the installation will not proceed. Likewise, if the signature verification fails, the installation is terminated. In this case the user will continue using current version. If it passes, the installation begins.

Section 3.3.1.4 above addresses part c).

3.3.2 AGD_PRE.1 Preparative Procedures

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

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

In addition, the evaluator performs the EAs specified below.

3.3.2.1 AGD_PRE.1 Evaluation Activity

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

The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

The evaluators reviewed [Admin] and [CCECG] and determined that they are written at a level that is appropriate for the intended audience. These guides relate to the logical setup and operational administration of the TOE.

3.3.2.2 AGD_PRE.1 Evaluation Activity

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

The evaluators observed that [CCECG] and [Admin] identify the same TOE model that is claimed in [ST]. While conducting the review of the [CCECG] against the claimed SFRs, the evaluators observed that all environmental components were discussed.

The TOE is consistently identified as version 10.5.2.

3.3.2.3 AGD_PRE.1 Evaluation Activity

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

The evaluators reviewed [Admin] and [CCECG] and determined that they describe how set up the TOE's logical interfaces for initial use and to configure the external interfaces specified as the TOE's Operational Environment by [ST].

3.3.2.4 AGD_PRE.1 Evaluation Activity

The evaluator shall examine the preparative procedures to ensure they include instructions on how to manage the TSF as a product and as a component of the larger Operational Environment in a manner that allows to preserve integrity of the TSF. The intent of this requirement is to ensure there exists adequate preparative procedures (guidance in most cases) to put the TSF in a secure state (i.e., evaluated configuration). AGD_PRE.1 lists general requirements, the specific assurance activities implementing it are performed as part of FMT_SMF.1, FMT_MTD.1 and FMT_MOF.1 series of SFRs

The evaluators observed that the TOE's documentation includes guidance on configuring the TOE's interactions with its operational environment where needed. This includes setting up trusted channels to the TOE's Operational Environment even though the data carried over those channels is outside the scope of [NDcPP]. This ensures that the TOE can be deployed properly in its intended context while being configured in a secure manner as claimed by [ST].

3.3.2.5 AGD_PRE.1 Evaluation Activity

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

The evaluators reviewed [CCECG] and observed that the CLI uses SSH by default. However, [CCECG] also includes instructions for `enable common criteria` mode, which ensures that SSH is configured in the manner claimed by [ST]. It also includes instructions for additional configuration steps to ensure that the SSH configuration is consistent with [ST].

The `enable common criteria` command enables all settings required for Common Criteria compliance. This includes all cryptographic protocols and ciphersuites.

The evaluators reviewed [CCECG] and observed that section “Set Passwords” states that at installation, there are no default passwords for these accounts:

- The cliadmin account is the CLI administrative user account.
- The data admin password is used for encrypting credentials and backups.
- The uiadmin account is the administrative account for the graphical user interface, used when logging in to the client user interface.

3.4 Class ALC: Life-Cycle Support

3.4.1 ALC_CMC.1 Labeling of the TOE

When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

3.4.2 ALC_CMS.1 TOE CM Coverage

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

3.5 Class ATE: Tests

3.5.1 ATE_IND.1 Independent Testing – Conformance

The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

The evaluator shall consult Appendix B when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section B.4.3.1.

The evaluators developed a test plan ([Test]) to list all of the individual test evaluation activities for the TOE based on the claimed SFRs. The test plan lists, for each evaluation activity, the test results (including supporting evidence where necessary) and the testing verdict. In all cases, the tests were observed to be passing.

3.6 Class AVA: Vulnerability Assessment

3.6.1 AVA_VAN.1 Vulnerability Survey

While vulnerability analysis is inherently a subjective activity, a minimum level of analysis can be defined and some measure of objectivity and repeatability (or at least comparability) can be imposed on the vulnerability analysis process. In order to achieve such objectivity and

repeatability it is important that the evaluator shall follow a set of well-defined activities and documents their findings so others can follow their arguments and come to the same conclusions as the evaluator. While this does not guarantee that different evaluation facilities will identify exactly the same type of vulnerabilities or come to exactly the same conclusions, the approach defines the minimum level of analysis and the scope of that analysis and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities.

In order to meet these goals some refinement of the AVA_VAN.1 CEM work units is needed. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

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

3.6.1.1 AVA_VAN.1 Evaluation Activity (Documentation)

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

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

The developer shall provide documentation identifying the list of software and hardware components[7] that compose the TOE. Hardware components should identify compute-capable hardware components, at a minimum that must include the processor, and where applicable, discrete crypto ASICs, TPMs, etc. used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic implementations, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

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

a. documentation describing the allocation of requirements between distributed TOE components as in [NDcPP, 3.4]
b. a mapping of the auditable events recorded by each distributed TOE component as in [NDcPP, Table 2]
c. additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.4.1.2 and 3.5.1.2.

The TOE is not distributed.

3.6.1.2 AVA_VAN.1 Evaluation Activity

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

The evaluators conducted vulnerability research and penetration testing to determine the vulnerability of the TSF to attackers with Basic Attack Potential.

The evaluators conducted searches in public vulnerability repositories for the following Type 1 flaws based on the guidance specified in [ND-SD]:

- The list of software and hardware components that comprise the TOE
- The TOE name (including model information as appropriate).

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

- National Vulnerability Database (https://nvd.nist.gov/).
- Vulnerability Notes Database (http://www.kb.cert.org/vuls/)
- Tipping Point Zero Day Initiative (https://www.zerodayinitiative.com/advisories/published/)

The evaluation team performed a search of the following product specific databases:

- RedHat (https://access.redhat.com/security/)
- OpenSSL (https://openssl-library.org/news/vulnerabilities/)

These searches were last performed on 15 October 2025.

Specifically, the following search terms were used in these searches:

- The list of software and hardware components that comprise the TOE:
  - TOE Name: RedSeal Server v10.5
  - Processor:
    - Intel Xeon 5217
  - The processors consist of the following microarchitectures:
    - Cascade Lake
  - The underlying OS the TOE runs upon:
    - RHEL 9.6
  - Relevant Software included in the TOE that was provided to the evaluator, including but not limited to:
    - OpenSSL 3.2.2-6
    - Dell BSAFE SSL-J 7.3.1
    - Temurin JRE 17.0.15+6

Additionally, the evaluators performed fuzz testing of the TOE as specified in Section A.1.4 of [ND-SD]. The evaluators observed the TOE did not react adversely to the packets directed at the TOE or respond to the packets. This testing did not discover any vulnerabilities in the TOE.

The evaluation team determined that no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential. This information is documented in [VA].