www.GossamerSec.com

Assurance Activity Report for Ruckus SmartZone WLAN Controllers & Access Points with WIDS, R5.2.1.3

Version 0.4
09/18/23

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

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

Document: AAR-VID11382

Revision History

Revision	Date	Authors	Summary
Version 0.1	06/26/23	Cody Cummins	Initial draft
Version 0.2	08/04/23	Cody Cummins	ECR Comments Addressed; Equivalence and CAVP Certificates sections updated
Version 0.3	09/11/23	Cody Cummins	ECR Comments Addressed; FDP_IFC.1 activities added
Version 0.4	9/18/23	Kevin Cummins	ECR Comments Addressed

The TOE Evaluation was Sponsored by:

CommScope Technologies LLC
130 Holger Way
San Jose, CA 95134

Evaluation Personnel:

- Kevin Cummins
- Cody Cummins

Common Criteria Versions:

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

Common Evaluation Methodology Versions:

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

Table of Contents

1. Introduction

This document presents evaluations results of the Ruckus SmartZone WLAN Controllers & Access Points with WIDS, R5.2.1.3 NDcPP22e/WIDS10/WLANAS10 evaluation. This document contains a description of the assurance activities and associated results as performed by the evaluators.

1.1 Equivalence

This section explains why the test subset was adequate to address all product installations.

1.1.1 Evaluated Platform Equivalence

The Security Target identifies the models in section *1.4.1 TOE Architecture.* The TOE has the following Access Point TOE components: R650 (Including R650-WW), R750 (including T750SE/T750 Omni and T750-WW), and R850. The TOE also has the following Wireless Controllers: SmartZone 144 (SZ-144), SmartZone 300 (SZ-300), virtual SmartZone (vSZ-H hosted on a physical device), and virtual SmartZone – Data plane (vSZ-D hosted on a physical device).

Section *1.4.1.1 Physical Boundaries* of the ST includes tables with the following hardware information:

Controller	CPU
Smart Zone 144 (SZ 144)	Intel(R) Xeon(R) CPU D-2143IT (Skylake)
Smart Zone 300 (SZ 300)	Intel(R) Xeon(R) CPU E5-2695 v3 (Haswell)
Ruckus virtual SmartZone (vSZ) on VMware ESXi 7.0	Intel(R) Xeon(R) Silver 4309Y CPU (Ice Lake)
Ruckus virtual SmartZone – Data plane (vSZ-D) on VMware ESXi 7.0	Intel(R) Xeon(R) Silver 4309Y CPU (Ice Lake)

Controller CPU Identification

AP	CPU
R650 (Including R650-WW)	Qualcomm IPQ8071 (ARMv8)
R750 (including T750SE/T750 Omni and T750-WW)	Qualcomm IPQ8076 (ARMv8)
R850	Qualcomm IPQ8078(ARMv8)

AP CPU Identification

All physical controller models share the same processor assembly x86_64 and run the same 5.2.1.3 software. However, due to some hardware differences, there is a unique software image for the SZ144 and SZ300 series, and thus all relevant tests, according to Section 7 Requirement Allocation in the ST, were performed on both software images except where noted explicitly in the test writeups. The differencing characteristics affect only non-TSF relevant functionality (such as # of access points, WLAN and wireless clients, and amount of storage) and therefore support security equivalency of the models in terms of hardware. The physical appliances that were tested are the SZ144 and SZ300.

The Security Target also identifies a virtual model of the SZ controller. The virtual model is provided as a single image which can be run in two different mode's, High Scale (vSZ-H) and Essentials (vSZ-E) which support different numbers of APs and clients. As they are two modes of the same image, the code for the security functionality is identical between the two images. This vSZ-H is a replica of the physical SZ300 from a TSF relevant functionality standpoint while the vSZ-E is a replica of the SZ144. The only differences between the virtual and physical platforms are some extra code required for the virtualization process and varying physical hardware. The vSZ-H was tested according to Section 7 Requirement Allocation in the ST except where noted explicitly in the test writeups. The virtual platform also uses a vSZ-D which works the same way regardless of which virtual controller it is registered to. The purpose of the vSZ-D is to limit the strain on the virtual SZ controller by establishing the ipsec tunnel with access points for WLAN data. This tunnel uses the same Strongswan code that the SZ controller uses. The vSZ-D was tested according to Section 7 Requirement Allocation in the ST except where noted explicitly in the test writeups.

All access points included in the evaluation run the same software image. The only differencing characteristics affect only non-TSF relevant functionality (such as hardware differences that affect, speed range, and number of clients) and therefore support security equivalency of the models in terms of hardware. The APs that were tested in the eval were the R650, R750, and R850. The evaluator tested tested each AP against a different controller. So testing between the SZ144 and AP used the R750, testing between the SZ300 and AP used the R650, testing between the vSZ and AP used the R850.

Different models of the TOE leverage the same code for encrypted channels. This code is included as part of the different software images. Most notably, the TOE's IKE and IPsec implementation, operates equivalently across all models. All SZ controller models contain and use an identical IKE/IPsec/X.509 implementation for external communication with a syslog server. Furthermore, all models, including SZ controllers, vSZ-D and AP models use that same IKE/IPsec/X.509 implementation for internal communication between TOE models for the WLAN data tiunnel. The IPsec functionality was fully tested against the three SZ models tested in the evaluation, the SZ144, SZ300, and vSZ-H. To verify the equivalent IKE and IPsec implementation was working on the other model types (vSZ-D and AP) a successful IPSec connection for internal communication between two components was verified for all models used in testing.

In addition, the TOE's SSH client implementation, operates equivalently across the AP and vSZ-D models for the internal communication between TOE models for central management. The SSH client testing was fully tested against the R850 AP. To verify the equivalent SSH client implementation was working on the other model types (vSZ-D and AP) a successful SSH client connection for internal communication between two components was verified for all models used in testing.

Below is a table to state which devices each test was performed on. Note only the SFRs where testing was explicitly performed are included in this table. There are several tests that are met by references back to other tests that were not included in this table. That reference can be found in the test writeups. See text above for which AP was used with a given controller. The vSZ-D results are provided as part of the vSZ results.

Requirement	Devices Tested

WIDS10:FAU_ARP_EXT.1	All Controllers
NDcPP22e/WIDS10/WLANAS10:FAU_GEN.1	SZ144, vSZ, vSZ-D, AP
WIDS10:FAU_INV_EXT.1	All Controllers
WIDS10:FAU_INV_EXT.2	All Controllers
WIDS10:FAU_INV_EXT.3	All Controllers
WIDS10:FAU_RPT_EXT.1	All Controllers
WIDS10:FAU_SAA.1	All Controllers and APs
NDcPP22e:FAU_STG_EXT.1	All Controllers
WIDS10:FAU_WID_EXT.2	All Controllers and APs
NDcPP22e:FCO_CPC_EXT.1	All
WLANAS10:FCS_CKM.1/WPA	All Controllers and APs
WLANASEP10:FCS_CKM.2/PMK	All Controllers and APs
NDcPP22e:FCS_IPSEC_EXT.1	All Controllers
NDcPP22e:FCS_NTP_EXT.1	All Controllers
NDcPP22e:FCS_SSHC_EXT.1	R850 AP
NDcPP22e:FCS_SSHS_EXT.1	All Controllers
NDcPP22e:FCS_TLSC_EXT.1	All Controllers
NDcPP22e:FCS_TLSS_EXT.1	All Controllers
WLANAS10:FIA_8021X_EXT.1	All Controllers and APs
NDcPP22e:FIA_AFL.1	All Controllers
NDcPP22e:FIA_PMG_EXT.1	All Controllers
WLANAS10:FIA_PSK_EXT.1	All Controllers
WLANAS10:FIA_UAU.6	All Controllers
NDcPP22e:FIA_UIA_EXT.1	All Controllers

NDcPP22e:FIA_X509_EXT.1/Rev	All Controllers
NDcPP22e:FIA_X509_EXT.2	All Controllers
NDcPP22e:FIA_X509_EXT.3	All Controllers
WLANAS10:FMT_SMF.1 /AccessSystem	All Controllers
WIDS10:FMT_SMF.1	All Controllers
NDcPP22e:FMT_SMR.2	All Controllers
WLANAS10:FMT_SMR_EXT.1	All Controllers
WLANAS10:FPT_FLS.1	SZ144 and R650
NDcPP22e:FPT_ITT.1	All – Note that for the virtual platform the SSH is between the vSZ and AP and the IPsec is between the vSZ-D and AP
NDcPP22e:FPT_STM_EXT.1	All Controllers
NDcPP22e:FPT_TST_EXT.1	All
NDcPP22e:FPT_TUD_EXT.1	All
NDcPP22e:FTA_SSL.3	All Controllers
NDcPP22e:FTA_SSL.4	All Controllers
NDcPP22e:FTA_SSL_EXT.1	All Controllers
NDcPP22e:FTA_TAB.1	All Controllers
WLANASEP10:FTA_TSE.1	All Controllers and APs
NDcPP22e:FTP_ITC.1	All Controllers
WLANAS10:FTP_ITC.1	All Controllers and APs
NDcPP22e:FTP_TRP.1/Admin	All Controllers
NDcPP22e:FTP_TRP.1/Join	All

1.1.2 CAVP Certificates

The TOE has several cryptographic libraries in each distributed component. The Controllers have version 5.2.1.3 of Ruckus Smartzone SSL Crypto Library and version 5.2.1.3 of Ruckus Smartzone Kernel Crypto Library. The associated CAVP certificates are:

Ruckus Smartzone SSL Crypto Library version 5.2.1.3:

Requirements	Functions	SZ Cert	SZv Cert	SZvD Cert
	Cryptographic key generation			
FCS_CKM.1	RSA schemes using cryptographic key sizes of 2048-bits, 3072-bits	C2082	A2457	A2458
FCS_CKM.1	ECC schemes using 'NIST curves' P-256, P-384, P-521	C2082	A2457	A2458
FCS_CKM.1	FFC schemes using cryptographic key sizes of 2048-bits, 3072-bits	C2082	A2457	A2458
	Cryptographic key establishment/distribution			
FCS_CKM.2	RSA-based key establishment schemes	Vendor Affirmed	Vendor Affirmed	
FCS_CKM.2	Elliptic curve-based key establishment schemes	A2456	A2457	A2458
FCS_CKM.2	Finite field-based key establishment schemes	A2456	A2457	A2458
	Encryption/Decryption			
FCS_COP.1/ DataEncryption	AES CBC and GCM (128, 192 and 256 bits)	C2082	A2457	A2458
FCS_COP.1/ DataEncryption	AES CTR (128 and 256 bits)	C2082	A2457	A2458
	Cryptographic hashing			
FCS_COP.1/Hash	SHA-1/256/384/512 (digest sizes 160, 256, 384 and 512 bits)	C2082	A2457	A2458
	Keyed-hash message authentication			
FCS_COP.1/ KeyedHash	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (key and output MAC sizes 160, 256, 384 and 512 respectively)	C2082	A2457	A2458
	Cryptographic signature services			
FCS_COP.1/SigGen	RSA Digital Signature Algorithm (rDSA) (modulus 2048)	C2082	A2457	A2458
FCS_COP.1/SigGen	Elliptic Curve Digital Signature Algorithm (ECDSA) with an elliptical curve size of 256, 384, or 521 bits	C2082	A2457	A2458
	Random bit generation			
FCS_RBG_EXT.1	CTR_DRBG (AES) and Hash_DRBG with HW based noise sources (256 bits)	C2082	A2457	A2458

Ruckus Smartzone Kernel Crypto Library version 5.2.1.3

Requirements	Functions	SZ Cert	SZv Cert	SZvD Cert
	Encryption/Decryption			

Requirements	Functions	SZ Cert	SZv Cert	SZvD Cert
FCS_COP.1/ DataEncryption	AES CBC (128 and 256 bits)	C2077	A3731	A3732
	Cryptographic hashing			
FCS_COP.1/Hash	SHA-1/256/384/512 (digest sizes 160, 256, 384 and 512 bits)	C2077	A3731	A3732
	Keyed-hash message authentication			
FCS_COP.1/ KeyedHash	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (key and output MAC sizes 160, 256, 384 and 512 respectively)	C2077	A3731	A3732

The APs have three cryptographic libraries – Ruckus Access Point SSL Crypto Library (version 5.2.1.3), Ruckus Access Point Kernel Crypto Library (version 5.2.1.3), and Ruckus Access point Radio Crypto Library (version 1.0). The associated CAVP certificates are:

Ruckus Access point Radio Crypto Library (version 1.0):

Requirements	Functions	AP Cert
	Encryption/Decryption	
FCS_COP.1/DataEncryption [WLAN]	AES CCMP (128 and 256 bits)	5312

Ruckus Access Point SSL Crypto Library (version 5.2.1.3):

Requirements	Functions	AP Cert
	Cryptographic key generation	
FCS_CKM.1	RSA schemes using cryptographic key sizes of 2048-bits, 3072-bits	C2093
FCS_CKM.1	ECC schemes using 'NIST curves' P-256, P-384, P-521	C2093
	Cryptographic key establishment/distribution	
FCS_CKM.2	RSA-based key establishment schemes	Vendor Affirmed
FCS_CKM.2	Elliptic curve-based key establishment schemes	C2093
	Cryptographic key distribution	
FCS_CKM.2/GTK	AES Key Wrap in EAPOL-Key frame decryption only (as TOE acts as Authenticator only, and not as a supplicant) (128 bits)	C2093
	Encryption/Decryption	
FCS_COP.1/ DataEncryption	AES CBC and GCM (128 and 256 bits)	C2093
FCS_COP.1/ DataEncryption	AES CTR (128 and 256 bits)	C2093
	Cryptographic signature services	

Requirements	Functions	AP Cert
FCS_COP.1/SigGen	RSA Digital Signature Algorithm (rDSA) (modulus 2048)	C2093
FCS_COP.1/SigGen	Elliptic Curve Digital Signature Algorithm (ECDSA) with an elliptical curve size of 256, 384, or 521 bits	C2093
	Cryptographic hashing	
FCS_COP.1/Hash	SHA-1/256/384/512 (digest sizes 160, 256, 384 and 512 bits)	C2093
	Keyed-hash message authentication	
FCS_COP.1/ KeyedHash	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (key and output MAC sizes 160, 256, 384 and 512 respectively)	C2093
	Random bit generation	
FCS_RBG_EXT.1	CTR_DRBG (AES) and Hash_DRBG with HW based noise sources (256 bits)	C2093

Ruckus Access Point Kernel Crypto Library (version 5.2.1.3):

Requirements	Functions	AP Cert
	Cryptographic key generation	
FCS_CKM.1(2) [WLAN]	WPA2 128-bit cryptographic key derivation	C2092 See WFA certificates below
	Encryption/Decryption	
FCS_COP.1/ DataEncryption	AES CBC, GCM (128 and 256 bits)	C2092
	Cryptographic hashing	
FCS_COP.1/Hash	SHA-1/256/384/512 (digest sizes 160, 256, 384 and 512 bits)	C2092
	Keyed-hash message authentication	
FCS_COP.1/ KeyedHash	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512 (key and output MAC sizes 160, 256, 384 and 512 respectively)	C2092

The TOE has successfully completed certification (including WPA2 Enterprise) and received Wi-Fi CERTIFIED Interoperability Certificates from the Wi-Fi Alliance. The Wi-Fi Alliance maintains a website providing further information about the testing program: http://www.wi-fi.org/certification.

Device Name	Wi-Fi Alliance Certificate Numbers
R650 (Including R650-WW)	WFA100880
R750 (Including T750SE/T750 Omni and T750-WW)	WFA99977
R850	WFA101708

1.2 References

The following evidence was used to complete the Assurance Activities:

- Ruckus SmartZone WLAN Controllers & Access Points with WIDS, R5.2.1.3 Security Target, Version 0.6, 09/18/2023 **(ST)**
- RUCKUS FIPS and Common Criteria Configuration Guide for SmartZone and APs, 5.2.1.3, Part Number: 800-72735-001 Rev D, October 2023 **(Admin Guide)**

2. Protection Profile SFR Assurance Activities

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

2.1 Security audit (FAU)

2.1.1 Security Alarms (WIDS10:FAU_ARP.1)

2.1.1.1 WIDS10:FAU_ARP.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS describes where to find the WIDS alerts on the Administrator console/interface.

Section 6.1 of **ST** states the TOE displays WIDS alerts to the administrator on the Controller and includes the identity of APs and EUDs involved, signal strength, accurate event timestamp, description of alert and severity level. A table of all the APs and clients detected by the monitoring APs by can be viewed under Report > Rogue Devices. This table also specifies if the rogue devices hit a configured WIDS event. The formal audits for the detection of these WIDS alerts can be viewed under Access Points > Monitoring APs > specific monitoring AP > Events.

Component Guidance Assurance Activities: The evaluator shall use the operational guidance for instructions on where the alerts generated are displayed within the WIDS interface. If 'capture raw frame traffic that triggers the violation is selected', the evaluator shall use the operational guidance to configure the traffic capture capabilities.

Section “Reports” in **Admin Guide** describes where the Rogue APs and Clients can be viewed in a table which includes the column Classification Policy defining any WIDS event matches. Section “Audit/Event Alert” describes how to view the corresponding audit events when a WIDS event is detected. Capture raw frame traffic is not selected.

Component Testing Assurance Activities: Test 1: The evaluator shall perform a series of events or generate traffic that would successfully trigger an alert for each of the rules defined in FAU_SAA.1. The evaluator should verify and record whether the TOE generated the alert for each rule, and provided sufficient details. The evaluator should also record the events or traffic that was generated as each alert was attempted to be triggered and record the details provided by the TOE in the alert.

Test 2: [conditional] If capturing of raw frames was selected, verify that the packet capture was triggered and stored as appropriate.

Test 1: The evaluator performed testing of WIDS events for FAU_SAA.1. The evaluator verified the TOE generated the alert for each rule, when traffic matched the event.

Test 2 This test is not applicable as the capturing of raw frames was not selected.

2.1.2 Security Alarm Filtering (WIDS10:FAU_ARP_EXT.1)

2.1.2.1 WIDS10:FAU_ARP_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS describes the ability of the TOE to filter WIDS/WIPS alerts.

Section 6.1 of **ST** states the administrator is able to configure the WIDS alerts from the Controller and can filter the events collected by classification. The administrator can select WIDS alerts based on the following:

- Accumulation or combination of rules known to indicate a potential security violation (signature-based attacks),
- Detection of non-allowlisted AP,
- Detection of non-allowlisted EUD,
- Detection of authorized EUD establishing peer-to-peer connection with any other EUD,
- Detection of EUD bridging two network interfaces,
- Detection of unauthorized point-to-point wireless bridges by allowlisted APs,
- Alert generated by violation of user defined signature,
- Detection of ICS connection,
- Detection of traffic with excessive transmit power level,
- Detection of MAC spoofing,
- Detection of unauthorized AP broadcasting authorized SSIDs,
- Detection of authorized AP broadcasting an unauthorized SSID,
- Detection of allowlisted EUD connected to unauthorized SSID,
- Detection of NULL SSID associations,
- Detection of active probing,
- Detection of packet flooding/DoS/DDoS,
- Detection of RF-based denial of service,
- Detection of deauthentication flooding,
- Detection of disassociation flooding,
- Detection of request-to-send/clear-to-send abuse,
- Detection of unauthorized authentication scheme use,
- Detection of unauthorized encryption scheme use,

- Detection of unencrypted traffic,
- Detection of allowlisted EUD or AP that is using weak/outdated WLAN protocols and protocol implementations,
- Detection of extremely high numbers of client devices using a particular allowlisted AP,
- Detection of a high number of failed attempts to join the WLAN in a short period of time,
- Detection of the use of active WLAN scanners (e.g. wardriving tools) to generate WLAN traffic, such as Probes, Auths, and Assoc frames,
- Detection of the physical location of an identified WLAN threat by using triangulation,
- Detection of an SSID using weak/unsupported/disallowed encryption options,
- Detection of AP SSID larger than 32 bytes

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance includes instructions on enabling and disabling alerts.

Section "Classifying a Rogue Policy" in **AGD** describes how to define classification rules for WIDS events. Section "Different Rule types and Classification" describes all the preset rules for WIDS events, while "Signature Based Detection Rule" describes how an administrator can create a custom rule.

Component Testing Assurance Activities: Test 1:

Step 1: The evaluator shall use the operational guidance to enable/disable detection of available detection capabilities through the WIDS administrator interface. The evaluator shall then generate traffic that would successfully trigger the alert. The evaluator should verify that the TOE generated the alert.

Step 2: The evaluator shall disable the alert. The evaluator shall then generate events as in previous test that should successfully trigger the alert. The evaluator shall verify that the TOE did not generate an alert.

Test 1:

Step 1: The evaluator enabled detection of a WIDS event. The evaluator saw when generating traffic to match the WIDS event, an alert was triggered by the TOE.

Step 2: The evaluator next disabled the alert. The evaluator saw when generating traffic to match the WIDS event, an alert was no longer triggered by the TOE.

2.1.3 Audit Data Generation (NDcPP22e/WLANAS10:FAU_GEN.1)

2.1.3.1 NDcPP22e:FAU_GEN.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.3.2 NDcPP22e/WLANAS10:FAU_GEN.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.1 in the **ST** states the TOE generates audit records for start-up and shutdown of the TOE, all administrator actions, and for all the events identified in Table 4 and Table 5 Auditable Events. Audit records include date and time of the event, type of event, user identity that caused the event to be generated, the outcome of the event, as well as the additional content listed in column 3 of **Table 4** and **Table 5**. Any event that needs to be audited due to a user action is recorded with the identity of the user along with timestamps.

Section 6.1 in the **ST** also states that for cryptographic keys, the act of importing and deleting a key is audited with the key ID and the associated administrator account that performed the action is recorded.

Section 7 in the **ST** provides a mapping of the distributed TOE components to the SFRs and the required audit events. The evaluator confirmed that all components generating audit information for a particular SFR also contribute to that SFR and that the table accounts for all TOE auditable events.

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

The evaluator shall also make a determination of the administrative actions related to TSF data related to configuration changes. The evaluator shall examine the guidance documentation and make a determination of which administrative commands, including subcommands, scripts, and configuration files, are related to the

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

The "Audit Records" section of the **Admin Guide** provides a list of all auditable events required by FAU_GEN.1. From a review of the **ST**, the **Admin Guide** and through testing, the evaluator determined that the list of auditable events includes all administrative actions related to TSF data configuration changes. These administrative actions are consistent with the security requirements implemented in the TOE and were found to have appropriate management capabilities identified in the guidance documentation.

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

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

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

The evaluator created a list of the required audit events and collected these audit events when running the other security functional tests described by the protection profiles. The evaluator then recorded these audit events for in the proprietary Detailed Test Report (DTR). The security management events are handled in a similar manner. The evaluator collected the required audits for all TOE components, via syslog, during the course of testing. When the administrator was required to set a value for testing, the audit record associated with the administrator action was collected and recorded in the DTR. The evaluator ensured the collected audits matched the mapping of relevant components for each SFR as demonstrated in Section 7 of the ST.

2.1.4 Audit Data Generation (WIDS) (WIDS10:FAU_GEN.1/WIDS)

2.1.4.1 WIDS10:FAU_GEN.1.1/WIDS

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.4.2 WIDS10:FAU_GEN.1.2/WIDS

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator shall test the TOE's ability to correctly generate audit records by having the TOE generate audit records in accordance with the evaluation activities associated with the functional requirements in this PP-Module. When verifying the test results, the evaluator shall ensure the audit records generated during testing match the format specified in the administrative guide, and that the fields in each audit record have the proper entries.

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

The evaluator created a list of the required audit events and collected these audit events when running the other security functional tests described by the protection profiles. The evaluator then recorded these audit events for in the proprietary Detailed Test Report (DTR). The evaluator collected the required audits for all TOE components, via syslog, during the course of testing. The evaluator ensured the collected audits matched the mapping of relevant components for each SFR as demonstrated in Section 7 of the **ST**. The evaluator verified that each collected audit matched the expected format demonstrated in the **Admin Guide**.

2.1.5 Audit Data Generation (WLANAS10:FAU_GEN.1/WLAN)

2.1.5.1 WLANAS10:FAU_GEN.1.1/WLAN

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will test the TOE's ability to correctly generate audit records by having the TOE generate audit records in accordance with the evaluation activities associated with the functional requirements in this PPModule. When verifying the test results, the evaluator will ensure the audit records generated during testing match the format specified in the administrative guide and that the fields in each audit record have the proper entries.

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

The evaluator created a list of the required audit events and collected these audit events when running the other security functional tests described by the protection profiles. The evaluator then recorded these audit events for in the proprietary Detailed Test Report (DTR). The evaluator collected the required audits for all TOE components, via syslog, during the course of testing. The evaluator ensured the collected audits matched the mapping of relevant components for each SFR as demonstrated in Section 7 of the **ST**. The evaluator verified that each collected audit matched the expected format demonstrated in the **Admin Guide**.

2.1.6 User identity association (NDcPP22e:FAU_GEN.2)

2.1.6.1 NDcPP22e:FAU_GEN.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

The TSS requirements for FAU_GEN.2 are covered by the TSS requirements for FAU_GEN.1.

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

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

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

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

This test was accomplished as part of NDcPP21:FAU_GEN.1.

2.1.7 Security Audit Generation (NDcPP22e:FAU_GEN_EXT.1)

2.1.7.1 NDcPP22e:FAU_GEN_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

Component Guidance Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

Component Testing Assurance Activities: For distributed TOEs, the requirements on TSS, Guidance Documentation and Tests regarding FAU_GEN_EXT.1 are already covered by the corresponding requirements for FAU_GEN.1.

The above assurance activities for TSS, Guidance and Testing are already covered by the corresponding assurance activities for NDcPP22e/WLANAS10:FAU_GEN.1.

2.1.8 Security Audit Generation - per TD0651 (WLANAS10:FAU_GEN_EXT.1)

2.1.8.1 WLANAS10:FAU_GEN_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.1.9 Intrusion Detection System - Intrusion Detection Methods (WIDS10:FAU_IDS_EXT.1)

2.1.9.1 WIDS10:FAU_IDS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS includes which intrusion detection method(s) the TOE utilizes. If multiple methods are selected, the evaluator shall confirm that the TSS describes how the different methods are incorporated.

Section 6.1 of **ST** states the TOE performs signature-based intrusion detection.

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions on how to configure the TOE in order for it to detect such intrusions.

Section “Classifying a Rogue Policy” in **AGD** describes how to define classification rules for WIDS events. Section “Different Rule types and Classification” describes all the preset rules for WIDS events, while “Signature Based Detection Rule” describes how an administrator can create a custom rule.

Component Testing Assurance Activities: Depending on the detection technique used by the TOE, the evaluator shall confirm and note the existence of the capability and test for the appropriate selection-based requirements.

The TOE utilized signature-based detection which can be seen in WIDS10:FAU_SAA.1_t6 where the evaluator tested the detection of a user defined signature.

2.1.10 Environmental Inventory (WIDS10:FAU_INV_EXT.1)

2.1.10.1 WIDS10:FAU_INV_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.10.2 WIDS10:FAU_INV_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.10.3 WIDS10:FAU_INV_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS describes how the presence of authorized EUDs and APs is presented by the TOE. The evaluator shall verify that the TSS includes where in the WIDS interface the list of detected APs and EUDs is displayed.

Section 6.1 of **ST** states the TOE determines if a given AP or EUD is allowed based on its MAC address. It can detect allowed and non-allowed APs and EUDs. A table of all the APs and clients detected by the monitoring APs by can be viewed under Report > Rogue Devices which includes whether the device is allowlisted or not.

Component Guidance Assurance Activities: The evaluator shall verify that the operational guidance provides instructions on how to view authorized and unathorized APs and EUDs that are within range of the TOE sensors.

Section “Reports” in **AGD** describes where the Rogue APs and Clients can be viewed in a table which includes whether the device is allowlisted or not.

Component Testing Assurance Activities: Test 1:

Step 1: Per guidance in FMT_SMF.1/WIDS, add MAC Addresses or other unique device identifier for an AP and EUD to the allowlist.

Step 2: Deploy the AP and EUD that were added to allowlist within the range of the TOE's sensors.

Step 3: Verify that the devices are classified as authorized.

Step 4: Remove the EUD from the allowlist.

Step 5: Verify that the EUD is classified as unauthorized.

Step 6: Remove the AP from the allowlist.

Step 7: Verify that the AP is classified as unauthorized.

Test 2:

Step 1: Deploy an allowlisted AP and EUD, and connect the EUD to the AP.

Step 2: Verify that the list of detected APs and EUDs contains the allowlisted AP and EUD that were just deployed.

Step 3: If the AP and EUD are detected verify that they are classified as allowlisted devices.

Test 3:

Step 1: Deploy a non-allowlisted AP and EUD and connect the EUD to the AP.

Step 2: Verify that the list of detected APs and EUDs contains the non-allowlisted AP and EUD that were just deployed.

Step 3: If the AP and EUD are detected verify that they are not classified as allowlisted devices.

Test 1:The evaluator added an AP and an EUD to the TOE’s allowlist using their MAC addresses. The evaluator deployed the AP and EUD within the range of the TOE's sensors and viewed that they were detected as allowlisted.

The evaluator removed the AP and EUD form the allowlist and viewed that they were no longer detected as allowlisted.

Test 2: The evaluator added an AP and an EUD to the TOE’s allowlist using their MAC addresses. The evaluator deployed the AP and EUD within the range of the TOE's sensors, with the EUD connected to the AP, and viewed that they were detected as allowlisted.

Test 3: The evaluator removed the AP and an EUD from the TOE’s allowlist. The evaluator deployed the AP and EUD within the range of the TOE's sensors, with the EUD connected to the AP, and viewed that they were not detected as allowlisted.

2.1.11 Characteristics of Environmental Objects (WIDS10:FAU_INV_EXT.2)

2.1.11.1 WIDS10:FAU_INV_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.11.2 WIDS10:FAU_INV_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.11.3 WIDS10:FAU_INV_EXT.2.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS explains the capability of detecting the information specified in the requirements for all APs and EUDs within the TOE's wireless range.

Section 6.1 of the **ST** state the TOE can detect all of the required characteristics of the APs and EUDs in its operating environment. This includes the current RF band, current channel, MAC Address, received signal strength, device detection timestamps, classification of APs and EUDs for all APs and EUDs within range of the TOE's wireless sensors. In addition, for APs the details of encryption, number of connected EUDs, received frames/packets, beacon rate, SSID of AP (if not hidden) are detected. In addition, for EUDs the details of SSID and BSSID of AP it is connected to and DHCP configuration are detected.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance in order to verify that there are instructions that show how to locate the device inventory mentioned above.

Section “Reports” in **Admin Guide** provides the details on how to view the detected EUDs/APs in range and the details of the devices that are provided.

Component Testing Assurance Activities: Test 1:

Step 1: Deploy an allowlisted AP, non-allowlisted AP and two allowlisted EUDs.

Step 2: Connect one allowlisted EUD to the allowlisted AP and one to the non-allowlisted AP.

Step 3: Check the WIDS user interface for a list of detected APs and EUDs.

Step 4: Verify that current RF band, current channel, MAC Address, received signal strength, device detection timestamps, classification of device, are part of the information presented on the WIDS user interface for all the APs and EUDs detected. For APs verify that encryption, number of connected EUDs, SSID (if not hidden), received frames/packets and beacon rate are presented. For EUDs verify that the SSID and BSSID of AP it is connected and DHCP configuration is presented.

Test 1: The evaluator added an AP and two EUD to the TOE's allowlist using their MAC addresses. The evaluator deployed the 2 EUDs, with one connected to the allowlisted AP and one connected to a non-allowlisted AP. The evaluator viewed the list of detected devices and verified all the required info was provided. This included current RF band, current channel, MAC Address, received signal strength, device detection timestamps, and classification of device for all devices. Encryption, number of connected EUDs, SSID (if not hidden), received frames/packets and beacon rate for APs and for EUDs, the SSID and BSSID of AP it is connected and DHCP configuration.

2.1.12 Location of Environmental Objects (WIDS10:FAU_INV_EXT.3)

2.1.12.1 WIDS10:FAU_INV_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.12.2 WIDS10:FAU_INV_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS includes information on location tracking, optimal number of sensors and sensor placement to meet the required level of accuracy.

The evaluator shall verify that the TSS contains information regarding the TSF's ability to record signal strength of hardware operating within range of its sensors.

Section 6.1 of the **ST** states the TOE can detect the physical location of APs and EUDs within 15 feet of their actual location as well as the signal strength of hardware operating within range of the TOE's wireless sensors. To ensure the physical location of APs and EUDs can be detected, the controller requires the use of 2 managed sensor in order to perform triangulation.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance for instructions on how to configure location tracking, how to load a location map (if applicable), and where in the TSF administrator interface the location of APs and EUDs can be viewed.

If the option for detection of RF power levels above a predetermined threshold is selected, the evaluator shall use the operational guidance to set or check what the threshold is in a given test. The evaluator should also verify that the operational guidance provides instruction on how to configure the TOE to generate an alert when the threshold is exceeded.

Section “Locating a Rogue Device” of the **Admin Guide** describes how to identify the estimated location of an AP/EUD using an admin defined location map.

Section “Creating a Monitoring Access Point (AP)” describes how to configure the RSSI Threshold.

Component Testing Assurance Activities: Test 1:

Step 1: Deploy an AP within 25 feet of the sensors.

Step 2: Verify the TSF provides location tracking information about the AP.

Step 3: Verify the AP location presented is within 25 feet of the actual location.

Test 2:

Step 1: Deploy an AP within range of the sensors.

Step 2: Check the WIDS user interface for a list of detected APs and EUDs.

Step 3: Verify that the current received signal strength is part of the information presented on the WIDS user interface about the APs and EUDs.

Test 1: The evaluator deployed an AP within range of the TOE’s sensors. Two sensors were required for triangulation. The evaluator verified the TOE provided the location of the AP when requested and was accurate within 25 feet of the actual location.

Test 2: The evaluator deployed an AP within range of the TOE’s sensors. The evaluator verified the TOE detected the AP and presented the received signal strength from the AP.

2.1.13 Intrusion Detection System - Reporting Methods (WIDS10:FAU_RPT_EXT.1)

2.1.13.1 WIDS10:FAU_RPT_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.13.2 WIDS10:FAU_RPT_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS includes which method the TOE utilizes.

Section 6.1 of **ST** states The TOE uses syslog to report its WIDS alerts.

Component Guidance Assurance Activities: There are no operational guidance activities for this SFR.

There are no operational guidance activities for this SFR.

Component Testing Assurance Activities: Depending on the detection technique used by the TOE, the evaluator shall confirm and note the existence of the capability.

Test 1:

Step 1: Deploy an allowlisted AP and connect it to the protected wired infrastructure via wire.

Step 2: Confirm that the TSF can observe and capture traffic and events generated by the AP.

Step 3: Confirm that the TSF can use the reporting mechanisms specified in the TSS.

Step 4: Verify that the TSF can import and export observable event data in each of the formats specified in the TSS.

Test 1: The evaluator performed testing of WIDS events for FAU_SAA.1. The evaluator verified the TOE generated the alert for each rule, when traffic matched the event. In each case the validator further verified that the WIDS alert was reported as an audit to a remote syslog server as expected.

The evaluator also verified that WIDS configuration data could be imported and exported to and from the TOE.

2.1.14 Potential Violation Analysis (WIDS10:FAU_SAA.1)

2.1.14.1 WIDS10:FAU_SAA.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.14.2 WIDS10:FAU_SAA.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS describes the ability of the TOE to detect the network behavior described by the SFR. The evaluator shall verify that the TSS describes the methods that the TOE uses to detect the presence of unauthorized connections and unauthorized network traffic. The evaluator shall examine the TSS to verify that it describes the denial of service attacks that can be detected by the TOE. The evaluator shall verify that the TSS describes the ability of the TOE to detect when unauthorized WLAN authentication schemes and encryption schemes are used. The evaluator shall verify that the TSS describes the ability of the TOE to detect when unauthorized APs and EUDs send or receive unencrypted data.

Section 6.1 of **ST** states the TOE collects wireless intrusion detection system (WIDS) alerts. Each wireless sensor (AP) monitors wireless traffic, and forwards all data including alerts to the Controller in real time. The administrator is able to configure/enable these signature based WIDS alerts from the Controller and can filter the events collected by classification. The administrator can select WIDS alerts based on the following:

- Accumulation or combination of rules known to indicate a potential security violation (signature-based attacks),
- Detection of non-allowlisted AP,
- Detection of non-allowlisted EUD,
- Detection of authorized EUD establishing peer-to-peer connection with any other EUD,
- Detection of EUD bridging two network interfaces,
- Detection of unauthorized point-to-point wireless bridges by allowlisted APs,
- Alert generated by violation of user defined signature,
- Detection of ICS connection,
- Detection of traffic with excessive transmit power level,
- Detection of MAC spoofing,
- Detection of unauthorized AP broadcasting authorized SSIDs,
- Detection of authorized AP broadcasting an unauthorized SSID,
- Detection of allowlisted EUD connected to unauthorized SSID,
- Detection of NULL SSID associations,

- Detection of active probing,
- Detection of packet flooding/DoS/DDoS,
- Detection of RF-based denial of service,
- Detection of deauthentication flooding,
- Detection of disassociation flooding,
- Detection of request-to-send/clear-to-send abuse,
- Detection of unauthorized authentication scheme use,
- Detection of unauthorized encryption scheme use,
- Detection of unencrypted traffic,
- Detection of allowlisted EUD or AP that is using weak/outdated WLAN protocols and protocol implementations,
- Detection of extremely high numbers of client devices using a particular allowlisted AP,
- Detection of a high number of failed attempts to join the WLAN in a short period of time,
- Detection of the use of active WLAN scanners (e.g. wardriving tools) to generate WLAN traffic, such as Probes, Auths, and Assoc frames,
- Detection of the physical location of an identified WLAN threat by using triangulation,
- Detection of an SSID using weak/unsupported/disallowed encryption options,
- Detection of AP SSID larger than 32 bytes

Component Guidance Assurance Activities: If the ability of the TSF to detect the different potential security violations is configurable, the evaluator shall verify that the operational guidance provides instructions on how to configure the TOE.

Section “Classifying a Rogue Policy” in **Admin Guide** describes how to define classification rules for WIDS events. Section “Different Rule types and Classification” describes all the preset rules for WIDS events, while “Signature Based Detection Rule” describes how an administrator can create a custom rule.

Component Testing Assurance Activities: Test 1: Detection of non-allowlisted AP:

Step 1: Deploy a non-allowlisted AP.

Step 2: Verify that the AP is detected as a non-allowlisted AP.

Test 2: Detection of non-allowlisted EUD:

Step 1: Deploy a non-allowlisted EUD.

Step 2: Verify that the EUD is detected as an non-allowlisted EUD.

Test 3: Detection of authorized EUD establishing peer-to-peer connection with any other EUD:

Test 3.1: Create the following connections between two allowlisted EUDs.

Windows Ad Hoc Connection

Mac OS Ad Hoc

Linux Ad Hoc

Wi-Fi Direct

Test 3.2: Create the following connections between one allowlisted EUD and a non-allowlisted EUD

Windows Ad Hoc Connection

Mac OS Ad Hoc

Linux Ad Hoc

Wi-Fi Direct

Verify that alerts were generated by each of the connections in each test.

Test 4: Detection of EUD bridging two network interfaces:

Bridge two network interfaces on an allowlisted EUD (one must be the wireless card listed as allowlisted).

Step 1: Create a Windows Hosted Network with an allowlisted EUD.

Step 2: Connect a different allowlisted EUD to the network.

Verify that alerts were generated by each of the connections in each test.

Test 5: Detection of unauthorized point to point wireless bridges by allowlisted APs:

Step 1: Setup a point-to-point wireless bridge using allowlisted APs in the range of the wireless sensors.

Step 2: Verify that the TSF detects the bridge.

Test 6: Alert generated by violation of user defined signature:

Step 1: Setup a user defined detection signature.

Step 2: Verify that the TSF generates an alert once the rules of signature have been violated.

Test 7: Detection of ICS connection:

Step 1: Setup an Internet Connection Sharing (ICS) connection.

Step 2: Verify that the TSF detects the establishment of the ICS connection.

Test 8: Detection of traffic with excessive transmit power level:

Step 1: Configure a source of network traffic that can exceed the maximum transmit power levels of 100mW on 2.4GHz and 200mW on 5GHz.

Step 2: Configure a user defined signature to detects traffic with transmit power levels that exceed the maximum.

Step 3: Commence with the transmission of network traffic at excessive power levels.

Step 4: Collect wireless traffic with range of the TSF.

Step 5: Verify that the TSF detects wireless traffic that exceeds 100mW on 2.4GHz and 200mW on 5GHz.

Test 9: Detection of MAC spoofing:

Test 9.1:

Step 1: Spoof mac address of allowliste EUD connected to an allowlisted AP on a second EUD.

Step 2: Connect EUD with spoofed MAC address to another allowlisted AP while the valid EUD it is spoofing is connected to the first AP.

Step 3: Verify that the TSF detected the MAC spoofing.

Test 9.2:

Step 1: Spoof mac address of allowliste AP on a second AP.

Step 2: Verify that the TSF detected the MAC spoofing.

Test 10: Detection of unauthorized AP broadcasting authorized SSIDs:

Step 1: Configure a non-allowlisted AP to operate on a set channel on the 2.4 GHz band broadcasting an authorized SSID.

Step 2: Verify that the TSF detects the non-allowlisted AP broadcasting an authorized SSID.

Step 3: Repeat the test utilizing the 5 GHz band.

Test 11: Detection of authorized AP broadcasating an unathorized SSID:

Step 1: Configure an allowlisted AP to operate on a set channel on the 2.4 GHz band broadcasting an unauthorized SSID.

Step 2: Verify that the TSF detects the non-allowlisted AP broadcasting an authorized SSID.

Step 3: Repeat the test utilizing the 5 GHz band.

Test 12: Detection of allowlisted EUD connected to unauthorized SSID:

Step 1: Configure an allowlisted AP to operate on a set channel on the 2.4 GHz band with an unauthorized SSID.

Step 2: Connect an allowlisted EUD to the AP.

Step 3: Verify that the TSF detects the allowlisted EUD associated to the allowlisted AP broadcasting an unauthorized SSID.

Step 4: Repeat the test utilizing the 5 GHz band.

Test 13: Detection of NULL SSID associations:

Step 1: Deploy allowlisted AP.

Step 2: Configure the AP to have null SSID.

Step 3: Attempt to connect an allowlisted EUD to the AP without supplying the correct AP SSID.

Step 4: Verify that the AP does not permit the EUD to complete an association by returning a Probe Request.

Step 5: If an association does occur, confirm that an alert is triggered due to a violation of policy.

Test 14: Detection of active probing: ?Step 1: Perform an active scan on the subnet of the WLAN.

Step 2: Record tools used and type of scan performed.

Step 3: Verify that the TSF detects the active probing.

Test 15: Detection of packet flooding/DoS/DDoS: ?Step 1: Generate a large amount of TCP and UDP traffic from a single EUD.

Step 2: Verify that the TSF detects the network-based DoS.

Step 3: Generate a large amount of TCP and UDP traffic from multiple EUDs.

Step 4: Verify that the TSF detects the network-based DDoS.

Test 16: Detection of RF-based denial of service:

Step 1: Deploy an allowlisted AP and configure to stay in a particular channel.

Step 2: Connect an allowlisted EUD to the AP.

Step 3: Use an RF Jammer or signal generator on the same frequency as the AP and EUD to create a RF-based DoS.

Step 4: Verify that the TOE detects the RF-based DoS.

Test 17: Detection of deauthentication flooding:

Test 17.1:

Step 1: Deploy allowlisted AP and configure to a set channel.

Step 2: Connect an allowlisted EUD to the AP.

Step 3: Send an flood of deauthentication frames to the EUD using the MAC address of allowlisted AP it is connected to.

Step 4: Verify that the TSF detects the deauthentication flood.

Test 17.2:

Step 1: Deploy allowlisted AP and configure to a set channel.

Step 2: Connect an allowlisted EUD to the AP.

Step 3: Send an flood of deauthentication frames with the MAC address of allowlisted AP as the source and destination as a broadcast.

Step 4: Verify that the TSF detects the deauthentication flood.

Test 18: Detection of disassociation flooding:

Step 1: Deploy an allowlisted AP and connect authorized EUDs.

Step 2: Generate disassociation frames from an unauthorized EUD.

Step 3: Verify that the TSF detected the disassociation flooding.

Test 19: Detection of request-to-send/clear-to-send abuse:

Step 1: Deploy allowlisted AP and configure to a set channel.

Step 2: Connect two allowlisted EUDs to the AP.

Step 3: Send an flood of CTS frames to reserve RF medium.

Step 4: Verify that the TSF detects the CTS abuse.

Test 20: Detection of unauthorized authentication scheme use:

The evaluator shall configure the TOE, per FMT_SMF.1/WIDS, with 802.1x authentication as the only mode of authorized WLAN authentication scheme.

Test 20.1: (TD0613 applied)

Step 1: Deploy an allowlisted AP with open authentication.

Step 2: Verify that the TSF detects the AP broadcasting an unauthorized authentication schemes. If detected the test is satisfied. If not detected perform steps 3 and 4.

Step 3: Connect an allowlisted EUD to AP.

Step 4: Verify that the TSF detects the AP and the EUD using unauthorized authentication schemes.

Test 20.2:

Step 1: Deploy an allowlisted AP that uses pre-shared key authentication.

Step 2: Verify that the TSF detects the AP broadcasting an unauthorized authentication schemes. If detected the test is satisfied. If not detected perform steps 3 and 4.

Step 3: Connect an allowlisted EUD to AP.

Step 4: Verify that the TSF detects the AP and the EUD using unauthorized authentication schemes.

Test 21: Detection of unauthorized encryption scheme use:

Test 21.1:

Step 1: Configure the TOE with 128 bit AES encryption type as the only allowed encryption scheme.

Step 2: Deploy an allowlisted AP with no encryption.

Step 3: Connect an allowlisted EUD to AP.

Step 4: Verify that the TOE detects the AP and the EUD using unauthorized encryption schemes.

Test 21.2:

Step 1: Configure the TOE with 128 bit AES encryption type as the only allowed encryption scheme.

Step 2: Deploy an allowlisted AP that uses TKIP encryption only.

Step 3: Connect an allowlisted EUD to AP.

Step 4: Verify that the TSF detects the AP and the EUD using unauthorized encryption schemes.

Test 22: Detection of unencrypted traffic:

Test 22.1: (TD0610 applied)

Step 1: Deploy an allowlisted AP with no encryption.

Step 2: Connect an allowlisted EUD to AP and generate traffic.

Step 3: Verify that the TOE detects unencrypted data frames being sent between the allowlisted AP and EUD.

Step 4: Attempt to connect a non-allowlisted EUD to AP.

Step 5: If the connection attempt succeeds, generate traffic from the non-allowlisted EUD and verify that the TSF detects unencrypted data frames being sent between the allowlisted AP and non-allowlisted EUD.

Test 22.2:

Step 1: Deploy a non-allowlisted AP with no encryption.

Step 2: Connect an allowlisted EUD to AP and generate traffic.

Step 3: Verify that the TSF detects unencrypted data frames being between the non-allowlisted AP and allowlisted EUD.

Test 23: Detection of allowlisted EUD or AP that is using weak/outdated WLAN protocols and protocol implementations:

Step 1: Deploy an allowlisted AP that utilizes the 802.11g or older WLAN protocol.

Step 2: Verify that the TSF detects the weak/outdated WLAN protocol and generates an alert.

Test 24: Detection of extremely high numbers of client devices using a particular allowlisted AP:

Step 1: Deploy an allowlisted AP.

Step 2: Configure a threshold amount of client devices that can use a particular AP.

Step 3: Connect enough client devices to the AP to purposely exceed the defined threshold.

Step 4: Verify that the TSF detects when the client usage exceeds the threshold.

Test 25: Detection of a high number of failed attempts to join the WLAN in a short period of time:

Step 1: Deploy an allowlisted AP.

Step 2: Configure a threshold amount of connection attempts that can occur in a particular timeframe.

Step 3: Attempt to authenticate to the AP with enough client devices to purposely exceed the defined threshold.

Step 4: Verify that the TSF detects when the connection attempts within the specic timeframe exceeds the threshold.

Test 26: Detection of the use of active WLAN scanners (e.g. wardriving tools) to generate WLAN traffic:

Step 1: Deploy an allowlisted AP.

Step 2: Verify that the TSF detects when WLAN scanners are the source of WLAN traffic.

Test 27: Detection of the physical location of an identified WLAN threat by using triangulation:

Step 1: Deploy a non-allowlisted AP or EUD within range of the TSF.

Step 2: Verify that the TSF can track and locate the AP or EUD to within 25 feet.

Test 28: Detection of an SSID using weak/unsupported/disallowed encryption options:

Step 1: Deploy an allowlisted AP and configure its encryption options.

Step 2: Change the encryption options the AP advertises.

Step 3: Verify that the TSF detects when the AP's encryption options change.

Test 29: Detection of AP SSID larger than 32 bytes:

Step 1: Deploy an allowlisted AP and configure its SSID to be larger than 32 bytes.

Step 2: Configure a user defined signature on the WIDS to detect when an SSID is larger than 32 bytes.

Step 3: Verify that the TSF detects when the AP's SSID is larger than 32 bytes.

Test 30: Detection of excessive WPS negotiations:

Step 1: Deploy an allowlisted AP and permit WPS authentication.

Step 2: Configure a threshold amount of WPS connections that are allowed in a specific amount of time on the AP

Step 3: Verify that the TSF detects when the AP's WPS connection threshold has been exceeded.

For each WIDS event described in this test activity, the administrator first defined a policy to configured a WIDS alert for that event. The evaluator generated traffic in each case to matching the WIDS event. The evaluator confirmed the event was reported by the TOE both via the WebUI functionality and via audits sent to a remote syslog server.

2.1.15 Protected Audit Event Storage (NDcPP22e\WIDS10:FAU_STG_EXT.1)

2.1.15.1 NDcPP22e\WIDS10:FAU_STG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.15.2 NDcPP22e\WIDS10:FAU_STG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.15.3 NDcPP22e\WIDS10:FAU_STG_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

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

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

For distributed TOEs the evaluator shall examine the TSS to ensure it describes which TOE components are storing audit information locally and which components are buffering audit information and forwarding the information to

another TOE component for local storage. For every component the TSS shall describe the behaviour when local storage space or buffer space is exhausted.

Section 6.1 in the **ST** states that in the distributed TOE, the AP and vSZ-D components authenticate to the SZ/vSZ (Controller) and once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. The AP and vSZ-D buffer their audit data locally and forward it to the Controller via SSH in real time. All audit messages are propagated via the SZ/vSZ to an external audit server securely via IPsec in real-time. This buffer occurs in RAM. In the event that the buffer becomes full all new audits will be discarded although there is an age out mechanism to prevent this from happening. Any audits in the buffer that are not able to be sent to the controller within 30 minutes are deleted from the buffer. Alternatively, the audit messages can be stored in the Controller itself. When stored locally, and local storage is full, oldest audit records are overwritten. The Controller can store 14 archives of application logs with each log size of up to 100 MB. Only authorized administrators have access to the audit records.

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

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

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

Section “Enabling Other Secured Communication Services” of **Admin Guide** describes how to enable sending logs to a remote syslog server, via the web interface for the SmartZone controller. The syslog server is reachable through an IPsec tunnel and it will receive the logs in a secured way from the controller if the System IPsec feature is enabled. Audit logs from the AP and vSZ-D are sent to the Controller which stores the audit logs locally and sends them to the syslog server. The audit logs are not stored locally on the AP or vSZ-D. As indicated in Section 6.1 of the **ST**, once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. For AP and vSZ-D, the audit logs are not stored on the local server, they are buffered and sent to the controller in real-time. This buffer occurs in RAM. In the event buffer is full all new audits are discarded although there is an age out mechanism to prevent this from happening.

The external syslog port number must be 514. When an external syslog server is configured, all the audit data or events are sent to the external syslog server simultaneously. The SmartZone controller uses log rotation to overwrite oldest audit records to prevent local storage space becoming full. For the file system log file, a maximum of 14 archives of application logs are stored with each log up to 100 MB.

Sections “Configuring System IPsec using Preshared Key” and “Configuring System IPsec using Certificates” in the **Admin Guide** describe the steps to configure the IPsec tunnel that is used for the external syslog communication.

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

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

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

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

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

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

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

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

Test 1: The external audit server was utilizing rsyslogd version 8.16.0. The evaluator established a connection and demonstrated that the syslog messages were successfully sent to the syslog server through the encrypted channel. All of the audits demonstrated in NDcPP21:FAU_GEN.1 were collected from the syslog server.

Test 2: All audit data for all TOE components is stored on the SZ Controller. The evaluator verified that when the local audit storage was filled, the existing audit data was overwritten using the following rule: Overwrite oldest records first.

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

Test 4: This test was performed as part of test 1 and test 2.

2.1.16 Protected Local Audit Event Storage for Distributed TOEs (NDcPP22e:FAU_STG_EXT.4)

2.1.16.1 NDcPP22e10:FAU_STG_EXT.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator examines the TSS to confirm that it describes which TOE components store their security audit events locally and which send their security audit events to other TOE components for local storage. For the latter, the target TOE component(s) which store security audit events for other TOE components shall be identified. For every sending TOE component the corresponding receiving TOE component(s) need to be identified. For every transfer of audit information between TOE components it shall be described how the data is secured during transfer according to FTP_ITC.1 or FPT_ITT.1.

For each TOE component which does not store audit events locally by itself, the evaluator confirms that the TSS describes how the audit information is buffered before sending to another TOE component for local storage.

Section 6.1 in the **ST** states that in the distributed TOE, the AP and vSZ-D components authenticate to the SZ/vSZ (Controller) and once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. The AP and vSZ-D buffer their audit data locally and automatically forward it to the Controller via SSH in real time without any additional configuration needed. All audit messages are propagated via the SZ/vSZ to an external audit server securely via IPsec in real-time. Alternatively, the audit messages can be stored in the Controller itself. When stored locally, and local storage is full, oldest audit records are overwritten. The Controller can store 10 archives of application logs with each log size of up to 10 MB. Only authorized administrators have access to the audit records.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to ensure that it describes how the link between different TOE components is established if audit data is exchanged between TOE components for local storage. The guidance documentation shall describe all possible configuration options

for local storage of audit data and provide all instructions how to perform the related configuration of the TOE components.

The evaluator shall also ensure that the guidance documentation describes for every TOE component which does not store audit information locally how audit information is buffered before transmission to other TOE components.

Section "Enabling Other Secured Communication Services" of **Admin Guide** describes how to enable sending logs to a remote syslog server, via the web interface for the SmartZone controller. The syslog server is reachable through an IPsec tunnel and it will receive the logs in a secured way from the controller if the System IPsec feature is enabled. Audit logs from the AP and vSZ-D are sent to the Controller which stores the audit logs locally and sends them to the syslog server. The audit logs are not stored locally on the AP or vSZ-D. As indicated in Section 6.1 of the **ST**, once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. The AP and vSZ-D buffer their audit data locally and automatically forward it to the Controller via SSH in real time without any additional configuration needed.

The external syslog port number must be 514. When an external syslog server is configured, all the audit data or events are sent to the external syslog server simultaneously. The SmartZone controller uses log rotation to overwrite oldest audit records to prevent local storage space becoming full. For the file system log file, a maximum of 10 archives of application logs are stored with each log up to 10 MB.

Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in the **Admin Guide** describe the steps to configure the IPsec tunnel that is used for the external syslog communication.

Component Testing Assurance Activities: For at least one of each type of distributed TOE components (sensors, central nodes, etc.), the following tests shall be performed using distributed TOEs.

Test 1: For each type of TOE component, the evaluator shall perform a representative subset of auditable actions and ensure that these actions cause the generation of appropriately formed audit records. Generation of such records can be observed directly on the distributed TOE component (if there is appropriate interface), or indirectly after transmission to a central location.

Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the

TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

While performing these tests, the evaluator shall verify that the TOE behaviour observed during testing is consistent with the descriptions provided in the TSS and the Guidance Documentation. Depending on the TOE configuration, there might be a large number of different possible configurations. In such cases, it is acceptable to perform subset testing, accompanied by an equivalency argument describing the evaluator's sampling methodology.

Test 1 - The audits generated for each TOE component were captured as part of NDcPP22E:FAU_GEN.1.

Test 2 - The IPsec syslog connection was demonstrated in NDcPP22E:FTP_ITC.1.

Test 3 – The SSH distributed TOE channel in which audits are sent from the AP or vSZ-D to the SZ controller was demonstrated in NDcPP22E: FTP_ITT.1.

2.1.17 Protected Remote Audit Event Storage for Distributed TOEs (NDcPP22E:FAU_STG_EXT.5)

2.1.17.1 NDcPP22E:FAU_STG_EXT.5.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator examines the TSS to confirm that it describes which TOE components store their security audit events locally and which send their security audit events to other TOE components for local storage. For the latter, the target TOE component(s) which store security audit events for other TOE components shall be identified. For every sending TOE component the corresponding receiving TOE component(s) need to be identified. For every transfer of audit information between TOE components it shall be described how the data is secured during transfer according to FTP_ITC.1 or FPT_ITT.1. For each TOE component which does not store audit events locally by itself, the evaluator confirms that the TSS describes how the audit information is buffered before sending to another TOE component for local storage.

Section 6.1 in the **ST** states that in the distributed TOE, the AP and vSZ-D components authenticate to the SZ/vSZ (Controller) and once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. The AP and vSZ-D buffer their audit data locally and automatically forward it to the Controller via SSH in real time without any additional configuration needed. This buffer occurs in RAM. In the event that the buffer becomes

full all new audits will be discarded although there is an age out mechanism to prevent this from happening. Any audits in the buffer that are not able to be sent to the controller within 30 minutes are deleted from the buffer. All audit messages are propagated via the SZ/vSZ to an external audit server securely via IPsec in real-time. Alternatively, the audit messages can be stored in the Controller itself. When stored locally, and local storage is full, oldest audit records are overwritten. The Controller can store 10 archives of application logs with each log size of up to 10 MB. Only authorized administrators have access to the audit records.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to ensure that it describes how the link between different TOE components is established if audit data is exchanged between TOE components for local storage. The guidance documentation shall describe all possible configuration options for local storage of audit data and provide all instructions how to perform the related configuration of the TOE components. The evaluator shall also ensure that the guidance documentation describes for every TOE component which does not store audit information locally how audit information is buffered before transmission to other TOE components.

Section "Enabling Other Secured Communication Services" of **Admin Guide** describes how to enable sending logs to a remote syslog server, via the web interface for the SmartZone controller. The syslog server is reachable through an IPsec tunnel and it will receive the logs in a secured way from the controller if the System IPsec feature is enabled. Audit logs from the AP and vSZ-D are sent to the Controller which stores the audit logs locally and sends them to the syslog server. The audit logs are not stored locally on the AP or vSZ-D. As indicated in Section 6.1 of the **ST**, once an administrator allows the SZ/vSZ to communicate to the AP and vSZ-D they form a distributed TOE. The AP and vSZ-D buffer their audit data locally and automatically forward it to the Controller via SSH in real time without any additional configuration needed.

The external syslog port number must be 514. When an external syslog server is configured, all the audit data or events are sent to the external syslog server simultaneously. The SmartZone controller uses log rotation to overwrite oldest audit records to prevent local storage space becoming full. For the file system log file, a maximum of 10 archives of application logs are stored with each log up to 10 MB.

Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in the **Admin Guide** describe the steps to configure the IPsec tunnel that is used for the external syslog communication.

Component Testing Assurance Activities: For at least one of each type of distributed TOE components (sensors, central nodes, etc.), the following tests shall be performed using distributed TOEs.

Test 1: For each type of TOE component, the evaluator shall perform a representative subset of auditable actions and ensure that these actions cause the generation of appropriately formed audit records. Generation of such records can be observed directly on the distributed TOE component (if there is appropriate interface), or indirectly after transmission to a central location.

Test 2: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to the external audit server (as specified in FTP_ITC.1), the evaluator shall configure a trusted channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the

subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

Test 3: For each type of TOE component that, in the evaluated configuration, is capable of transmitting audit information to another TOE component (as specified in FTP_ITT.1 or FTP_ITC.1, respectively), the evaluator shall configure a secure channel and confirm that audit records generated as a result of actions taken by the evaluator are securely transmitted. It is sufficient to observe negotiation and establishment of the secure channel with the TOE component and the subsequent transmission of encrypted data to confirm this functionality. Alternatively, the following steps shall be performed: The evaluator induces audit record transmission, then reviews the packet capture around the time of transmission and verifies that no audit data is transmitted in the clear.

While performing these tests, the evaluator shall verify that the TOE behaviour observed during testing is consistent with the descriptions provided in the TSS and the Guidance Documentation. Depending on the TOE configuration, there might be a large number of different possible configurations. In such cases, it is acceptable to perform subset testing, accompanied by an equivalency argument describing the evaluator's sampling methodology.

Test 1 - The audits generated for each TOE component were captured as part of NDcPP22E:FAU_GEN.1.

Test 2 - The IPsec syslog connection was demonstrated in NDcPP22E:FTP_ITC.1.

Test 3 – The SSH distributed TOE channel in which audits are sent from the AP or vSZ-D to the SZ controller was demonstrated in NDcPP22E: FTP_ITT.1.

2.1.18 Wireless Intrusion Detection - Malicious Environmental Objects (WIDS10:FAU_WID_EXT.1)

2.1.18.1 WIDS10:FAU_WID_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.18.2 WIDS10:FAU_WID_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS describes how the TOE detects malicious APs/EUDs and whether the TOE supports automatic detection. The evaluator shall verify that the TSS includes how the TOE determines if a given SSID is authorized.

Section 6.1 of **ST** states the TOE can distinguish between benign and malicious APs and EUDs based on whether they have been allow-listed based on their MAC address. A WIDS rule for unauthorized SSIDs can be configured to detect a specific unauthorized SSID matching that is set by the administrator. A separate WIDS rule for any unauthorized SSIDs can be configured. In this case the authorized SSIDs are the ones explicitly configured for the WLANs managed by the TOE.

Component Guidance Assurance Activities: If TOE supports automatic detection, the evaluator shall verify that the operational guidance contains instructions for configuring the automatic detection metrics. The evaluator shall verify that the operational guidance provides instructions on how to configure SSIDs as authorized.

Section "Allowed Device Profile" of **Admin Guide** defines how to configure a device to be detected as authorized/unauthorized based on the AP/EUDs MAC address. Section "Different Rule types and Classification" contains information about the SSID WIDS rule which can detect a specific unauthorized SSID matching that is set by the administrator. A separate WIDS rule for any unauthorized SSIDs can be configured. In this case the authorized SSIDs are the ones explicitly configured for the WLANs managed by the TOE. The "Trusted Communication Channels" section of the **Admin Guide** demonstrates how to configure a WLAN with an SSID which in turn determines the authorized SSIDs.

Component Testing Assurance Activities: For test 1 and 2 below the evaluator shall verify that the TOE detects and appropriately classifies the APs and EUDs. It is acceptable if the TOE uses different but equivalent descriptors for the classification. If the TOE does not support automatic detection metrics and equates a non-allowlisted AP/EUD as malicious, than it is sufficient that the the classification given to the AP/EUD in step 1 is the same as in step 2. If the TOE supports automatic detection metrics and distinguishes between a non-allowlisted AP/EUD and a malicious AP/EUD, then the classification for the AP/EUD should differ between step 1 and step 2.

Test 1:

Step 1: Deploy a non-allowlisted AP in the area of the WIDS sensor, but take no action against the network. Verify that the AP is classified as non-authorized.

Step 2: Deploy a non-allowlisted AP in the area of the WIDS sensor and launch an attack against the network. This can be any variation of Fake AP, Spoof AP, Flood or DoS attack.

Step 3: Verify that the AP is classified as malicious.

Test 2:

Step 1: Deploy a non-allowlisted EUD in the area of the WIDS sensor, but take no action against the network. Verify that the EUD is classified as non-authorized.

Step 2: Launch an RF Flooding, DoD/DDoS, masqueraded or spoofing attack against authorized AP with an unauthorized EUD.

Step 3: Verify that the EUD is classified as malicious.

Test 3:

Step 1: Deploy an AP with an unauthorized SSID in the area of the WIDS sensor.

Step 2: Verify that the TOE detects the unauthorized SSID.

Test 1: In WIDS10:FAU_SAA.1_t1 the evaluator verified that a non-allowlisted AP is classified as non-authorized.

Throughout WIDS10:FAU_SAA.1 the evaluator shows that a policy can be configured to detect a Rogue device as malicious. WIDS10:FAU_SAA.1_t9.2 demonstrates spoofing an AP which was detected as malicious.

Test 2: In WIDS10:FAU_SAA.1_t2 the evaluator verified that a non-allowlisted EUD is classified as non-authorized.

Throughout WIDS10:FAU_SAA.1 the evaluator shows that a policy can be configured to detect a Rogue EUD as malicious. WIDS10:FAU_SAA.1_t9.1 demonstrates spoofing an EUD which was detected as malicious.

Test 3: Refer to WIDS10:FAU_SAA.1_t11 where the detection of an unauthorized SSID is tested.

2.1.19 Wireless Intrusion Detection - Passive Information Flow Monitoring (WIDS10:FAU_WID_EXT.2)

2.1.19.1 WIDS10:FAU_WID_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.19.2 WIDS10:FAU_WID_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.1.19.3 WIDS10:FAU_WID_EXT.2.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall verify that the TSS includes which channels the TOE can detect and monitor. Additionally, the TSS shall include whether the TOE simultaneously or nonsimultaneously monitors network traffic across these channels. The evaluator shall verify that the TSS includes information on if the sensors are completely passive, by default, or if the sensors ability to transmit data is configurable.

Section 6.1 of **ST** states the TOE monitor channels on the 2.4GHz and 5.0GHz network bands. The senor scans nonsimultaneoulsy, cycling through the channels based on the configured scan rate. The administrator can prevent transmissions by the sensor by placing the AP in a monitoring only mode.

Component Guidance Assurance Activities: The evaluator shall review the operational guidance for how to configure the TOE to monitor the channels as selected in the SFR. If the sensor ability to transmits data is configurable, the evaluator shall review the operational guidance for how to disable wireless transmissions from the sensor. The evaluator shall verify that the operational guidance provides instructions on how to specify and confirm that stateful frame capture and inspection is being performed.

Section “Monitoring Access Points (APs)” in **Admin Guide** describes how to put the APs into a monitoring only mode to prevent wireless transmissions.

Section “Creating an AP Zone” under header section “Configuring RUCKUS GRE and IPSec in WLAN-Concept” includes details on configuring the channels an AP uses, including Montioring APs.

There is no additional required steps to specify stateful frame inspection as it is performed by default once a monitoring AP is configured.

Component Testing Assurance Activities: Channels Monitored

Test 1: Channels on On 5GHz band

Step 1: Configure the TSF to monitor the channels as selected in the SFR.

Step 2: Deploy an AP on at least 2 different channels within the regulatory domain on 5GHz band.

Step 3: Verify that the AP gets detected on each channel tested.

Test 2: Channels on 2.4GHz band

Step 1: Configure the TSF to monitor the channels as selected in the SFR.

Step 2: Deploy AP on at least 2 different channels within the regulatory domain on 2.4GHz band.

Step 3: Verify that the AP gets detected on each channel tested.

Test 3: Channels on 4.9GHz band (if selected)

Step 1: Configure the TSF to monitor the channels specified in the SFR.

Step 2: Deploy AP and set to channels within the 4.9 GHz band outlined in the TSS.

Step 3: Verify that the AP gets detected on each channel tested.

Test 4: Non-standard channel frequencies (if selected)

Step 1: Configure the TSF to monitor the channels as selected in the SFR.

Step 2: Deploy AP on at least 2 different channels on non-standard channel frequencies.

Step 3: Verify that the AP gets detected on each channel tested.

Test 5: Channels outside regulatory domain (if selected)

Step 1: Configure the TSF to monitor the channels as selected in the SFR.

Step 2: Deploy an AP on at least 2 different channels outside the regulatory domain on 5GHz band.

Step 3: Deploy AP on at least 2 different channels outside the regulatory domain on 2.4GHz band.

Step 4: Verify that the AP gets detected on each channel tested.

(TD0611 applied)

Wireless Sensor Transmission of Data

If the TOE provides the ability to disable wireless transmission, the evaluator shall follow the operational guidance to configure the sensor to not transmit wirelessly. The evaluator shall then deploy a signal analyzer in order to check for wireless emanations from the TOE.

Repeat the two tests below, for both the 2.4GHz and the 5 GHz band.

Test 1:

Step 1: Boot a sensor and using the signal analyzer observe to check if any eminations are coming from the sensor.

Step 2: Verify that the signal analyzer does not pick up emanations from the sensor.

Test 2:

Step 1: During normal sensor operations, observe the analyzer for about 10 minutes to check if any eminations are coming from the sensor.

Step 2: Verify that the signal analyzer does not pick up emanations from the sensor.

Stateful Frame Inspection

Test 1:

Step 1: Deploy allowlisted AP.

Step 2: Connect an allowlisted EUD to the AP.

Step 3: Deploy a protocol analyzer or native capability within the WIDS Controller between the AP and EUD.

Step 4: Verify from the network traffic packet capture that all frames are being inspected to validate their connection state from the TSF.

Channels Monitored

Test 1: Since the TOE non-simultaneously monitors the channels within a band, the evaluator configured an AP on the 5GHz band to be on channel 149 first and then changed it to channel 161. The AP was detected successfully in both cases.

Test 2: Since the TOE non-simultaneously monitors the channels withing a band, the evaluator configured an AP on the 2.4GHz band to be on channel 6 first and then changed it to channel 11. The AP was detected successfully in both cases.

Test 3: Not applicable as the TOE does not support the monitoring of channels within the 4.9 GHz band.

Test 4: Not applicable as the TOE does not support the monitoring of non-standard channel frequencies.

Test 5: Not applicable as the TOE does not support the monitoring of channels outside the regulatory domain.

Wireless Sensor Transmission of Data

The evaluator configured a spectrum analyzer to collect the full sweep for Wi-Fi 2.4 GHz as configured by default in the scanner software. The evaluator first took a baseline with the analyzer (using clear write), but no device. Next the evaluator captured a baseline of the AP configured with a WLAN. Subsequently, the evaluator placed the AP in monitoring mode, in which the AP does not transmit data and rebooted the AP with the analyzer active (using max hold and clear write) and collected the test result (using max hold) allowing it to sit for about ten minutes. This test was repeated with 5 GHz.

This result was observed after restarting each TOE device five times in the enclosure and waiting for over 5 minutes after each restart while the spectrum analyzer was set to max hold so any unexpected signals would be

detected during that period. The evaluator also restarted each TOE device one time into recovery mode and observed the same lack of signals after the boot started.

Given the device with Wi-Fi disabled has the same signals after the boot started (over time given the max hold) as the baseline when no device is present in contrast with the early boot signals on the device, the evaluator concluded that disabling Wi-Fi effectively disables the radio.

Stateful Frame Inspection

The evaluator connected an Allowlisted EUD to an Allowlisted AP and used the TOE's built in, on demand, packet capture to collect the traffic relating to the AP.

The evaluator observed wireless frames as expected, indicating that the TOE was likely scanning all wireless frames in the claimed frequency bands. The detection of WIDS events for FAU_SAA.1 further confirmed that wireless frames were being scanned as expected.

2.2 Communication (FCO)

2.2.1 Component Registration Channel Definition (NDcPP22e/WIDS10:FCO_CPC_EXT.1)

2.2.1.1 NDcPP22e/WIDS10:FCO_CPC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.2 NDcPP22e/WIDS10:FCO_CPC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.2.1.3 NDcPP22e/WIDS10:FCO_CPC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below. In carrying out these activities the evaluator shall determine answers to the following questions based on a combination of documentation analysis and testing (possibly also using input from carrying out the Evaluation Activities for the relevant registration channel, such as FTP_TRP.1(2)/Join), and shall report the answers.

Questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities:

a) What stops [The intent of the phrasing 'what stops...' as opposed to 'what secures...' is for the evaluator to pursue the answer to its lowest level of dependency, i.e. a level at which the security can clearly be seen to depend on things that are under appropriate control. For example, a channel may be protected by a public key that is provided to the relying party in a self-signed certificate. This enables cryptographic mechanisms to be applied to provide authentication (and therefore invites an answer that 'the check on the public key certificate secures...'), but does not ultimately stop an attacker from apparently authenticating because the attacker can produce their own self-signed certificate. The question 'what stops an unauthorised component from successfully communicating...' focuses attention on what an attacker needs to do, and therefore pushes the answer down to the level of whether a self-signed certificate could be produced by an attacker. Similarly a well-known key, or a key that is common to a type of device rather than an individual device, may be used in a confidentiality mechanism but does not provide confidentiality because an attacker can find the well-known key or obtain his own instance of a device containing the non-unique key.] a component from successfully communicating with TOE components (in a way that enables it to participate as part of the TOE) before it has properly authenticated and joined the TOE?

b) What is the enablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

1) What stops anybody other than a Security Administrator from carrying out this step?

2) How does the Security Administrator know that they are enabling the intended component to join? (Identification of the joiner might be part of the enablement action itself or might be part of secure channel establishment, but it must prevent unintended joining of components)

c) What stops a component successfully joining if the Security Administrator has not carried out the enablement step; or, equivalently, how does the TOE ensure that an action by an authentic Security Administrator is required before a component can successfully join?

d) What stops a component from carrying out the registration process over a different, insecure channel?

e) If the FTP_TRP.1(2)/Join channel type is selected in FCO_CPC_EXT.1.2 then how do the registration process and its secure channel ensure that the data is protected from disclosure and provides detection of modification?

f) Where the registration channel does not rely on protection of the registration environment, does the registration channel provide a sufficient level of protection (especially with regard to confidentiality) for the data that passes over it?

g) Where the registration channel is subsequently used for normal internal communication between TOE components (i.e. after the joiner has completed registration), do any of the authentication or encryption features of the registration channel result in use of a channel that has weaker protection than the normal FPT_ITT.1 requirements for such a channel?

h) What is the disablement step? (Describe what interface it uses, with a reference to the relevant section and step in the operational guidance).

i) What stops a component successfully communicating with other TOE components if the Security Administrator has carried out the disablement step?

The evaluator shall examine the TSS to confirm that it:

a) Describes the method by which a Security Administrator enables and disables communications between pairs of TOE components.

b) Describes the relevant details according to the type of channel in the main selection made in FCO_CPC_EXT.1.2:

- First type: the TSS identifies the relevant SFR iteration that specifies the channel used

- Second type: the TSS (with support from the operational guidance if selected in FTP_TRP.1.3/Join) describes details of the channel and the mechanisms that it uses (and describes how the process ensures that the key is unique to the pair of components) - see also the Evaluation Activities for FTP_TRP.1(2)/Join.

The evaluator shall confirm that if any aspects of the registration channel are identified as not meeting FTP_ITC.1 or FPT_ITT.1, then the ST has also selected the FTP_TRP.1(2)/Join option in the main selection in FCO_CPC_EXT.1.2.

Section 6.2 of the **ST** states that the TOE requires the use of a dedicated channel for the AP and vSZ-D to register with a Controller. When an AP and vSZ-D discover a Controller via manual configuration or optionally via DHCP options, they will attempt to connect. Upon successful connection to the Controller, no data transfer is allowed between the Controller and AP/vSZD. An administrator must manually approve the AP/vSZ-D which enables data transfer between the Controller and AP/vSZ-D. An administrator also has the ability to manually revoke the connection between Controller and AP/vSZ-D. After registering multiple APs to the same controller the administrator has the option to enable a mesh configuration. In this configuration, any distributed TOE traffic is routed wirelessly from a mesh AP to a root AP wired directly to the Controller.

Section 6.9 of the **ST** states that the TOE supports distributed TOE communication. The SmartZone controller is configured first, then other components, AP and vSZ-D are configured to join to the controller by assigning the IP address of the controller. After approval of the join request those appliances are managed by the SmartZone

Controller. This initial registration is performed over a dedicated channel. After registration, SSH is used for all management of the distributed TOE components (AP and vSZ-D) by the SmartZone controller and IPsec is used for the data tunnel. This initial configuration matches the claim of FPT_TRP.1/Join in the main selection of FCO_CPC_EXT.1.2.

Component Guidance Assurance Activities: (Note: paragraph 274 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

The evaluator shall examine the guidance documentation to confirm that it contains instructions for enabling and disabling communications with any individual component of a distributed TOE. The evaluator shall confirm that the method of disabling is such that all other components can be prevented from communicating with the component that is being removed from the TOE (preventing the remaining components from either attempting to initiate communications to the disabled component, or from responding to communications from the disabled component).

The evaluator shall examine the guidance documentation to confirm that it includes recovery instructions should a connection be unintentionally broken during the registration process.

If the TOE uses a registration channel for registering components to the TOE (i.e. where the ST author uses the FTP_ITC.1/FPT_ITT.1 or FTP_TRP.1(2)/Join channel types in the main selection for FCO_CPC_EXT.1.2) then the evaluator shall examine the Preparative Procedures to confirm that they:

a) describe the security characteristics of the registration channel (e.g. the protocol, keys and authentication data on which it is based) and shall highlight any aspects which do not meet the requirements for a steady-state inter-component channel (as in FTP_ITC.1 or FPT_ITT.1)

b) identify any dependencies between the configuration of the registration channel and the security of the subsequent inter-component communications (e.g. where AES-256 inter-component communications depend on transmitting 256 bit keys between components and therefore rely on the registration channel being configured to use an equivalent key length)

c) identify any aspects of the channel can be modified by the operational environment in order to improve the channel security, and shall describe how this modification can be achieved (e.g. generating a new key pair, or replacing a default public key certificate).

As background for the examination of the registration channel description, it is noted that the requirements above are intended to ensure that administrators can make an accurate judgement of any risks that arise from the default registration process. Examples would be the use of self-signed certificates (i.e. certificates that are not chained to an external or local Certification Authority), manufacturer-issued certificates (where control over aspects such as revocation, or which devices are issued with recognised certificates, is outside the control of the operational environment), use of generic/non-unique keys (e.g. where the same key is present on more than one instance of a device), or well-known keys (i.e. where the confidentiality of the keys is not intended to be strongly protected - note that this need not mean there is a positive action or intention to publicise the keys).

In the case of a distributed TOE for which the ST author uses the FTP_TRP.1(2)/Join channel type in the main selection for FCO_CPC_EXT.1.2 and the TOE relies on the operational environment to provide security for some aspects of the registration channel security then there are additional requirements on the Preparative Procedures as described in section 3.4.1.2.

Section "AP Models that Support FIPS Mode" in the **Admin Guide** indicates that while TLS is used to perform the initial discovery of the SZ controller, this TLS is not claimed or tested in the evaluation due to limited certificate verification capabilities during the registration of the TOE components. The TOE requires the use of a dedicated channel for the AP and vSZ-D to register with the Controller. The administrator must perform the registration of TOE components in a controlled environment in which there is a segregated network with only TOE components present.

Section "Joining vSZ-D to the vSZ Controller" in the **Admin Guide** describes the steps for joining of the vSZ-D to the vSZ Controller. The IP address of the vSZ controller is configured on the vSZ-D which then initiates the attempt to connect to the controller. The administrator then selects and approves the vSZ-D via the controller's Web UI. Once the registration process is approved, an SSH connection is automatically established and all communication between the vSZ-D and the vSZ Controller is over SSH. The connection with the vSZ-D can similarly be disabled if the administrator selects the vSZ-D and then clicks the 'Delete' button. Deleting the vSZ-D from the controller prevents all other components from communicating with the vSZ-D. If the connection between the vSZ-D and vSZ is broken then it resumes back automatically and no manual intervention is required.

Section "Joining AP to (v)SZ Controller" in the **Admin Guide** describes the steps for joining of the AP to the SZ Controller. The IP address of the SZ controller is configured on the AP which then initiates the attempt to connect to the controller. The administrator then selects and approves the AP via the controller's Web UI. Once the registration process is approved, an SSH connection is automatically established between the AP and the SZ Controller. The connection with the AP can similarly be disabled if the administrator selects the AP and then clicks the 'Delete' button. Deleting the AP from the controller prevents all other components from communicating with the AP. If the connection is broken it will be resumed/reattempted without any user intervention.

Section "Configuring Regular Mesh" in the **Admin Guide** describes how to setup a wireless mesh configuration with a Root AP and a Mesh AP. This configuration does not change the distributed TOE channel's encryption.

Section "Management Channel Between AP/vSZ-D and Controller" in the **Admin Guide** further describes the SSH communication between the AP and SZ controller and between the vSZ-D and the vSZ controller. The SSH algorithms, parameters and rekey limits are not configurable. The communication is only through public key auth (no password based authentication).

Section "Administrating the controller remotely" in the **Admin Guide** provides all of the SSH algorithms and key exchange methods supported by the TOE in its implementations as both client and server for remote authentication and for communication between the distributed TOE components.

Before any communication with a wireless client occurs, an IPsec tunnel is formed between the AP and SZ or the vSZ-D. Once the wireless client is authenticated, all user data traffic is protected using IPsec. Section "Configuring

Ruckus GRE and IPsec in the WLAN" in the **Admin Guide** states that Ruckus GRE and IPsec is a configuration of the IPsec tunnel between the AP and the SZ or vSZ-D and provides instructions for configuring the IPsec tunnel between these components. If the IPsec connection between AP and vSZ-D is unintentionally broken, it is recovered automatically and no manual intervention is required.

Component Testing Assurance Activities: (Note: paragraph 274 lists questions for which the evaluator needs to determine and report answers through the combination of the TSS, Guidance Documentation, and Tests Evaluation Activities.)

The evaluator shall carry out the following tests:

a) Test 1.1: the evaluator shall confirm that an IT entity that is not currently a member of the distributed TOE cannot communicate with any component of the TOE until the non-member entity is enabled by a Security Administrator for each of the non-equivalent TOE components [An 'equivalent TOE component' is a type of distributed TOE component that exhibits the same security characteristics, behaviour and role in the TSF as some other TOE component. In principle a distributed TOE could operate with only one instance of each equivalent TOE component, although the minimum configuration of the distributed TOE may include more than one instance (see discussion of the minimum configuration of a distributed TOE, in section B.4). In practice a deployment of the TOE may include more than one instance of some equivalent TOE components for practical reasons, such as performance or the need to have separate instances for separate subnets or VLANs.] that it is required to communicate with (non-equivalent TOE components are as defined in the minimum configuration for the distributed TOE)

b) Test 1.2: the evaluator shall confirm that after enablement, an IT entity can communicate only with the components that it has been enabled for. This includes testing that the enabled communication is successful for the enabled component pair, and that communication remains unsuccessful with any other component for which communication has not been explicitly enabled.

Some TOEs may set up the registration channel before the enablement step is carried out, but in such a case the channel must not allow communications until after the enablement step has been completed.

The evaluator shall repeat Tests 1.1 and 1.2 for each different type of enablement process that can be used in the TOE.

c) Test 2: The evaluator shall separately disable each TOE component in turn and ensure that the other TOE components cannot then communicate with the disabled component, whether by attempting to initiate communications with the disabled component or by responding to communication attempts from the disabled component.

d) Test 3: The evaluator shall carry out the following tests according to those that apply to the values of the main (outer) selection made in the ST for FCO_CPC_EXT.1.2.

1) If the ST uses the first type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_ITC.1 or FPT_ITT.1 according to the second selection - the evaluator shall ensure that the test coverage for these SFRs includes their use in the registration process.

2) If the ST uses the second type of communication channel in the selection in FCO_CPC_EXT.1.2 then the evaluator tests the channel via the Evaluation Activities for FTP_TRP.1(2)/Join.

3) If the ST uses the 'no channel' selection then no test is required.

e) Test 4: The evaluator shall perform one of the following tests, according to the TOE characteristics identified in its TSS and operational guidance:

1) If the registration channel is not subsequently used for inter-component communication, and in all cases where the second selection in FCO_CPC_EXT.1.2 is made (i.e. using FTP_TRP.1(2)/Join) then the evaluator shall confirm that the registration channel can no longer be used after the registration process has completed, by attempting to use the channel to communicate with each of the endpoints after registration has completed.

2) If the registration channel is subsequently used for intercomponent communication then the evaluator shall confirm that any aspects identified in the operational guidance as necessary to meet the requirements for a steady-state intercomponent channel (as in FTP_ITC.1 or FPT_ITT.1) can indeed be carried out (e.g. there might be a requirement to replace the default key pair and/or public key certificate).

f) Test 5: For each aspect of the security of the registration channel that operational guidance states can be modified by the operational environment in order to improve the channel security (cf. AGD_PRE.1 refinement item 2 in (cf. the requirements on Preparative Procedures in 3.5.1.2), the evaluator shall confirm, by following the procedure described in the operational guidance, that this modification can be successfully carried out.

Test 1.1 and 1.2 - There exists no means for communication to occur between the TOE components before registration. The evaluator viewed via wireshark that no communication was occurring between the SZ controller and AP or vSZ-D before registration. There also exists no interface for any TOE component to attempt to communicate with any other TOE component before registration. The registration of the AP or vSZ-D to the SZ controller was demonstrated in FTP_TRP.1/Join test 1. After the join the evaluator saw the distributed TOE communication which was demonstrated in FPT_ITT.1 test 3.

Test 2 - The evaluator separately deleted the AP and the vSZ-D from the SZ controller and viewed that the device was no longer registered and that all distributed communications between the TOE components had stopped as a result of the deletion. There exists no interface to attempt to communicate between the components once deleted other than attempting to reregister.

Test 3 – This test was met by NDcPP22e:FTP_TRP.1/Join.

Test 4 - This test is met by design. There is no means to reuse a registration channel after an AP or vSZ-D have been registered to the SZ controller. The evaluator demonstrated that the TOE component could not be re-approved for registration on the SZ controller. The evaluator attempted to access the initial WebUI configuration of the AP

which was used during install to configure the SZ controller. The evaluator viewed that access was locked down due to registration to the SZ controller, thus a new SZ controller could not be set unless the AP was unregistered. The evaluator also viewed that the setup command on the vSZ-D that was used during install to assign a SZ controller was not available, thus a new SZ controller could not be set unless the vSZ-D was unregistered. All of these things show that there is no means to reuse the FTP_TRP.1/Join channel once registered.

Test 5 –The TOE does not claim any means to improve the security of the distributed TOE channel.

2.3 Cryptographic support (FCS)

2.3.1 Cryptographic Key Generation (NDcPP22e:FCS_CKM.1)

2.3.1.1 NDcPP22e:FCS_CKM.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 in the **ST** states that the TOE supports cryptographic key generation for the following schemes: RSA schemes using key sizes of 2048 bits or greater, ECC schemes using NIST curves P-256, P-384, and P-521, FFC schemes using key sizes of 2048 bits or greater, FFC safe-primes schemes using Diffie-Hellman group 14 that meet RFC 3526, Section 3, WPA2/WPA3 128 bit cryptographic key derivation.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode automatically limits the key sizes to those supported by the TOE.

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

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

Key Generation for FIPS PUB 186-4 RSA Schemes

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

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

a) Random Primes:

- Provable primes

- Probable primes

b) Primes with Conditions:

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

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

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

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

Key Generation for Elliptic Curve Cryptography (ECC)

FIPS 186-4 ECC Key Generation Test

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

FIPS 186-4 Public Key Verification (PKV) Test

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

Key Generation for Finite-Field Cryptography (FFC)

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

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

- Primes q and p shall both be provable primes

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

and two ways to generate the cryptographic group generator g:

- Generator g constructed through a verifiable process

- Generator g constructed through an unverifiable process.

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

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

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

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

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

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

- g != 0,1

- q divides p-1

- g^q mod p = 1

- g^x mod p = y

for each FFC parameter set and key pair.

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.2 Cryptographic Key Generation (Symmetric Keys for WPA2 Connections) (WLANAS10:FCS_CKM.1/WPA)

2.3.2.1 WLANAS10:FCS_CKM.1.1/WPA

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The cryptographic primitives will be verified through evaluation activities specified elsewhere in this PPModule. The evaluator will verify that the TSS describes how the primitives defined and implemented by this PP-Module are used by the TOE in establishing and maintaining secure connectivity to the wireless clients. This description will include how the GTK and PTK are generated or derived. The TSS will also provide a description of the developer's methods of assuring that their implementation conforms to the cryptographic standards; this includes not only testing done by the developing organization, but also proof of third-party testing that is performed (e.g. WPA2 certification). The evaluator will ensure that the description of the testing methodology is of sufficient detail to determine the extent to which the details of the protocol specifics are tested.

Section 6.3 of the **ST** states the TOE supports cryptographic key distribution for 802.11 PMK key reception from an 802.1X authentication server, AES Key Wrap in EAPOL-Key frame and Group Key Handshake. In the distributed TOE deployment, AP acts as the authenticator and derives the 802.11 keys. With WPA2/WPA3-enterprise connections via 802.1X, all communications between the AP and wireless client is encapsulated in EAP and all traffic between AP and the controller is secured via SSH. The controller to RADIUS server communication is secured via RADSEC. The TOE supports standard WPA2/WPA3-enterprise with 802.11i where 4 way EAP handshake is between the Authenticator (AP) and the wireless client.

AP sends the EAPOL identity to the wireless client, which responds with a radius access request. Upon successful authentication, radius access accept message is received by the client. A PMK is then generated by the RADIUS server for Authenticator and client after which a PTK is derived. The Authenticator also generates a GTK. If the 4-way handshake fails, the client will not be allowed to access the network or connect to the AP.

Upon a successful 4-way handshake, the Authenticator will allow for WLAN data to pass through the system to the Controller in a tunnel architecture or to intended destination in distributed architecture.

The PTK (total 384 bits) is derived into three parts. The second part is KEK and used to encrypt GTK to be sent as 3rd message in WPA2/WPA3 handshake. Third part is TK, which is actually used to encrypt/decrypt communication between both AP and Client.

The Ruckus implementation of PTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.3 → Pairwise key hierarchy (PTK derivation using the PRF function)

The Ruckus implementation of GTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.4 --> Group key hierarchy (GTK derivation using the PRF function)

When the RADIUS protocol is used between the TOE and the authentication server, the MS-MPPE-Recv-Key attribute (vendor-id = 17; see Section 2.4.3 in IETF RFC 2548-1999 [B30]) is used to transport the PMK to TOE. The GTK is distributed by using AES Key Wrap in an EAPOL-Key frame in accordance with the 802.11-2012 standard for the packet format and timing considerations.

Certification testing performed by the Wi-Fi Alliance demonstrates that the TOE implements the IEEE 802.11-2012 standard correctly. Refer to the Wi-Fi Alliance certificates for compliance, https://www.wi-fi.org/certification.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will perform the following test using a packet sniffing tool to collect frames between the TOE and a wireless client:

Step 1: The evaluator will configure the AP to an unused channel and configure the WLAN sniffer to sniff only on that channel (i.e., lock the sniffer on the selected channel). The sniffer should also be configured to filter on the MAC address of the TOE and client.

Step 2: The evaluator will configure the TOE to communicate with a WLAN client using IEEE 802.11-2020 and a 256-bit (64 hex values 0-f) pre-shared key, setting up the connections as described in the operational guidance. The pre-shared key is only used for testing.Step 3: The evaluator will start the sniffing tool, initiate a connection between the TOE and WLAN client, and allow the TOE to authenticate, associate, and successfully complete the four-way handshake with the client.

Step 4: The evaluator will set a timer for one minute, at the end of which the evaluator will disconnect the client from the TOE and stop the sniffer.

Step 5: The evaluator will identify the four-way handshake frames (denoted EAPOL-key in Wireshark captures) and derive the PTK from the four-way handshake frames and pre-shared key as specified in IEEE 802.11-2020.

Step 6: The evaluator will select the first data frame from the captured packets that was sent between the client and TOE after the four-way handshake successfully completed and without the frame control value 0x4208 (the first two bytes are 08 42). The evaluator will use the PTK to decrypt the data portion of the packet as specified in IEEE 802.11-2020 and verify that the decrypted data contains ASCII-readable text.

Step 7: The evaluator will repeat Step 6 for the next two data frames between the TOE and client, and without frame control value 0x4208.

Additionally, the evaluator will test the PRF function using the test vectors from:

- Section 2.4 'The PRF Function - PRF (key, prefix, data, length)' of the IEEE 802.11-02/362r6 document 'Proposed Test vectors for IEEE 802.11 TGi' dated September 10, 2002

- Annex J.3 'PRF reference implementation and test vectors' of IEEE 802.11-2020

Test 1 – The evaluator configured the TOE as specified in steps 1 and 2 above, started a packet capture and then initiated a connection between the TOE and WLAN client. After 1 minute, the evaluator disconnected, stopped the packet capture and then decrypted the first 3 data frames of the packet capture and verified that they contained human readable ACSll text.

Test 2 – Regarding the PRF function of IEEE 802.11-2012, the TOE has WiFi Alliance certificates as identified in Section 1.1.2.

2.3.3 Cryptographic Key Establishment (NDcPP22e:FCS_CKM.2)

2.3.3.1 NDcPP22e:FCS_CKM.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Scheme | SFR | Service

RSA | FCS_TLSS_EXT.1 | Administration

ECDH | FCS_SSHC_EXT.1 | Audit Server

ECDH | FCS_IPSEC_EXT.1 | Authentication Server

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

(TD0580 applied)

Section 6.3 of the **ST** states that the TOE supports cryptographic key generation for the following schemes: RSA schemes using key sizes of 2048 bits or greater, ECC schemes using NIST curves P-256, P-384, and P-521, FFC schemes using key sizes of 2048 bits or greater, FFC schemes using Diffie-Hellman group 14 that meet RFC 3526, Section 3 and WPA2/WPA3 128 bit cryptographic key derivation.

The table below identifies the usage for each key establishment scheme supported by the TOE:

Security Function	Communication Type	Key Establishment Scheme
Remote administration via the Controller	TLS/HTTPS	RSA Schemes ECC Schemes FFC Schemes
Remote administration via the Controller; Control/Management channel between the Controller (SZ) and the AP and between the Controller (SZ) and the vSZ-D.	SSH	RSA Schemes ECC Schemes FFC Safe-Primes (DH-14)
Trusted channel for Syslog; Data channel between the Controller and the AP and between the vSZ-D and the AP	IPsec	RSA Schemes ECC Schemes FFC Safe-Primes (DH-14)
Trusted channel for authentication services	RadSec	RSA Schemes ECC Schemes FFC Schemes

Trusted channel for WLAN clients	WPA3/WPA2	WPA3/WPA2 128 bit Key derivation

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section "Using FIPS-related CLI commands" of **Admin Guide** describes the steps to enable FIPS mode which ensures that the key sizes are restricted to those supported by the TOE.

Component Testing Assurance Activities: Key Establishment Schemes

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

SP800-56A Key Establishment Schemes

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

Function Test

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

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

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

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

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

Validity Test

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

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

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

RSA-based key establishment

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

FFC Schemes using 'safe-prime' groups

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

(TD0580 applied)

For RSA-based key establishment, the evaluator performed testing of the protocols using RSA (TLS, SSH, IPsec, and RadSec) using known good implementations when testing the protocols for their corresponding SFRs. For FFC Schemes using 'safe-prime' groups the evaluator performed testing of the protocols using DH-14 (SSH and IPsec)

using known good implementations when testing the protocols for their corresponding SFRs. Refer to the CAVP certificates identified in Section 1.1.2 for the remaining testing.

2.3.4 Cryptographic Key Distribution (GTK) (WLANAS10:FCS_CKM.2/GTK)

2.3.4.1 WLANAS10:FCS_CKM.2.1/GTK

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will check the TSS to ensure that it describes how the GTK is wrapped prior to being distributed using the AES implementation specified in this PP-Module, and also how the GTKs are distributed when multiple clients connect to the TOE.

Section 6.3 of the **ST** states the TOE supports cryptographic key distribution for 802.11 PMK key reception from an 802.1X authentication server, AES Key Wrap in EAPOL-Key frame and Group Key Handshake. In the distributed TOE deployment, AP acts as the authenticator and derives the 802.11 keys. With WPA2/WPA3-enterprise connections via 802.1X, all communications between the AP and wireless client is encapsulated in EAP and all traffic between AP and the controller is secured via SSH. The controller to RADIUS server communication is secured via RADSEC. The TOE supports standard WPA2/WPA3-enterprise with 802.11i where 4 way EAP handshake is between the Authenticator (AP) and the wireless client.

AP sends the EAPOL identity to the wireless client, which responds with a radius access request. Upon successful authentication, radius access accept message is received by the client. A PMK is then generated by the RADIUS server for Authenticator and client after which a PTK is derived. The Authenticator also generates a GTK. If the 4-way handshake fails, the client will not be allowed to access the network or connect to the AP.

Upon a successful 4-way handshake, the Authenticator will allow for WLAN data to pass through the system to the Controller in a tunnel architecture or to intended destination in distributed architecture.

The PTK (total 384 bits) is derived into three parts. The second part is KEK and used to encrypt GTK to be sent as 3rd message in WPA2/WPA3 handshake. Third part is TK, which is actually used to encrypt/decrypt communication between both AP and Client.

The Ruckus implementation of PTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.3 → Pairwise key hierarchy (PTK derivation using the PRF function)

The Ruckus implementation of GTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)

- Section 11.6.1.4 --> Group key hierarchy (GTK derivation using the PRF function)

When the RADIUS protocol is used between the TOE and the authentication server, the MS-MPPE-Recv-Key attribute (vendor-id = 17; see Section 2.4.3 in IETF RFC 2548-1999 [B30]) is used to transport the PMK to TOE. The GTK is distributed by using AES Key Wrap in an EAPOL-Key frame in accordance with the 802.11-2012 standard for the packet format and timing considerations.

Certification testing performed by the Wi-Fi Alliance demonstrates that the TOE implements the IEEE 802.11-2012 standard correctly. Refer to the Wi-Fi Alliance certificates for compliance, https://www.wi-fi.org/certification.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will perform the following test using a packet sniffing tool to collect frames between a wireless client and the TOE (which may be performed in conjunction with the evaluation activity for FCS_CKM.1/PMK.

To fully test the broadcast and multicast functionality, these steps will be performed as the evaluator connects multiple clients to the TOE. The evaluator will ensure that GTKs established are sent to the appropriate participating clients.

Step 1: The evaluator will configure the AP to an unused channel and configure the WLAN sniffer to sniff only on that channel (i.e., lock the sniffer on the selected channel). The sniffer should also be configured to filter on the MAC address of the TOE and client.

Step 2: The evaluator will configure the TOE to communicate with the client using IEEE 802.11-2020 and a 256-bit (64 hex values 0-f) pre-shared key, setting up the connections as described in the operational guidance. The pre-shared key is only used for testing.

Step 3: The evaluator will start the sniffing tool, initiate a connection between the TOE and client, and allow the client to authenticate, associate and successfully complete the four-way handshake with the TOE.

Step 4: The evaluator will set a timer for one minute, at the end of which the evaluator will disconnect the TOE from the client and stop the sniffer.

Step 5: The evaluator will identify the four-way handshake frames (denoted EAPOL-key in Wireshark captures) and derive the PTK and GTK from the four-way handshake frames and pre-shared key as specified in IEEE 802.11-2020.

Step 6: The evaluator will select the first data frame from the captured packets that was sent between the TOE and client after the four-way handshake successfully completed, and with the frame control value 0x4208 (the first two bytes are 08 42). The evaluator will use the GTK to decrypt the data portion of the selected packet as specified in IEEE 802.11-2020 and verify that the decrypted data contains ASCII-readable text.

Step 7: The evaluator will repeat Step 6 for the next two data frames with frame control value 0x4208.

The evaluator will also perform the following testing based on the supported GTK distribution methods:

AES Key Wrap (AES-KW Tests)

Test 1: The evaluator will test the authenticated encryption functionality of AES-KW for each combination of the following input parameter lengths:

128 and 256 bit key encryption keys (KEKs)

Three plaintext lengths:

1. One of the plaintext lengths will be two semi-blocks (128 bits).

2. One of the plaintext lengths will be three semi-blocks (192 bits).

3. The third data unit length will be the longest supported plaintext length less than or equal to 64 semi-blocks (4096 bits).

For each combination, generate a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KW authenticated encryption. To determine correctness, the evaluator will use the same key and plaintext values and encrypt them using a known good implementation of AES-KW authenticatedencryption, and ensure that the resulting respective ciphertext values are identical.

Test 2: The evaluator will test the authenticated-decryption functionality of AES-KW using the same test as for authenticated-encryption, replacing plaintext values with ciphertext values and AES-KW authenticated-encryption with AES-KW authenticated-decryption. Additionally, the evaluator will modify one byte of the ciphertext, attempt to decrypt the modified ciphertext, and ensure that a failure is returned rather than plaintext.

AES Key Wrap with Padding (AES-KWP Tests)

Test 1: The evaluator will test the authenticated-encryption functionality of AES-KWP for each combination of the following input parameter lengths:

128 and 256 bit key encryption keys (KEKs)

Three plaintext lengths. One plaintext length will be one octet. One plaintext length will be 20 octets (160 bits). One plaintext length will be the longest supported plaintext length less than or equal to 512 octets (4096 bits).

using a set of 100 key and plaintext pairs and obtain the ciphertext that results from AES-KWP authenticated encryption. To determine correctness, the evaluator will use the AES-KWP authenticatedencryption function of a known good implementation.

Test 2: The evaluator will test the authenticated-decryption functionality of AES-KWP using the same test as for AES-KWP authenticated-encryption, replacing plaintext values with ciphertext values and AES-KWP authenticated-encryption with AES-KWP authenticated-decryption. Additionally, the evaluator will modify one byte of the ciphertext, attempt to decrypt the modified ciphertext, and ensure that a failure is returned rather than plaintext.

See Test Case WLANAS10:FCS_CKM.1/WPA where this same test has already been performed.

In regard to the AES-KW claimed, the TOE has CAVP certificates as identified in Section 1.1.2.

2.3.5 Cryptographic Key Distribution (PMK) (WLANAS10:FCS_CKM.2/PMK)

2.3.5.1 WLANAS10:FCS_CKM.2.1/PMK

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will examine the TSS to determine that it describes how the PMK is transferred (that is, through what EAP attribute) to the TOE.

Section 6.3 of the **ST** states the TOE supports cryptographic key distribution for 802.11 PMK key reception from an 802.1X authentication server, AES Key Wrap in EAPOL-Key frame and Group Key Handshake. In the distributed TOE deployment, AP acts as the authenticator and derives the 802.11 keys. With WPA2/WPA3-enterprise connections via 802.1X, all communications between the AP and wireless client is encapsulated in EAP and all traffic between AP and the controller is secured via SSH. The controller to RADIUS server communication is secured via RADSEC. The TOE supports standard WPA2/WPA3-enterprise with 802.11i where 4 way EAP handshake is between the Authenticator (AP) and the wireless client.

AP sends the EAPOL identity to the wireless client, which responds with a radius access request. Upon successful authentication, radius access accept message is received by the client. A PMK is then generated by the RADIUS server for Authenticator and client after which a PTK is derived. The Authenticator also generates a GTK. If the 4-way handshake fails, the client will not be allowed to access the network or connect to the AP.

Upon a successful 4-way handshake, the Authenticator will allow for WLAN data to pass through the system to the Controller in a tunnel architecture or to intended destination in distributed architecture.

The PTK (total 384 bits) is derived into three parts. The second part is KEK and used to encrypt GTK to be sent as 3rd message in WPA2/WPA3 handshake. Third part is TK, which is actually used to encrypt/decrypt communication between both AP and Client.

The Ruckus implementation of PTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.3 → Pairwise key hierarchy (PTK derivation using the PRF function)

The Ruckus implementation of GTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.4 --> Group key hierarchy (GTK derivation using the PRF function)

When the RADIUS protocol is used between the TOE and the authentication server, the MS-MPPE-Recv-Key attribute (vendor-id = 17; see Section 2.4.3 in IETF RFC 2548-1999 [B30]) is used to transport the PMK to TOE. The GTK is

distributed by using AES Key Wrap in an EAPOL-Key frame in accordance with the 802.11-2012 standard for the packet format and timing considerations.

Certification testing performed by the Wi-Fi Alliance demonstrates that the TOE implements the IEEE 802.11-2012 standard correctly. Refer to the Wi-Fi Alliance certificates for compliance, https://www.wi-fi.org/certification.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will establish a session between the TOE and a RADIUS server according to the configuration guidance provided. The evaluator will then examine the traffic that passes between the RADIUS server and the TOE during a successful attempt to connect a wireless client to the TOE to determine that the PMK is not exposed.

The evaluator established a connection between the SZ controller and RADIUS server and started a packet capture. The evaluator then examined the traffic and determined that the PMK is not exposed.

2.3.6 Cryptographic Key Destruction (NDcPP22e:FCS_CKM.4)

2.3.6.1 NDcPP22e:FCS_CKM.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Note that where selections involve 'destruction of reference' (for volatile memory) or 'invocation of an interface' (for non-volatile memory) then the relevant interface definition is examined by the evaluator to ensure that the interface supports the selection(s) and description in the TSS. In the case of non-volatile memory the evaluator

includes in their examination the relevant interface description for each media type on which plaintext keys are stored. The presence of OS-level and storage device-level swap and cache files is not examined in the current version of the Evaluation Activity.

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

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

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

Section “8 Key Clearing” of the **ST** contains a table that identifies and describes all secret and private keys and Critical Security Parameters (CSPs), provides their storage location and describes how and when they are cleared. Zeroization is performed when setting the Controller out of or into a FIPS-Approved mode of operation using the “fips enable/disable” command. This is required so that keys and other sensitive information established in one mode cannot be used in another. Other than changing the FIPS mode by using the fips enable/disable command on the Controller, zeroization can happen if the CLI command "fips zeroization" is issued on vSZ-D or “zeroize-all csp” command on AP (via controller).

For secure connections, keys can be stored in RAM. Keys stored in RAM are destroyed with a power cycle/reboot on all TOE components. On the SZ and vSZ-D, when the connection (session) is terminated, the keys are erased by writing zeros in the corresponding RAM location. On the AP, when the session is terminated, the keys are erased with a pseudo random pattern written in the corresponding RAM location. When keys are stored in Flash and zeroized via command, the keys are erased in a similar way to the keys found in RAM. On the SZ and vSZ-D, the keys are erased by writing zeros in the corresponding Flash location. On the AP, the keys are erased with a pseudo random pattern written in the corresponding Flash location. This is represented by session termination in the table below.

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

For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are

logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command [Where TRIM is used then the TSS and/or guidance documentation is also expected to describe how the keys are stored such that they are not inaccessible to TRIM, (e.g. they would need not to be contained in a file less than 982 bytes which would be completely contained in the master file table)] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

There are no situations that could prevent or delay key destruction described in the TSS of the **ST** or in the **Admin Guide**.

Component Testing Assurance Activities: None Defined

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

2.3.7.1 NDcPP22e:FCS_COP.1.1/DataEncryption

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the **ST** states that The TOE performs encryption and decryption using AES in CBC CCMP, CTR, and GCM mode with key sizes of either 128, 192 (CBC and GCM), or 256.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode automatically limits the key sizes to those supported by the TOE.

Component Testing Assurance Activities: AES-CBC Known Answer Tests

There are four Known Answer Tests (KATs), described below. In all KATs, the plaintext, ciphertext, and IV values shall be 128-bit blocks. The results from each test may either be obtained by the evaluator directly or by supplying

the inputs to the implementer and receiving the results in response. To determine correctness, the evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

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

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

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

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

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

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

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

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

AES-CBC Multi-Block Message Test

The evaluator shall test the encrypt functionality by encrypting an i-block message where 1 < i <=10. The evaluator shall choose a key, an IV and plaintext message of length i blocks and encrypt the message, using the mode to be

tested, with the chosen key and IV. The ciphertext shall be compared to the result of encrypting the same plaintext message with the same key and IV using a known good implementation.

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

AES-CBC Monte Carlo Tests

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

Input: PT, IV, Key

for i = 1 to 1000:

if i == 1:

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

PT = IV

else:

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

PT = CT[i-1]

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

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

AES-GCM Test

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

128 bit and 256 bit keys

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

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

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

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

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

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

AES-CTR Known Answer Tests

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

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

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

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

KAT-3 To test the encrypt functionality, the evaluator shall supply a set of key values for each selected keysize as described below and obtain the ciphertext values that result from AES encryption of an all zeros plaintext using the

given key values. A set of 128 128-bit keys, a set of 192 192-bit keys, and/or a set of 256 256-bit keys. Key_i in each set shall have the leftmost i bits be ones and the rightmost N-i bits be zeros, for i in [1, N].

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

AES-CTR Multi-Block Message Test

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

AES-CTR Monte-Carlo Test

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

Input: PT, Key

for i = 1 to 1000:

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

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

There is no need to test the decryption engine.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.8 Cryptographic Operation (AES Data Encryption/Decryption) (WLANAS10:FCS_COP.1/DataEncryption)

2.3.8.1 WLANAS10:FCS_COP.1.1/DataEncryption

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: There are no additional TSS evaluation activities for this component beyond what the NDcPP requires.

There are no additional TSS evaluation activities for this component beyond what the NDcPP requires.

Component Guidance Assurance Activities: There are no additional guidance evaluation activities for this component beyond what the NDcPP requires.

There are no additional guidance evaluation activities for this component beyond what the NDcPP requires.

Component Testing Assurance Activities: In addition to the tests required by the NDcPP, the evaluator will perform the following testing:

AES-CCM Tests

The evaluator will test the generation-encryption and decryption-verification functionality of AES-CCM for the following input parameter and tag lengths:

- 128 bit and 256 bit keys

- Two payload lengths. One payload length will be the shortest supported payload length, greater than or equal to zero bytes. The other payload length will be the longest supported payload length, less than or equal to 32 bytes (256 bits).

- Two or three associated data lengths. One associated data length will be 0, if supported. One associated data length will be the shortest supported payload length, greater than or equal to zero bytes. One associated data length will be the longest supported payload length, less than or equal to 32 bytes (256 bits). If the implementation supports an associated data length of 216 bytes, an associated data length of 216 bytes will be tested.

- Nonce lengths. All supported nonce lengths between 7 and 13 bytes, inclusive, will be tested.

- Tag lengths. All supported tag lengths of 4, 6, 8, 10, 12, 14, and 16 bytes will be tested.

Due to the restrictions that IEEE 802.11 specifies for this mode (nonce length of 13 and tag length of 8), it is acceptable to test a subset of the supported lengths as long as the selections fall into the ranges specified above. In this case, the evaluator will ensure that these are the only supported lengths. To test the generation-encryption functionality of AES-CCM, the evaluator will perform the following four tests:

Test 1: For each supported key and associated data length and any supported payload, nonce and tag length, the evaluator will supply one key value, one nonce value and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Test 2: For each supported key and payload length and any supported associated data, nonce and tag length, the evaluator will supply one key value, one nonce value and 10 pairs of associated data and payload values, and obtain the resulting ciphertext.

Test 3: For each supported key and nonce length and any supported associated data, payload, and tag length, the evaluator will supply one key value and 10 associated data, payload and nonce value 3-tuples, and obtain the resulting ciphertext.

Test 4: For each supported key and tag length and any supported associated data, payload, and nonce length, the evaluator will supply one key value, one nonce value and 10 pairs of associated data and payload values and, obtain the resulting ciphertext.

To determine correctness in each of the above tests, the evaluator will compare the ciphertext with the result of generation-encryption of the same inputs with a known good implementation.

To test the decryption-verification functionality of AES-CCM, for each combination of supported associated data length, payload length, nonce length, and tag length, the evaluator will supply a key value and 15 nonce, associated data and ciphertext 3-tuples, and obtain either a fail result or a pass result with the decrypted payload. The evaluator will supply 10 tuples that should fail and 5 that should pass per set of 15.

Additionally, the evaluator will use tests from the IEEE 802.11-02/362r6 document 'Proposed Test vectors for IEEE 802.11 TGi', dated September 10, 2002, Section 2.1 AES-CCMP Encapsulation Example and Section 2.2 Additional AES-CCMP Test Vectors to verify further the IEEE 802.11-2020 implementation of AES-CCMP.

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

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

2.3.9.1 NDcPP22e:FCS_COP.1.1/Hash

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the **ST** states the TOE supports cryptographic hashing services using SHA-1, SHA-256, SHA-384, and SHA-512 with digest sizes 160, 256, 384, and 512. Section 6.3 further describes how these hash functions are used in support of trusted channels using IPsec, TLS, RadSec and SSH.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section "Using FIPS-related CLI commands" of **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode ensures that the hash sizes are restricted to those supported by the TOE.

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

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

Short Messages Test - Bit-oriented Mode

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

Short Messages Test - Byte-oriented Mode

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

Selected Long Messages Test - Bit-oriented Mode

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

Selected Long Messages Test - Byte-oriented Mode

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

Pseudorandomly Generated Messages Test

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

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

2.3.10.1 NDcPP22e:FCS_COP.1.1/KeyedHash

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the **ST** states the TOE supports keyed-hash message authentication using HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512 using SHA-1/256/384/512 with 160/256/384/512 bit keys to produce a 160/256/384/512 output MAC. Section 6.3 further describes how these hash functions are used in support of trusted channels using IPsec and SSH.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section “Using FIPS-related CLI commands” of **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode ensures that the hash sizes are restricted to those supported by the TOE.

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

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

2.3.11.1 NDcPP22e:FCS_COP.1.1/SigGen

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the ST states The TOE supports the use of RSA with 2048 bit key sizes and ECDSA signatures with curves P-256, P-384, and P-521 for cryptographic signatures. Digital signatures are used in TLS and SSH communications and on product updates.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section “Using FIPS-related CLI commands” of **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode ensures that the hash sizes are restricted to those supported by the TOE.

Component Testing Assurance Activities: ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

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

ECDSA FIPS 186-4 Signature Verification Test

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

RSA Signature Algorithm Tests

Signature Generation Test

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

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

Signature Verification Test

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

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

The TOE has been CAVP tested. Refer to the CAVP certificates identified in Section 1.1.2.

2.3.12 HTTPS Protocol (NDcPP22e:FCS_HTTPS_EXT.1)

2.3.12.1 NDcPP22e:FCS_HTTPS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.2 NDcPP22e:FCS_HTTPS_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.12.3 NDcPP22e:FCS_HTTPS_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.3 of the **ST** states the TOE provides HTTPS (via TLS1.2), as specified in RFC 2818, to provide a secure interface for remote administrative functions.

Component Guidance Assurance 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.

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

Section "Using FIPS-related CLI commands" of **Admin Guide** describes the steps to enable FIPS mode. Enabling FIPS mode ensures that the hash sizes are restricted to those supported by the TOE.

Section "Administrating the Controller Remotely" of **Admin Guide** describes that the controller (SmartZone) can be accessed remotely using SSH or Web UI.

Component Testing Assurance 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.

Administration via HTTPS was successfully demonstrated in NDcPP22E:FTP_TRP.1/Admin and the TLS protocol was tested in NDcPP22E:FCS_TLSS_EXT.1. The TOE does not support X509 client authentication.

2.3.13 IPsec Protocol - per TD0633 (NDcPP22e:FCS_IPSEC_EXT.1)

2.3.13.1 NDcPP22e:FCS_IPSEC_EXT.1.1

TSS Assurance Activities: The evaluator shall examine the TSS and determine that it describes what takes place when a packet is processed by the TOE, e.g., the algorithm used to process the packet. The TSS describes how the SPD is implemented and the rules for processing both inbound and outbound packets in terms of the IPsec policy. The TSS describes the rules that are available and the resulting actions available after matching a rule. The TSS describes how those rules and actions form the SPD in terms of the BYPASS (e.g., no encryption), DISCARD (e.g., drop the packet), and PROTECT (e.g., encrypt the packet) actions defined in

RFC 4301.

As noted in section 4.4.1 of RFC 4301, the processing of entries in the SPD is non-trivial and the evaluator shall determine that the description in the TSS is sufficient to determine which rules will be applied given the rule structure implemented by the TOE. For example, if the TOE allows specification of ranges, conditional rules, etc., the evaluator shall determine that the description of rule processing (for both inbound and outbound packets) is sufficient to determine the action that will be applied, especially in the case where two different rules may apply. This description shall cover both the initial packets (that is, no SA is established on the interface or for that particular packet) as well as packets that are part of an established SA.

Section 6.3 of the **ST** states that the TOE supports IPsec communication between the TOE components – in the hardware deployment between the Controller (SZ) and AP and in the virtual deployment between the vSZ-D and the AP. It also supports IPsec for communication to an external syslog server via the Controller (SZ/vSZ). The TOE uses the encapsulating security payload (ESP) to protect the payloads of the traffic. Only IKEv2 is supported for both IPsec implementations.

For IPsec between AP and SZ/vSZ-D, the Security policy is based on the WLAN configuration - only specific WLAN traffic that the administrator chooses is protected via IPsec tunnel between AP and SZ/vSZ-D. All other traffic including WLANs that are not configured for IPsec is bypassed and transmitted locally from AP. All traffic that does not match the configured network is discarded.

For IPsec between Controller and external component, once the IPsec configuration is defined in Controller, it initiates the tunnel to the external component defined in the controller. All inbound and outbound traffic to/from Audit server is protected by IPsec in tunnel mode. All other traffic on the network is bypassed. All traffic that does not match the configured network is discarded. When a packet destined for the Audit server is processed by the TOE and it determines it requires IPsec, it uses active SA settings or creates new SAs for initial connections with the IPsec peer.

Guidance Assurance Activities: The evaluator shall examine the guidance documentation to verify it instructs the Administrator how to construct entries into the SPD that specify a rule for processing a packet. The description includes all three cases â€“ a rule that ensures packets are encrypted/decrypted, dropped, and flow through the TOE without being encrypted. The evaluator shall determine that the description in the guidance documentation is consistent with the description in the TSS, and that the level of detail in the guidance documentation is sufficient to allow the administrator to set up the SPD in an unambiguous fashion. This includes a discussion of how ordering of rules impacts the processing of an IP packet.

As indicated in Section 6.3 of the **ST**, the IPsec Security policy for WLAN traffic between TOE components is based on the WLAN configuration while the IPsec Security policy for traffic between the TOE and an external syslog server is defined on the Controller. All defined traffic is protected via the IPsec tunnel while all other traffic on the network is bypassed. All traffic that does not match the configured network is discarded.

Section "System IPsec" in the **Admin Guide** indicates that the System IPsec is the IPsec tunnel between the SZ controller and an external syslog server. Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in the **Admin Guide** describes how to enable IPsec from the controller to external systems such as a syslog server. Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in the **Admin Guide** indicates that the Ruckus GRE and IPsec is the IPsec tunnel between the AP and the SZ or vSZ-D and describes how to enable IPsec between the AP and SZ or vSZ-D.

Testing Assurance Activities: The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

a) Test 1: The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The selectors used in the construction of the rule shall be different such that the evaluator can generate a packet and send packets to the gateway with the appropriate fields (fields that are used by the rule - e.g., the IP addresses, TCP/UDP ports) in the packet header. The evaluator performs both positive and negative test cases for each type of rule (e.g. a packet that matches the rule and another that does not match the rule). The evaluator observes via the audit trail, and packet captures that the TOE exhibited the expected behaviour: appropriate packets were dropped, allowed to flow without modification, encrypted by the IPsec implementation.

b) Test 2: The evaluator shall devise several tests that cover a variety of scenarios for packet processing. As with Test 1, the evaluator ensures both positive and negative test cases are constructed. These scenarios must exercise the range of possibilities for SPD entries and processing modes as outlined in the TSS and guidance documentation. Potential areas to cover include rules with overlapping ranges and conflicting entries, inbound and outbound packets, and packets that establish SAs as well as packets that belong to established SAs. The evaluator shall verify, via the audit trail and packet captures, for each scenario that the expected behavior is exhibited, and is consistent with both the TSS and the guidance documentation.

Test 1 and 2 – The TOE supports a basic SPD implementation based on the configured address of the external IPsec peer and the distributed TOE components configured for ITT communication for the IPsec data tunnel. The TOE demonstrated a Protect rule for communication to the external IPsec peer and for WLAN data passed from the AP

to the SZ controller/vSZ-D. All other communication on the same subnet is sent outside the IPsec tunnels, demonstrating a BYPASS rule. All packets that are not on the configured subnet of the TOE are dropped, thus demonstrating a DISCARD rule. Due to this SPD implementation, each rule is a negative case of the other rules. FIA_X509_EXT.1.1/Rev Test 3 demonstrates the SPD rules clearly. In this test the packets to the configured external IPsec peer are encrypted, thus successfully implementing the PROTECT rule, while the OCSP packets to a different address, but same subnet, are sent in the clear, thus demonstrating the BYPASS rule. FCS_IPSEC_EXT.1.2, test 1 demonstrates the DISCARD rule. In this test, the evaluator attempted to ping the TOE from an unknown subnet and confirmed that the TOE did not respond. The TOE does not support a more complex implementation of SPD rules; thus the full implementation was tested.

2.3.13.2 NDcPP22e:FCS_IPSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

The evaluator uses the guidance documentation to configure the TOE to carry out the following tests:

The evaluator shall configure the SPD such that there is a rule for dropping a packet, encrypting a packet, and allowing a packet to flow in plaintext. The evaluator may use the SPD that was created for verification of FCS_IPSEC_EXT.1.1. The evaluator shall construct a network packet that matches the rule to allow the packet to flow in plaintext and send that packet. The evaluator should observe that the network packet is passed to the proper destination interface with no modification. The evaluator shall then modify a field in the packet header; such that it no longer matches the evaluator-created entries (there may be a 'TOE create'' final entry that discards packets that do not match any previous entries). The evaluator sends the packet and observes that the packet was dropped.

As stated in FCS_IPSEC_EXT.1.1 above, all packets that do match the configured subnet on the TOE are dropped. The evaluator attempted to ping the TOE from an unknown subnet and viewed that the TOE did not respond.

2.3.13.3 NDcPP22e:FCS_IPSEC_EXT.1.3

TSS Assurance Activities: The evaluator checks the TSS to ensure it states that the VPN can be established to operate in transport mode and/or tunnel mode (as identified in FCS_IPSEC_EXT.1.3).

Section 6.3 of the **ST** indicates that the IPsec communication between the distributed TOE components operates in transport mode only, while the IPsec communication between the TOE and the external audit server operates in tunnel mode only.

Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation contains instructions on how to configure the connection in each mode selected.

The **Admin Guide** indicates that there is no configuration required for configuring the connection in tunnel mode or transport mode.

Section "Configuring System IPsec using Preshared Key" in the **Admin Guide** states that System IPsec supports tunnel mode only.

Section "Configuring System IPsec using Certificates" in the **Admin Guide** states that System IPsec supports tunnel mode only.

Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in the **Admin Guide** states that RuckusGRE over IPsec is supported in transport mode only.

Testing Assurance Activities: The evaluator shall perform the following test(s) based on the selections chosen:

Test 1: If tunnel mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in tunnel mode and also configures a VPN peer to operate in tunnel mode. The evaluator configures the TOE and the VPN peer to use any of the allowable cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator shall then initiate a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the tunnel mode.

Test 2: If transport mode is selected, the evaluator uses the guidance documentation to configure the TOE to operate in transport mode and also configures a VPN peer to operate in transport mode. The evaluator configures the TOE and the VPN peer to use any of the allowed cryptographic algorithms, authentication methods, etc. to ensure an allowable SA can be negotiated. The evaluator then initiates a connection from the TOE to connect to the VPN peer. The evaluator observes (for example, in the audit trail and the captured packets) that a successful connection was established using the transport mode.

Test 1 - The evaluator alternately configured a test peer to require only tunnel mode or only transport mode. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was only successful if the configured mode is supported by the TOE.

Test 2 - The connection between the SZ controller and the AP was captured via wireshark. The IPsec data tunnel for distributed TOE communication uses transport mode.

2.3.13.4 NDcPP22e:FCS_IPSEC_EXT.1.4

TSS Assurance Activities: The evaluator shall examine the TSS to verify that the algorithms are implemented. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in

FCS_COP.1(4)/KeyedHash Cryptographic Operations (for keyed-hash message authentication) and if the SHA-based HMAC function truncated output is utilized it must also be described.

Section 6.3 of the **ST** states that for IPsec between AP and SZ/vSZ-D, the TOE uses AES-128 with SHA1 and MODP 2048, AES-256 with SHA384 and ECP384. For IPsec between the Controller (SZ/vSZ) and external component, the TOE uses AES-128, AES-192, AES-256 with HMAC-SHA1, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512 and ECP384.

Guidance Assurance Activities: The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in **Admin Guide** describes how to enable IPsec from the controller to external systems. The supported integrity algorithms are SHA1, SHA256, SHA384, and SHA512.

Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode. The IKE and ESP Proposals supported are AES128-SHA1-MODP2048 and AES256-SHA384-ECP384.

Testing Assurance Activities: The evaluator shall configure the TOE as indicated in the guidance documentation configuring the TOE to use each of the supported algorithms, attempt to establish a connection using ESP, and verify that the attempt succeeds.

The evaluator configured IPsec using a preshared key on both the TOE and a test peer. The evaluator then alternately configured the test peer to accept each of the supported algorithms identified in the **ST**. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and verified that each attempt succeeded.

2.3.13.5 NDcPP22e:FCS_IPSEC_EXT.1.5

TSS Assurance Activities: The evaluator shall examine the TSS to verify that IKEv1 and/or IKEv2 are implemented.

For IKEv1 implementations, the evaluator shall examine the TSS to ensure that, in the description of the IPsec protocol, it states that aggressive mode is not used for IKEv1 Phase 1 exchanges, and that only main mode is used. It may be that this is a configurable option.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – in the hardware deployment between the Controller (SZ) and AP and in the virtual deployment between the vSZ-D and the AP. It also supports IPsec for communication to an external syslog server via the Controller (SZ/vSZ). The TOE uses the encapsulating security payload (ESP) to protect the payloads of the traffic. Only IKEv2 is supported for both IPsec implementations.

Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure it instructs the administrator how to configure the TOE to use IKEv1 and/or IKEv2 (as selected), and how to configure the TOE to perform NAT traversal for the following test (if selected).

If the IKEv1 Phase 1 mode requires configuration of the TOE prior to its operation, the evaluator shall check the guidance documentation to ensure that instructions for this configuration are contained within that guidance.

Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in **Admin Guide** describes how to enable IPsec from the controller to external systems. All IPsec tunnels are configured by default to perform NAT traversal.

Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode.

Each section makes it clear that only IKEv2 is supported by the TOE.

Testing Assurance Activities: Tests are performed in conjunction with the other IPsec evaluation activities.

a) Test 1: If IKEv1 is selected, the evaluator shall configure the TOE as indicated in the guidance documentation and attempt to establish a connection using an IKEv1 Phase 1 connection in aggressive mode. This attempt should fail. The evaluator should then show that main mode exchanges are supported.

b) Test 2: If NAT traversal is selected within the IKEv2 selection, the evaluator shall configure the TOE so that it will perform NAT traversal processing as described in the TSS and RFC 5996, section 2.23. The evaluator shall initiate an IPsec connection and determine that the NAT is successfully traversed.

Test 1 – Not applicable. The TOE does not support IKEv1 for IPsec.

Test 2 - The evaluator configured the connection between the TOE and test peer so that NAT was required - both devices were on separate class C networks with a NAT router bridging the two networks. The evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and verified that the connection was successful regardless of the NAT routing.

2.3.13.6 NDcPP22e:FCS_IPSEC_EXT.1.6

TSS Assurance Activities: The evaluator shall ensure the TSS identifies the algorithms used for encrypting the IKEv1 and/or IKEv2 payload, and that the algorithms chosen in the selection of the requirement are included in the TSS discussion.

Section 6.3 in the **ST** states that for IPsec between AP and SZ/vSZ-D, the TOE uses AES-128 with SHA1 and MODP 2048, AES-256 with SHA384 and ECP384. For IPsec between Controller and external component, the TOE uses AES-128, AES-192, AES-256 with HMAC-SHA1, HMAC-SHA256, HMAC-SHA384, HMAC-SHA512 and ECP384.

Guidance Assurance Activities: The evaluator ensures that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Sections “Configuring System IPsec using Preshared Key” and “Configuring System IPsec using Certificates” in **Admin Guide** describes how to enable IPsec from the controller to external systems. The supported encryption algorithms are AES128, AES192, and AES256.

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode. The IKE and ESP Proposals supported are AES128-SHA1-MODP2048 and AES256-SHA384-ECP384.

Testing Assurance Activities: The evaluator shall configure the TOE to use the ciphersuite under test to encrypt the IKEv1 and/or IKEv2 payload and establish a connection with a peer device, which is configured to only accept the payload encrypted using the indicated ciphersuite. The evaluator will confirm the algorithm was that used in the negotiation.

The evaluator alternately configured a test peer to accept only the specified IKE cipher for each attempted connection. In each case, the evaluator attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was only successful if the configured IKE cipher was supported on the TOE.

2.3.13.7 NDcPP22e:FCS_IPSEC_EXT.1.7

TSS Assurance Activities: The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 1 SA lifetime and/or the IKEv2 SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. The default time value for Phase 1 SAs is 4 hours, but is configurable from 1 minute to 24 hours for both connection types.

Guidance Assurance Activities: The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 24 hours is exceeded (e.g. configure a time value of 23h 45min to ensure the actual rekey is performed no later than 24h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 1 SA value of 24 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 24 hours. It is not permitted to configure a value of 24 hours if that leads to an actual rekey after more than 24hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Sections “Configuring IKE and ESP Rekeying Separately” in the **Admin Guide** describes how to configure IKE and ESP Rekeying for IPsec communication between the TOE and an external syslog server. It indicates that the rekey time can be set in hours or minutes.

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in **Admin Guide** describes how to enable IKE and ESP rekeying for IPsec communication between the AP and the SZ controller or vSZ-D.

Testing Assurance Activities: When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC 'A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered.'

Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1: If 'number of bytes' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish a SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 1 negotiation.

b) Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 24 hours for the Phase 1 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 1 SA lifetime that exceeds the Phase 1 SA lifetime on the TOE.

Test 1 – Not applicable. The TOE does not support a number of bytes lifetime.

Test 2 – The evaluator configured the TOE for a Phase 1 lifetime of 24 hours and a Phase 2 lifetime of 8 hours on the SZ144 and SZ300 while the peer was set with lifetimes of 25 and 9 hours respectively to ensure the TOE lifetime hit first. The evaluator established an IPsec connection to the TOE from an IPsec peer while capturing the ISAKMP packets with wireshark and verified the attempt was successful. The evaluator waited over 24 to ensure both phase lifetimes were hit. The packet capture showed the CREATE_CHILD_SA packets that occur for reestablishment of both phases before the lifetime was hit.

Since the time is configurable and the implementation the same between devices, the vSZ-H was set with lower values for ease and lowering the impact of testing time for rerunning this test. For the vSZ-H lifetime values of 2 hours and 45 minutes were used on the TOE with the testlab again configured with longer lifetime values.

2.3.13.8 NDcPP22e:FCS_IPSEC_EXT.1.8

TSS Assurance Activities: The evaluator shall ensure the TSS identifies the lifetime configuration method used for limiting the IKEv1 Phase 2 SA lifetime and/or the IKEv2 Child SA lifetime. The evaluator shall verify that the selection made here corresponds to the selection in FCS_IPSEC_EXT.1.5.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. The default time value for Phase 2 SAs is 1 hour, but it is configurable 1 minute to 8 hours for the connection to the AP/vSZ-D. The default time value for Phase 2 SAs is 4 hours, but it is configurable 1 minute to 8 hours for the connection to external components.

Guidance Assurance Activities: The evaluator shall verify that the values for SA lifetimes can be configured and that the instructions for doing so are located in the guidance documentation. If time-based limits are supported, configuring the limit may lead to a rekey no later than the specified limit. For some implementations, it may be necessary, though, to configure the TOE with a lower time value to ensure a rekey is performed before the maximum SA lifetime of 8 hours is exceeded (e.g. configure a time value of 7h 45min to ensure the actual rekey is performed no later than 8h). The evaluator shall verify that the guidance documentation allows the Administrator to configure the Phase 2 SA value of 8 hours or provides sufficient instruction about the time value to configure to ensure the rekey is performed no later than the maximum SA lifetime of 8 hours. It is not permitted to configure a value of 8 hours if that leads to an actual rekey after more than 8 hours. Currently there are no values mandated for the number of bytes, the evaluator just ensures that this can be configured if selected in the requirement.

Sections “Configuring IKE and ESP Rekeying Separately” in the **Admin Guide** describes how to configure IKE and ESP Rekeying for IPsec communication between the TOE and an external syslog server. It indicates that the rekey time can be set in hours or minutes.

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in **Admin Guide** describes how to enable IKE and ESP rekeying for IPsec communication between the AP and the SZ controller or vSZ-D.

Testing Assurance Activities: When testing this functionality, the evaluator needs to ensure that both sides are configured appropriately. From the RFC 'A difference between IKEv1 and IKEv2 is that in IKEv1 SA lifetimes were negotiated. In IKEv2, each end of the SA is responsible for enforcing its own lifetime policy on the SA and rekeying the SA when necessary. If the two ends have different lifetime policies, the end with the shorter lifetime will end up always being the one to request the rekeying. If the two ends have the same lifetime policies, it is possible that both will initiate a rekeying at the same time (which will result in redundant SAs). To reduce the probability of this happening, the timing of rekeying requests SHOULD be jittered.'

Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

Test 1: If 'number of bytes' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime in terms of the number of bytes allowed following the guidance documentation. The evaluator shall configure a test peer with a byte lifetime that exceeds the lifetime of the TOE. The evaluator shall establish a SA between the TOE and the test peer, and determine that once the allowed number of bytes through this SA is exceeded, a new SA is negotiated. The evaluator shall verify that the TOE initiates a Phase 2 negotiation.

Test 2: If 'length of time' is selected as the SA lifetime measure, the evaluator shall configure a maximum lifetime no later than 8 hours for the Phase 2 SA following the guidance documentation. The evaluator shall configure a test peer with a Phase 2 SA lifetime that exceeds the Phase 2 SA lifetime on the TOE.

Test 1 – Not applicable. The TOE does not support a number of bytes lifetime.

Test 2 - This test was captured alongside the phase 2 lifetime testing in FCS_IPSEC_EXT.1.8, test 2.

2.3.13.9 NDcPP22e:FCS_IPSEC_EXT.1.9

TSS Assurance Activities: The evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating 'x'. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of 'x' meets the stipulations in the requirement.

Section 6.3 of the **ST** states that for IPsec between the TOE components, the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in g^x mod p) using the DRBG specified in FCS_RBG_EXT.1 and having possible lengths of 224 or 384 bits (for DH Groups 14 and 20, respectively). The nonce generated is 32 bytes long (or 256 bits), thus being over 128 bits in size and larger than half the output size of SHA384, and is generated with the DRBG specified in FCS_RBG_EXT.1. For IPsec between the Controller (SZ/vSZ) and an external syslog server, the secret 'x' generated is 48 bytes long (or 384 bits) and is generated with the DRBG specified in FCS_RBG_EXT.1. The nonce generated is 32 bytes long (or 256 bits), thus being over 128 bits in size and half the output size of SHA512, and is generated with the DRBG specified in FCS_RBG_EXT.1.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.13.10 NDcPP22e:FCS_IPSEC_EXT.1.10

TSS Assurance Activities: If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Section 6.3 of the **ST** states that for IPsec between the TOE components, the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in g^x mod p) using the DRBG specified in FCS_RBG_EXT.1 and having possible lengths of 224 or 384 bits (for DH Groups 14 and 20, respectively). The nonce generated is 32 bytes long (or 256 bits), thus being over 128 bits in size and larger than half the output size of SHA384, and is generated with the

DRBG specified in FCS_RBG_EXT.1. For IPsec between the Controller (SZ/vSZ) and an external syslog server, the secret 'x' generated is 48 bytes long (or 384 bits) and is generated with the DRBG specified in FCS_RBG_EXT.1. The nonce generated is 32 bytes long (or 256 bits), thus being over 128 bits in size and half the output size of SHA512, and is generated with the DRBG specified in FCS_RBG_EXT.1.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: Each of the following tests shall be performed for each version of IKE selected in the FCS_IPSEC_EXT.1.5 protocol selection:

a) Test 1: If the first selection is chosen, the evaluator shall check to ensure that, for each DH group supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

b) Test 2: If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Test 1 and Test 2 – Refer to the TSS assurance activities above.

2.3.13.11 NDcPP22e:FCS_IPSEC_EXT.1.11

TSS Assurance Activities: The evaluator shall check to ensure that the DH groups specified in the requirement are listed as being supported in the TSS. If there is more than one DH group supported, the evaluator checks to ensure the TSS describes how a particular DH group is specified/negotiated with a peer.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. The IPsec tunnel between the AP and SZ/vSZ-D implements DH Groups 14 (2048-bit MODP) and 20 (384-bit Random ECP). The Administrator selects the DH group. The IPsec tunnel between the Controller and external component implements DH group 20 (384-bit Random ECP).

Guidance Assurance Activities: The evaluator ensures that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Sections "Configuring System IPsec using Preshared Key" and "Configuring System IPsec using Certificates" in the **Admin Guide** describes how to enable IPsec from the controller to external systems including how to configure the IKE and ESP encryption and integrity algorithms. The supported encryption algorithms are AES128, AES192, and AES256. The supported integrity algorithms are SHA1, SHA256, SHA384, and SHA512. The IKE encryption proposals should be greater than or equal to the ESP encryption proposal. By default DH group will be DH-20 [ECP-384] which cannot be changed.

Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode and how to configure the IKE and ESP encryption and integrity algorithms. The IKE and ESP Proposals supported are AES128-SHA1-MODP2048 and AES256-SHA384-ECP384 which correspond to DH group 14 and 20.

Testing Assurance Activities: For each supported DH group, the evaluator shall test to ensure that all supported IKE protocols can be successfully completed using that particular DH group.

The evaluator configured a test peer to accept only the supported DH groups. The evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was successful in each case.

2.3.13.12 NDcPP22e:FCS_IPSEC_EXT.1.12

TSS Assurance Activities: The evaluator shall check that the TSS describes the potential strengths (in terms of the number of bits in the symmetric key) of the algorithms that are allowed for the IKE and ESP exchanges. The TSS shall also describe the checks that are done when negotiating IKEv1 Phase 2 and/or IKEv2 CHILD_SA suites to ensure that the strength (in terms of the number of bits of key in the symmetric algorithm) of the negotiated algorithm is less than or equal to that of the IKE SA this is protecting the negotiation.

Section 6.3 in the **ST** states the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. As part of this negotiation for both of these connections, the TOE verifies that the negotiated phase 2 symmetric algorithm key strength is at most as large as the negotiated phase 1 key strength as configured on the TOE and peer via an explicit check.

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator simply follows the guidance to configure the TOE to perform the following tests.

a) Test 1: This test shall be performed for each version of IKE supported. The evaluator shall successfully negotiate an IPsec connection using each of the supported algorithms and hash functions identified in the requirements.

b) Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish a SA for ESP that selects an encryption algorithm with more strength than that being used for the IKE SA (i.e., symmetric algorithm with a key size larger than that being used for the IKE SA). Such attempts should fail.

c) Test 3: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an IKE SA using an algorithm that is not one of the supported algorithms and hash functions identified in the requirements. Such an attempt should fail.

d) Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish a SA for ESP (assumes the proper parameters where used to establish the IKE SA) that selects an encryption algorithm that is not identified in FCS_IPSEC_EXT.1.4. Such an attempt should fail.

Test 1 - The evaluator alternately configured a test peer to accept the supported IKE hash algorithms identified in the **ST**. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was only successful if the configured IKE hash was supported on the TOE. Note that the range of IKE algorithms was tested according to FCS_IPSEC_EXT.1.6.

Test 2 – The TOE meets this test by design. For both configuration of the IPsec external connection and the WLAN data tunnel for ITT communication, the TOE prevents the admin from selecting an ESP algorithm greater than the IKE algorithm.

Test 3- The evaluator alternately configured a test peer as follows: 1) Control test with all supported ciphers and hashes, 2) restrict the IKE cipher to 128-bit blowfish, 3) restrict the IKE hash to MD5, and 4) restrict the ESP cipher to 128-bit blowfish. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and confirmed that the connection was only successful for the control case.

Test 4 - This test was performed in FCS_IPSEC_EXT.1.12, test 3.

2.3.13.13 NDcPP22e:FCS_IPSEC_EXT.1.13

TSS Assurance Activities: The evaluator ensures that the TSS identifies RSA and/or ECDSA as being used to perform peer authentication. The description must be consistent with the algorithms as specified in FCS_COP.1(2)/SigGen Cryptographic Operations (for cryptographic signature).

If pre-shared keys are chosen in the selection, the evaluator shall check to ensure that the TSS describes how pre-shared keys are established and used in authentication of IPsec connections. The description in the TSS shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. The IPsec tunnel between the TOE components supports RSA X509 certificates, while the IPsec tunnel between the TOE and the external syslog server supports either RSA or ECDSA. This is consistent with the claims made in FCS_COP.1(2)/SigGen.

For IPsec between Controller and external component, the TOE supports both bit based and text based keys for Pre-shared keys. A TOE administrator specifies the PSK in the controller configuration. The PSK includes a combination of upper and lower-case letters, numbers, and supported special characters and must be 44-128 for Hex characters and 8-64 for ASCII.

Guidance Assurance Activities: The evaluator ensures the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

The evaluator shall check that the guidance documentation describes how pre-shared keys are to be generated and established. The description in the guidance documentation shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

The evaluator will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA and ensure a valid certificate for that CA is loaded into the TOE and marked 'trusted'.

Section “Configuring System IPsec using Preshared Key” in **Admin Guide** describes how to enable IPsec from the controller to external systems using a pre-shared key. The Preshared key is text-based with valid length range for ASCII from 8 through 64 and bit-based text is from 44 through 128.

Section “Configuring System IPsec using Certificates” in **Admin Guide** describes how to enable IPsec from the controller to external systems using certificates. This includes configuring the certificate to use the remote ID, and the OCSP information. Both RSA and ECDSA private keys are supported.

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode. This mode uses RSA authentication only.

Testing Assurance Activities: For efficiency sake, the testing is combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

The testing of successful IPsec authentication using pre-shared key was demonstrated throughout the testing of FCS_IPSEC_EXT.1, including, for example, FCS_IPSEC_EXT.1.4.

2.3.13.14 NDcPP22e:FCS_IPSEC_EXT.1.14

TSS Assurance Activities: The evaluator shall ensure that the TSS describes how the TOE compares the peer's presented identifier to the reference identifier. This description shall include which field(s) of the certificate are used as the presented identifier (DN, Common Name, or SAN). If the TOE simultaneously supports the same identifier type in the CN and SAN, the TSS shall describe how the TOE prioritizes the comparisons (e.g. the result of comparison if CN matches but SAN does not). If the location (e.g. CN or SAN) of non-DN identifier types must explicitly be configured as part of the reference identifier, the TSS shall state this. If the ST author assigned an additional identifier type, the TSS description shall also include a description of that type and the method by which that type is compared to the peer's presented certificate, including what field(s) are compared and which fields take precedence in the comparison.

Section 6.3 in the **ST** states that the TOE supports IPsec communication between the TOE components – AP and SZ/vSZ-D and also to external components. The tunnel between AP and SZ/vSZ-D implements X509 certificates using RSA. The TOE validates the following identifier:

- Full distinguish name (DN).

The IPsec tunnel between the TOE and the external syslog server implements X509 certificates, either ECDSA or RSA, the TOE validates one of the following identifiers:

- Full distinguish name (DN).

Guidance Assurance Activities: The evaluator shall ensure that the operational guidance describes all supported identifiers, explicitly states whether the TOE supports the SAN extension or not and includes detailed instructions on how to configure the reference identifier(s) used to check the identity of peer(s). If the identifier scheme implemented by the TOE does not guarantee unique identifiers, 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.

Section "Configuring System IPsec using Certificates" in **Admin Guide** describes how to enable IPsec from the controller to external systems using certificates. This includes configuring the certificate to use the remote ID, and the OCSP information. Both RSA and ECDSA private keys are supported.

Section "Configuring Ruckus GRE and IPsec in WLAN-Concept" in **Admin Guide** describes how to enable IPsec for the AP tunnel data using transport mode. It states that peer reference identifiers are not configurable. The SZ will autogenerate reference identifiers to the AP and the vSZ-D.

Testing Assurance Activities: In the context of the tests below, a valid certificate is a certificate that passes FIA_X509_EXT.1 validation checks but does not necessarily contain an authorized subject.

The evaluator shall perform the following tests:

Test 1: (conditional) For each CN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes CN checking over SAN (through explicit configuration of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the SAN so it contains an incorrect identifier of the correct type (e.g. the reference identifier on the TOE is example.com, the CN=example.com, and the SAN:FQDN=otherdomain.com) and verify that IKE authentication succeeds.

Test 2: (conditional) For each SAN/identifier type combination selected, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the field in the peer's presented certificate and shall verify that the IKE authentication succeeds. If the TOE prioritizes SAN checking over CN (through explicit specification of the field when specifying the reference identifier or prioritization rules), the evaluator shall also configure the CN so it contains an incorrect identifier formatted to be the same type (e.g. the reference identifier on the TOE is DNS-ID; identify certificate has an identifier in SAN with correct DNS-ID, CN with incorrect DNS-ID (and not a different type of identifier)) and verify that IKE authentication succeeds.

Test 3: (conditional) For each CN/identifier type combination selected, the evaluator shall:

a) Create a valid certificate with the CN so it contains the valid identifier followed by ''. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or

prioritization rules) for the same identifier type, the evaluator shall configure the SAN so it matches the reference identifier.

b) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the CN without the '' and verify that IKE authentication fails.

Test 4: (conditional) For each SAN/identifier type combination selected, the evaluator shall:

a) Create a valid certificate with an incorrect identifier in the SAN. The evaluator shall configure a string representation of the correct identifier in the DN. If the TOE prioritizes CN checking over SAN (through explicit specification of the field when specifying the reference identifier or prioritization rules) for the same identifier type, the addition/modification shall be to any non-CN field of the DN. Otherwise, the addition/modification shall be to the CN.

b) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the correct identifier (expected in the SAN) and verify that IKE authentication fails.

Test 5: (conditional) If the TOE supports DN identifier types, the evaluator shall configure the peer's reference identifier on the TOE (per the administrative guidance) to match the subject DN in the peer's presented certificate and shall verify that the IKE authentication succeeds.

Test 6: (conditional) If the TOE supports DN identifier types, to demonstrate a bit-wise comparison of the DN, the evaluator shall create the following valid certificates and verify that the IKE authentication fails when each certificate is presented to the TOE:

a) Duplicate the CN field, so the otherwise authorized DN contains two identical CNs.

b) Append '' to a non-CN field of an otherwise authorized DN.

Test 1 – Not applicable. The TOE does not support CN identifiers for IPsec.

Test 2 – Not applicable. The TOE does not support SAN identifiers for IPsec.

Test 3 – Not applicable. The TOE does not support CN identifiers for IPsec.

Test 4 – Not applicable. The TOE does not support SAN identifiers for IPsec.

Test 5 – The evaluator configured the TOE for certificate authentication and configured a test peer to send a certificate with a valid DN. The evaluator then initiated a connection between the TOE and the test peer and confirmed that the connection was successful. This test was iterated for both RSA and ECDSA.

Test 6 - The evaluator configured the TOE for certificate authentication and configured a test peer to send a certificate with two identical CNs and then a certificate with a null character appended to a non-CN field. The evaluator then initiated a connection between the TOE and the test peer and confirmed in each case that the connection failed. This test was iterated for both RSA and ECDSA.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.14 NTP Protocol (NDcPP22e:FCS_NTP_EXT.1)

2.3.14.1 NDcPP22e:FCS_NTP_EXT.1.1

TSS Assurance Activities: The evaluator shall examine the TSS to ensure it identifies the version of NTP supported, how it is implemented and what approach the TOE uses to ensure the timestamp it receives from an NTP timeserver (or NTP peer) is from an authenticated source and the integrity of the time has been maintained. The TOE must support at least one of the methods or may use multiple methods, as specified in the SFR element 1.2. The evaluator shall ensure that each method selected in the ST is described in the TSS, including the version of NTP supported in element 1.1, the message digest algorithms used to verify the authenticity of the timestamp and/or the protocols used to ensure integrity of the timestamp.

Section 6.3 in the **ST** states the TOE supports protected communication to an NTP server using NTP v4. Authentication is performed using SHA1 as the message digest algorithm.

Guidance Assurance Activities: 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.

Section “Configuring System Time” in **Admin Guide** describes how the controller uses an external Network Time Protocol (NTP) server to synchronize the times across cluster nodes and managed access points. This includes configuring multiple NTP servers and configuring the SHA1 NTP server authentication.

Testing Assurance Activities: 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.

The version of NTP was verified in NDcPP22e:FCS_NTP_EXT.1.2.

2.3.14.2 NDcPP22e:FCS_NTP_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: 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 administrator how to configure the TOE to use the chosen option(s).

Section "Configuring System Time" in **Admin Guide** describes how the controller uses an external Network Time Protocol (NTP) server to synchronize the times across cluster nodes and managed access points. This includes configuring multiple NTP servers and configuring the SHA1 NTP server authentication.

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

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

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

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

The TOE does not support a trusted channel for NTP, only message digest algorithms. The evaluator configured the NTP server to use an invalid MD5 NTP key, and configured the TOE to use a SHA1 key. The evaluator verified that the TOE did not synchronize with the time source and the time did not change. Next the evaluator configured the NTP server to use the valid SHA1 NTP key and confirmed that the time was synchronized correctly and that the packet capture identified that NTP version 4 and the SHA1 message digest algorithms were used. The successful time change was demonstrated in FPT_STM_EXT.1.

2.3.14.3 NDcPP22e:FCS_NTP_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: 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.

Section “Configuring System Time” in the **Admin Guide** indicates that the TOE does not accept broadcast and multicast NTP packets that would result in the timestamp being updated, these packets are ignored by default.

Testing Assurance Activities: 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 configured the NTP server to support periodic time updates to broadcast and multicast addresses and then verified that the NTP server sent broadcast and multicast packets and that the TOE timestamp did not change to the NTP server time.

2.3.14.4 NDcPP22e:FCS_NTP_EXT.1.4

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

Test 2: (The intent of this test is to ensure that the TOE would only accept NTP updates from configured NTP Servers).

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

(TD0528 applied)

Test 1: The evaluator first started by using the TOE's WebUI to configure 3 different NTP servers on the TOE and use them for time synchronization. The evaluator then started a packet capture and confirmed that the TOE synchronizes its' time through one of the NTP servers as can be seen in the full packet capture below. The evaluator also used the TOE's audit log to confirm that the TOE synchronized its' time through the NTP servers.

Test 2: The evaluator configured the TOE with 3 unreachable NTP connections and observed the current time on the TOE and NTP server. The evaluator set the time on the NTP server ahead by 1 hour and observed that NTP servers were unreachable. The evaluator collected network traffic while monitoring the time on the TOE while an untrusted NTP server was configured to broadcast to the TOE. As can be seen in this packet capture, the TOE ignored the broadcast and attempted to update time using traditional authenticated updates with the unreachable NTP servers.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.15 RadSec (WLANAS10:FCS_RADSEC_EXT.1)

2.3.15.1 WLANAS10:FCS_RADSEC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.15.2 WLANAS10:FCS_RADSEC_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will verify that the TSS description includes the use of RADIUS over TLS, as described in RFC 6614.

If X.509v3 certificates is selected, the evaluator will ensure that the TSS description includes the use of clientside certificates for TLS mutual authentication.

Section 6.3 in the **ST** states that the TOE supports RADIUS over TLS as specified in RFC 6614. The TOE performs the function of a RADSEC client and conducts mutual authentication via X.509v3 certificates. The TOE supports only TLS 1.2 and rejects all other TLS and SSL versions. The TOE supports the ciphersuites defined in FCS_TLSC_EXT.1. The TOE will present the supported elliptic curve extensions with the following NIST curves: secp256r1, secp384r1, secp521r1.

The reference identifier is configured by the administrator of the TOE. The Controller initiates the RADSEC connection and verifies the presented identifier (FQDN) of the server in the certificate that server presents to it. If the certificate is not valid, the connection is dropped. IP address, wild card is not supported. OCSP is supported to verify the validity of the certificate.

Component Guidance Assurance Activities: The evaluator will verify that any configuration necessary to meet the requirement must be contained in the guidance.

Section "RadSec (RADIUS OVER TLS)" in the **Admin Guide** describes the process to configure communication to a remote AAA server using Radius over TLS.

TLS cipher suites are not user configurable. Following is the list of cipher suites supported by SZ (RadSec client):

- ECDHE-RSA-AES128-SHA256
- ECDHE-RSA-AES256-GCM-SHA384
- ECDHE-RSA-AES256-SHA384
- ECDHE-RSA-AES128-GCM-SHA256
- DHE-RSA-AES128-SHA256
- DHE-RSA-AES256-SHA256

If TLS is enabled:

- Secondary server configuration is disabled.
- Only then the user can configure OCSP Validation and CN/SAN Identity.
- OCSP Validation is disabled by default.
- CN/SAN becomes a mandatory field. The validation is performed with the configured identity and is used by most of the certificates.

Component Testing Assurance Activities: The evaluator will demonstrate the ability to successfully establish a RADIUS over TLS connection with a RADIUS server. This test will be performed with X.509v3 certificates if selected and performed with preshared keys if selected.

A successful RadSec connection was demonstrated in NDcPP22e:FTP_ITC.1, test 4.

2.3.16 Random Bit Generation (NDcPP22e:FCS_RBG_EXT.1)

2.3.16.1 NDcPP22e:FCS_RBG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.16.2 NDcPP22e:FCS_RBG_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.3 in the **ST** indicates that the TOE implements a random bit generator (RBG) in accordance with ISO/IEC 18031:2011. RBG is seeded with both hardware and software sources at least 256 bits of entropy to the DRBG.

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

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

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes, including RNG functionality, are allowed in the TOE. Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode. No additional configuration other than enabling FIPS mode is needed to limit the cryptographic algorithms and processes available to be CC/FIPS compliant.

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

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

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

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

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

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

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

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

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

The TOE has been CAVP tested. Refer to the section entitled, "CAVP Certificate Justification" earlier in this document.

2.3.17 SSH Client Protocol - per TD0636 (NDcPP22e:FCS_SSHC_EXT.1)

2.3.17.1 NDcPP22e:FCS_SSHC_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.17.2 NDcPP22e:FCS_SSHC_EXT.1.2

TSS Assurance Activities: The evaluator shall check to ensure that the TSS contains a list of the public key algorithms that are acceptable for use for user authentication and that this list is consistent with asymmetric key generation algorithms selected in FCS_CKM.1, hashing algorithms selected in FCS_COP.1/Hash, and signature generation algorithms selected in FCS_COP.1/SigGen. The evaluator shall confirm the TSS is unambiguous in declaring the TOE's ability to authenticate itself to a remote endpoint with a user-based public key.

If password-based authentication method has been selected in the FCS_SSHC_EXT.1.2, then the evaluator shall confirm it is also described in the TSS.

The TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH client implementation supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

The ssh-rsa public key algorithm is consistent with the supported claims for FCS_COP.1/Hash and FCS_COP.1/SigGen.

Guidance Assurance Activities: 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 initiated by the TOE.

Section “Joining vSZ-D to the vSZ Controller” in the **Admin Guide** describes the steps for joining of the vSZ-D to the vSZ Controller. The IP address of the vSZ controller is configured on the vSZ-D which then initiates the attempt to connect to the controller. The administrator then selects and approves the vSZ-D via the controller’s Web UI. Once the registration process is approved, an SSH connection is automatically established and all communication between the vSZ-D and the vSZ Controller is over SSH. The connection with the vSZ-D can similarly be disabled if

the administrator selects the vSZ-D and then clicks the 'Delete' button. If the connection between the vSZ-D and vSZ is broken then it resumes back automatically and no manual intervention is required.

Section "Joining AP to (v)SZ Controller" in the **Admin Guide** describes the steps for joining of the AP to the SZ Controller. The IP address of the SZ controller is configured on the AP which then initiates the attempt to connect to the controller. The administrator then selects and approves the AP via the controller's Web UI. Once the registration process is approved, an SSH connection is automatically established between the AP and the SZ Controller. The connection with the AP can similarly be disabled if the administrator selects the AP and then clicks the 'Delete' button. If the connection is broken it will be resumed/reattempted without any user intervention.

Section "Management Channel Between AP/vSZ-D and Controller" in the **Admin Guide** further describes the SSH communication between the AP and SZ controller and between the vSZ-D and the vSZ controller. The SSH algorithms, parameters and rekey limits are not configurable. The communication is only through public key auth (no password based authentication).

Testing Assurance Activities: Test objective: The purpose of these tests is to check the authentication of the client to the server using each claimed authentication method.

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

Test 2: [Conditional] If password-based authentication method has been selected in the FCS_SSHC_EXT.1.2, then following the guidance documentation the evaluator shall configure the TOE to perform password-based authentication with a remote SSH server to demonstrate that the TOE can successfully authenticate using a password as an authentication method.

Test 1: The evaluator attempted to connect from the TOE to an SSH server using a pubkey. The TOE claims support for ssh-rsa public keys only. This supported public key algorithm resulted in a successful connection.

Test 2: The evaluator attempted to connect to the TOE using a SSH client alternately using the correct and incorrect password. The evaluator found that only the correct password would yield a successful SSH session.

2.3.17.3 NDcPP22e:FCS_SSHC_EXT.1.3

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

Section 6.3 in the **ST** states that packets larger than 256K bytes are dropped.

Guidance Assurance Activities: None Defined

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

The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured to send a large packet of 262145 bytes, 1 greater than the max threshold after the connection succeeded. The evaluator viewed via Wireshark and SSH server logs that the AP terminated the connection after receiving the large packet

2.3.17.4 NDcPP22e:FCS_SSHC_EXT.1.4

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

The TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH client implementation supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFRs.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section “Management Channel between AP/vSZ-D and Controller” in **Admin Guide** indicates that the SSH parameters and algorithms are non-configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the DP or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB. Section “Administrating the Controller” in the **Admin Guide** identifies all of the algorithms and key exchange methods supported by the TOE’s SSHv2 implementation (both client and server) which are consistent with the **ST**.

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

The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured to support each of the claimed supported ciphers, aes128-ctr, aes256-ctr, and aes256-gcm@openssh.com. The evaluator viewed that the client only presented these claimed ciphers and that each connection was successful.

2.3.17.5 NDcPP22e:FCS_SSHC_EXT.1.5

TSS Assurance Activities: The evaluator shall confirm the TSS describes how a host-key public key (i.e., SSH server's public key) is associated with the server identity.

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

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH client implementation supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Host Key and Public Key authentication algorithms: ssh-rsa
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

x509v3-based public key authentication algorithms are not supported.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section "Management Channel between AP/vSZ-D and Controller" in **Admin Guide** indicates that the SSH parameters and algorithms are non-configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the DP or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB. Section "Administrating the Controller" in the **Admin Guide** identifies all of the algorithms and key exchange methods supported by the TOE's SSHv2 implementation (both client and server) which are consistent with the **ST**.

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

Test objective: The purpose of this positive test is to check the authentication of the server by the client (when establishing the transport layer connection), and not for checking generation of the authentication message from the client (in the User Authentication Protocol). The evaluator is therefore intended to establish sufficient separate SSH connections (with an appropriately configured server) to cause the TOE to demonstrate use of all public key algorithms claimed in FCS_SSHC_EXT.1.5 in the ST.

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

Test 1 - The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured with a supported ssh-rsa public key. The evaluator viewed that the connection was successful.

Test 2 - The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured with an unsupported ssh-dss public key. The evaluator viewed that the connection was unsuccessful.

2.3.17.6 NDcPP22e:FCS_SSHC_EXT.1.6

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH client implementation supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section “Management Channel between AP/vSZ-D and Controller” in **Admin Guide** indicates that the SSH parameters and algorithms are non-configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the DP or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB. Section “Administrating the Controller” in the **Admin Guide** identifies all of the algorithms and key exchange methods supported by the TOE’s SSHv2 implementation (both client and server) which are consistent with the **ST**.

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

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

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

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

Test 1 - The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured to support each of the claimed MAC algorithms hmac-sha1, hmac-sha2-256, and hmac-sha2-512 . The evaluator viewed that the connections were successful.

Test 2 - The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured with hmac-md5. The evaluator viewed that the connection was unsuccessful.

2.3.17.7 NDcPP22e:FCS_SSHC_EXT.1.7

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH client implementation supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section “Management Channel between AP/vSZ-D and Controller” in **Admin Guide** indicates that the SSH parameters and algorithms are non-configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the DP or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB. Section

"Administrating the Controller" in the **Admin Guide** identifies all of the algorithms and key exchange methods supported by the TOE's SSHv2 implementation (both client and server) which are consistent with the **ST**.

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

The evaluator configured a test server spoofing as the SZ controller and waited for the AP to initiate an SSH connection. The test server was configured to support each of the claimed key exchange methods diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384, and ecdh-sha2-nistp521. The evaluator viewed that the connections were successful.

2.3.17.8 NDcPP22e:FCS_SSHC_EXT.1.8

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

1. Both thresholds are checked by the TOE.

2. Rekeying is performed upon reaching the threshold that is hit first.

Section 6.3 in the **ST** states that in the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

When the AP or vSZ-D joins the controller, it has the ability to transmit a rekey request to the SSH server. The SSH session key will be re-negotiated when either one of the following two conditions is met:

1) The maximum amount of data transmitted over the SSH connection exceeds 1G bytes
2) The maximum amount of connection time exceeds 1 hour.

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

Section "Management Channel between AP/vSZ-D and Controller" in **Admin Guide** states that the SSH encryption algorithm, SSH integrity MAC algorithm, SSH client and server parameters, and rekey limitation are not user configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the vSZ-D or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB.

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

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

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

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

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

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

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

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

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

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

The evaluator used a special debug command to lower the thresholds from 1GB and 1 hour for testing. The command allowed the evaluator to first set a rekey limit of 100K. The evaluator initiated a log file transfer from the AP to the SZ controller and viewed via AP debug logs that the SSH rekey was initiated by the AP several times for the ITT SSH connection to the SZ controller.

2.3.17.9 NDcPP22e:FCS_SSHC_EXT.1.9

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

Test 1 - The evaluator used a special debug command on the AP to clear the known host keys on the AP. The evaluator forced the AP to initiate a new client connection to the SZ controller and viewed that the connection failed.

Test 2- The evaluator configured the test server that successfully spoofed as the SZ controller in previous tests with a new host key. The evaluator viewed that the connection from the AP was unsuccessful.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

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

2.3.18.1 NDcPP22e:FCS_SSHS_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.3.18.2 NDcPP22e:FCS_SSHS_EXT.1.2

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

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

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

Section 6.3 in the **ST** indicates that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication). SSH for remote administration to the SmartZone controller support both public key and password-based authentication.

The SSH Server implementation (for both remote administration and between TOE components) supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

The public key authentication algorithms are consistent with the claims for FCS_COP.1/SigGen.

X509v3 public key algorithms are not supported.

Guidance Assurance Activities: None Defined

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

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

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

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

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

(TD0631 applied)

Test 1: The evaluator generated an authentication key pair for both of claimed public-key authentication methods, ssh-rsa and ecdsa-sha2-nistp384. The evaluator then imported both of these public keys into the TOE. The evaluator successfully connected to the TOE using the configured public keys.

Test 2: The evaluator attempted to connect to the TOE using a SSH client using a pubkey not configured on the TOE. The evaluator found that an unconfigured pubkey would not be used to establish an SSH session and the TOE would revert back to password authentication.

Test 3: The evaluator attempted to connect to the TOE using a SSH client alternately using the correct and incorrect password. The evaluator found that only the correct password would yield a successful SSH session.

Test 4: This test was performed as part of Test 3 as described above.

2.3.18.3 NDcPP22e:FCS_SSHS_EXT.1.3

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

Section 6.3 in the **ST** states that packets larger than 256K bytes are dropped.

Guidance Assurance Activities: None Defined

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

The evaluator attempted to connect to the TOE using an SSH client and then sent a packet too large (262150 bytes). The evaluator observed that the session was disconnected by the TOE.

2.3.18.4 NDcPP22e:FCS_SSHS_EXT.1.4

TSS Assurance Activities: The evaluator shall check the description of the implementation of this protocol in the TSS to ensure that optional characteristics are specified, and the encryption algorithms supported are specified as

well. The evaluator shall check the TSS to ensure that the encryption algorithms specified are identical to those listed for this component.

Section 6.3 of the **ST** indicates that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH Server implementation (for both remote administration and between TOE components) supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFRs.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section “Administrating the Controller” in **Admin Guide** indicates that there is no configuration needed to access the Controller via an SSH session. It identifies all of the algorithms and key exchange methods supported by the TOE’s SSHv2 server implementation which are consistent with the **ST**.

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

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

2.3.18.5 NDcPP22e:FCS_SSHS_EXT.1.5

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH Server implementation (for both remote administration and between TOE components) supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Host Key authentication algorithms: ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384
- Public Key authentication algorithms: ssh-rsa, ecdsa-sha2-nistp384
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section "Administrating the Controller" in **Admin Guide** indicates that there is no configuration needed to access the Controller via an SSH session. It identifies all of the algorithms and key exchange methods supported by the TOE's SSHv2 server implementation which are consistent with the **ST**.

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

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

Has effectively been moved to FCS_SSHS_EXT.1.2.

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

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

(TD0631 applied)

Test 1: The evaluator attempted to connect to the TOE using a SSH client alternately using each of the authentication algorithms that can be claimed to determine which host key algorithms are supported with successful connections. The evaluator confirmed that the TOE supports the ssh-rsa, rsa-sha2-256, rsa-sha2-512, and ecdsa-sha2-nistp384 algorithms as claimed. The evaluator followed up with a disallowed authentication algorithm and confirmed that it was not accepted.

Test 2: This test was performed as part of Test 1.

2.3.18.6 NDcPP22e:FCS_SSHS_EXT.1.6

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH Server implementation (for both remote administration and between TOE components) supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section “Administrating the Controller” in **Admin Guide** indicates that there is no configuration needed to access the Controller via an SSH session. It identifies all of the algorithms and key exchange methods supported by the TOE’s SSHv2 server implementation which are consistent with the **ST**.

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

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

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

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

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

2.3.18.7 NDcPP22e:FCS_SSHS_EXT.1.7

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

Section 6.3 in the **ST** states that the TOE supports SSHv2 for both secure remote administration as well as for communications between TOE components. In the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

The SSH Server implementation (for both remote administration and between TOE components) supports the following:

- Encryption Algorithms: aes128-ctr, aes256-ctr and aes256-gcm@openssh.com
- Public Key authentication algorithms: ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384
- Data Integrity MAC algorithms: hmac-sha1, hmac-sha2-256, hmac-sha2-512 and implicit
- Key Exchange Methods: diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384 and ecdh-sha2-nistp521

This matches the claims in the SFR.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which automatically includes limiting the SSH algorithms to those supported by the TOE.

Section "Administrating the Controller" in **Admin Guide** indicates that there is no configuration needed to access the Controller via an SSH session. It identifies all of the algorithms and key exchange methods supported by the TOE's SSHv2 server implementation which are consistent with the **ST**.

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

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

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

2.3.18.8 NDcPP22e:FCS_SSHS_EXT.1.8

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

a) Both thresholds are checked by the TOE.

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

Section 6.3 in the **ST** states that in the distributed TOE, the AP and vSZ-D are SSH clients which communicate to the SSH Server which is the SmartZone controller (SZ/vSZ) via only public key auth (No password-based authentication).

When the AP or vSZ-D joins the controller, it has the ability to transmit a rekey request to the SSH server. The SSH session key will be re-negotiated when either one of the following two conditions is met:

1) The maximum amount of data transmitted over the SSH connection exceeds 1G bytes
2) The maximum amount of connection time exceeds 1 hour.

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

Section "Management Channel between AP/vSZ-D and Controller" in **Admin Guide** indicates that the SSH parameters and algorithms are non-configurable. The rekey limitation is 1 hour or 1 GB of data traffic when the vSZ-D or AP connects to the SZ SSH server as an SSH client. The SSH client or server discards the data packets if the incoming packet size exceeds the packet size limitation; the maximum packet size is 256 KB.

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

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

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

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

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

Testing does not necessarily have to be performed with the threshold configured at the maximum allowed value of one gigabyte of transferred traffic, but the value used for testing shall not exceed one gigabyte. The evaluator

needs to ensure that the rekeying has been initiated by the TOE and not by the SSH client that is connected to the TOE.

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

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

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

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

The evaluator attempted to connect to the TOE using a SSH client alternately waiting an hour and generating 100KB of data to ensure that a rekey happened before each of those thresholds. A special debug command was used to drop the rekey limit from 1 GB to 100KB. The evaluator confirmed that rekeying is performed according to the description in the TSS.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.19 TLS Client Protocol Without Mutual Authentication - per TD0634 & TD0670 (NDcPP22e:FCS_TLSC_EXT.1)

2.3.19.1 NDcPP22e:FCS_TLSC_EXT.1.1

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

Section 6.3 in the **ST** states that the TOE supports RADIUS over TLS as specified in RFC 6614. The TOE performs the function of a RADSEC client and conducts mutual authentication via X.509v3 certificates. The TOE supports only TLS 1.2 and rejects all other TLS and SSL versions. The TOE supports the ciphersuites defined in FCS_TLSC_EXT.1.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which ensures that the TLS version and available ciphersuites are restricted to those supported by the TOE.

Section “RadSec (RADIUS over TLS)” in the **Admin Guide** provides the steps for configuring RadSec and identifies the supported TLS cipher suites which are not user-configurable. This list matches the ciphersuites identified in the FCS_TLSC_EXT.1 requirement.

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

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

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

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

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

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

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

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

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

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

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

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

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

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

Test 1- The evaluator configured a test server to accept each ciphersuite allowed by the PP, a single ciphersuite at a time. While the test server listened for the single configured ciphersuite, the evaluator caused the TOE to attempt a connection to the test Server. The evaluator confirmed that each claimed TLS ciphersuite resulted in a successful connection.

Test 2 - The evaluator configured the TOE to connect to a test server and attempted two connections. During the first TLS negotiation the test server sent a valid certificate chaining to a CA known by the TOE. The certificate included the Server Authentication extended key usage (EKU) field. The PCAP for this part of the test shows that the client issued the Change Cipher Spec and Encrypted Handshake messages, which signify that the TLS connection is successful.

During the second connection, the server presented a certificate chaining to a CA known by the TOE. However, the certificate did not include the Server Authentication extended key usage (EKU) field. In the PCAP for this part of the test, the client generates a fatal alert and closes the connection.

Test 3 – The evaluator configured the TOE to negotiate an RSA cipher but to then send an ECDSA certificate. The TOE rejected the connection.

Test 4a - The evaluator configured the TOE to communicate with a test server that sends only a TLS_NULL_WITH_NULL_NULL ciphersuite in the server hello. The evaluator then attempted to establish a TLS session from the TOE to the test server and observed that the connection failed.

Test 4b - The evaluator configured the TOE to connect to a test server using TLS. During the connection the evaluator caused the server to choose a ciphersuite that the TOE did not offer in its Client Hello handshake message. The evaluator confirmed that the client rejected the connection.

Test 5a - The evaluator configured the TOE to connect to a test server using TLS. During the connection the evaluator caused the server to use a TLS version in the Server Hello that is a non-supported TLS version (version 1.4 represented by two bytes 0x0305). The evaluator verified that the client rejected the connection.

Test 5b - The evaluator configured the TOE to connect to a test server using TLS. During the connection the evaluator caused the server to modify the signature block in the Server's Key Exchange handshake message. The evaluator confirmed that the client rejected the connection.

Test 6a - The evaluator attempted a connection attempt to the TOE where the evaluator modified a byte in the Server Finished handshake message, verified that the server rejected the connection attempt after receiving the server Finished message and no application data was exchanged.

Test 6b - The evaluator garbled a message between the TOE and its TLS peer. The modification occurred after the Server sent the ChangeCipherSpec message. The evaluator observed that the Client denies the connection. Due to the nature of the error, regardless of whether the TOE is the client or server, the client is always the first to recognize the error.

Test 6c - The evaluator configured the TOE to connect to a test server using TLS. During the connection the evaluator caused the server to modify one byte in the server's nonce in the Server Hello handshake message. The evaluator verified that the client rejected the connection.

2.3.19.2 NDcPP22e:FCS_TLSC_EXT.1.2

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

Note that where a TLS channel is being used between components of a distributed TOE for FPT_ITT.1, the requirements to have the reference identifier established by the user are relaxed and the identifier may also be established through a 'Gatekeeper' discovery process. The TSS should describe the discovery process and highlight how the reference identifier is supplied to the 'joining' component. Where the secure channel is being used between components of a distributed TOE for FPT_ITT.1 and the ST author selected attributes from RFC 5280, the evaluator shall ensure the TSS describes which attribute type, or combination of attributes types, are used by the client to match the presented identifier with the configured identifier. The evaluator shall ensure the TSS presents an argument how the attribute type, or combination of attribute types, uniquely identify the remote TOE component; and the evaluator shall verify the attribute type, or combination of attribute types, is sufficient to support unique identification of the maximum supported number of TOE components.

If IP addresses are supported in the CN as reference identifiers, the evaluator shall ensure that the TSS describes the TOE's conversion of the text representation of the IP address in the CN to a binary representation of the IP

address in network byte order. The evaluator shall also ensure that the TSS describes whether canonical format (RFC5952 for IPv6, RFC 3986 for IPv4) is enforced.

Section 6.3 in the **ST** states that the reference identifier is configured by the administrator of the TOE. The Controller initiates the RADSEC connection and verifies the presented identifier (FQDN) of the server in the SAN field of the certificate that the server presents to it. If the certificate is not valid, the connection is dropped. IP address and wild card are not supported. OCSP is supported to verify the validity of the certificate.

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

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

Section “RadSec (RADIUS over TLS)” in the **Admin Guide** provides instructions for configuring the CN/SAN identity with the FQDN. When TLS encryption is enabled, the CN/SAN becomes a mandatory field.

TLS is not supported for distributed TOE communication.

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

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

or

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

or

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

Note that for some tests additional conditions apply.

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

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

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

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

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

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

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

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

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

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

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

2) [conditional]: The evaluator shall present a server certificate containing a wildcard in the left-most label (e.g. *.example.com). The evaluator shall configure the reference identifier with a single left-most label (e.g. foo.example.com) and verify that the connection succeeds if wildcards are supported or fails if wildcards are not

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

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

Test 6 [conditional]: If IP address identifiers 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 192.168.0.1.20, CN=2001:0DB8:0000:0000:0008:0800:200C:*when connecting to 2001:0DB8:0000:0000:0008:0800:200C:417A). The certificate shall not contain the SAN extension. The evaluator shall verify that the connection fails. The evaluator shall repeat this test for each supported IP address version (e.g. IPv4, IPv6). (TD0634 applied)

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

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

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

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

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

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

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

Test 1 - The evaluator configured the TOE to connect with the test server using TLS alternately configured with matching certificate identifiers and certificate identifiers that do not match the reference identifier in either the SAN or the CN. The evaluator confirmed that the connections were only successful when the identifier fulfilled the required rules.

Test 2-5 – These tests were demonstrated as part of the script results performed in test 1.

Test 6 – Not applicable. IP address reference identifiers are not supported by the TOE.

Test 7 -The TOE does not support TLS for FPT_ITT.1.

2.3.19.3 NDcPP22e:FCS_TLSC_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

Test 2: The evaluator shall then change the presented certificate(s) so that validation fails and show that the certificate is not automatically accepted. The evaluator shall repeat this test to cover the selected types of failure defined in the SFR (i.e. the selected ones from failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, failed determination of the revocation status). The evaluator performs the action indicated in the SFR selection observing the TSF resulting in the expected state for the trusted channel (e.g. trusted channel was established) covering the types of failure for which an override mechanism is defined.

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

Test 1 - The successful validation of the chain after the correct CA is added to the truststore was demonstrated in FCS_TLSC_EXT.1.1.

Test 2 - The failure to validate the certificate due to invalid cert parameters was demonstrated throughout the tests performed for FCS_TLSC_EXT.1 and FIA_X509_EXT.1/Rev testing, including the failed matching of the reference identifier, failed validation of the certificate path, failed validation of the expiration date, and failed determination of the revocation status. The TOE does not support administrator override of any of these certificate checks.

Test 3 – Not applicable. The TOE does not support administrator override of any of the certificate checks.

2.3.19.4 NDcPP22e:FCS_TLSC_EXT.1.4

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

Section 6.3 of the **ST** states that the TOE will present the supported elliptic curve extensions with the following NIST curves: secp256r1, secp384r1, secp521r. The TLS ciphersuites and NIST curves are not configurable.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which ensures that the TLS version, available ciphersuites and elliptic curves extensions are restricted to those supported by the TOE.

Section “RadSec (RADIUS over TLS)” in the **Admin Guide** provides the steps for configuring RadSec and identifies the supported TLS cipher suites which are not user-configurable. This list matches the ciphersuites identified in the FCS_TLSC_EXT.1 requirement.

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

Test 1 - The evaluator configured the TOE to connect to a test server using TLS with each TOE Supported Elliptic Curves/Supported Groups Extension. In each case the TLS connection was successful.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.3.20 TLS Server Protocol Without Mutual Authentication - per TD0635 (NDcPP22e:FCS_TLSS_EXT.1)

2.3.20.1 NDcPP22e:FCS_TLSS_EXT.1.1

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

Section 6.3 of the **ST** states that the TOE supports TLSS in support of remote administration via the Web UI using HTTPS. The TLS Server is also provided via the Controller and supports TLS 1.2. It supports the ciphersuites defined in FCS_TLSS_EXT.1.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which ensures that the TLS version, available ciphersuites and elliptic curves extensions are restricted to those supported by the TOE.

Section "Administrating the Controller Remotely" in the **Admin Guide** states that the Controller provides remote administration of the system through secure communication using the Web UI over TLS/HTTPS. There is no specific configuration needed to access the Web UI session. This section also identifies the use of TLS v1.2 and the supported ciphersuites which are consistent with those identified in the requirement.

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

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

send a Client Hello to the server containing only the TLS_NULL_WITH_NULL_NULL ciphersuite and verify that the server denies the connection.

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

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

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

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

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

Test 1 - The evaluator attempted to connect to the TOE using each of the ciphersuites identified by the Protection Profile. A packet capture was obtained for each connection attempt. The evaluator verified that successful connections were only established with the claimed ciphersuites.

Test 2 - The evaluator used the openssl s_client to attempt to connect to the TOE using the TLS_NULL_WITH_NULL_NULL ciphersuite and observed that the connection failed. The evaluator also used the openssl s_client to attempt to connect to the TOE using a ciphersuite not supported by the TOE and observed that the connection failed.

Test 3a - The evaluator attempted a connection to the TOE where the evaluator modified a byte in the Client Finished handshake message, verified that the server rejected the connection attempt after receiving the client Finished message and no application data was exchanged.

Test 3b - The evaluator viewed a successful TLS connection and saw 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 examined the Finished message and confirmed that it did 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, which it did not.

2.3.20.2 NDcPP22e:FCS_TLSS_EXT.1.2

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

Section 6.3 of the **ST** states that the TLS Server is also provided via the Controller and supports TLS 1.2. It supports the ciphersuites defined in FCS_TLSS_EXT.1. The TOE will deny connections from clients that request SSL2.0, SSL3.0, TLS1.0 and TLS1.1.

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

Section “FIPS Mode Overview” in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section “Using FIPS-related CLI commands” of the **Admin Guide** describes the steps to enable FIPS mode which ensures that the TLS version, available ciphersuites and elliptic curves extensions are restricted to those supported by the TOE.

Section “Administrating the Controller Remotely” in the **Admin Guide** states that the Controller provides remote administration of the system through secure communication using the Web UI over TLS/HTTPS. There is no specific configuration needed to access the Web UI session. This section also identifies the use of TLS v1.2 and the supported ciphersuites which are consistent with those identified in the requirement.

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

The evaluator alternately attempted to connect to the TOE using a control case (TLS not specified), SSL2.0, SSL3.0, TLSv1.0, TLSv1.1 and TLSv1.2 and confirmed that connections were accepted only when claimed TLS versions are used.

2.3.20.3 NDcPP22e:FCS_TLSS_EXT.1.3

TSS Assurance Activities: If using ECDHE and/or DHE ciphers, the evaluator shall verify that the TSS lists all EC Diffie-Hellman curves and/or Diffie-Hellman groups used in the key establishment by the TOE when acting as a TLS Server. For example, if the TOE supports TLS_DHE_RSA_WITH_AES_128_CBC_SHA cipher and Diffie-Hellman parameters with size 2048 bits, then list Diffie-Hellman Group 14.

Section 6.3 of the **ST** states that the TOE performs key establishment with ECDH over secp384r1, and DH parameters of size 3072 bits.

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

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which ensures that the TLS version, available ciphersuites and elliptic curves extensions are restricted to those supported by the TOE.

Section "Administrating the Controller Remotely" in the **Admin Guide** states that the Controller provides remote administration of the system through secure communication using the Web UI over TLS/HTTPS. There is no specific configuration needed to access the Web UI session. This section also identifies the use of TLS v1.2 and the supported ciphersuites which are consistent with those identified in the requirement.

Testing Assurance Activities: Test 1: [conditional] If ECDHE ciphersuites are supported:

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

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

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

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

Test 1 - The evaluator attempted to establish a TLS session with the TOE with the client specifying just one of the supported ECDHE key exchange methods in the Client Hello. The evaluator observed that the TOE successfully established connections using all of the claimed key exchanges.

Test 2 - The evaluator attempted to establish a TLS session with the TOE with the client specifying just one of the supported DHE key exchange methods in the Client Hello. The evaluator observed that the TOE successfully established connections using all of the claimed key exchanges.

Test 3 – Not applicable. The TOE does not support RSA key exchange methods for TLS as a server.

2.3.20.4 NDcPP22e:FCS_TLSS_EXT.1.4

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

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

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

Section 6.3 of the **ST** states that the TLS Server does not support session resumption based on session IDs or session tickets.

Guidance Assurance Activities: None Defined

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

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

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

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

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

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

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

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

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

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

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

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

a) The evaluator shall permit a successful TLS handshake to occur in which a session ticket is exchanged with the non-TOE client. The evaluator shall then attempt to correctly reuse the previous session by sending the session ticket in the ClientHello. The evaluator shall confirm that the TOE responds with a ServerHello with an empty

SessionTicket extension, NewSessionTicket, ChangeCipherSpec and Finished messages (as seen in figure 2 of RFC 5077).

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

Test 1: The evaluator initiated a connection attempt in which the Client Hello has a zero-length Session ID and a zero-length Session Ticket extension. The TOE did not send a NewSessionTicket message back at any point in the handshake process, but there was a non-zero session ID returned. The evaluator attempted a connection reusing the returned session ID and a new session ID was returned, demonstrating that session resumption was not supported.

Test 2: This test is not applicable as session resumption with session IDs is not supported.

Test 3 - This test is not applicable as session resumption with session tickets is not supported.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.4 User Data Protection (FDP)

2.4.1 Subset Information Flow Control (WIDS10:FDP_IFC.1)

2.4.1.1 WIDS10:FDP_IFC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: If this functionality is configurable, the evaluator shall verify that the operational guidance provides instructions on how to configure the TOE to monitor different types of IEEE 802.11 frame types and subtypes.

The **Admin Guide** does not provide instructions to configure monitoring of different types of IEEE 802.11 frame types and subtypes as is monitors all types by default.

Testing Assurance Activities: Test 1: Deploy an allowlisted AP/WIDS Start a traffic capture from the AP/WIDS sensor

Send a set number of frames to the sensor for all IEEE 802.11 a, b, g, n, ac frame types and subtypes from/to the following:

* authorized APs and authorized EUDs

* authorized APs and unauthorized EUDs

* unauthorized APs and authorized EUDs

Verify that there are frames from all the types and subtypes in the capture.

The TOE sensors were set up in Gossamer's lab which has an array of APs and EUDs covering all 802.11 frame types.

The TOE sensors only have limited packet capture functionality available to an administrator where wireless packets for a single defined MAC address can be collected. The TOE sensors however do have an available command when using SSH (only enabled via special debug commands and not a part of the evaluation) that can output recently captured 802.11 beacon frame details.

The evaluator ran this command for both 2.4GHz and 5GHz and saw that all the 802.11 frames were captured by the TOE for all frame types, including IEEE802.11 a, b, g, n, ac and also ax.

After a few minutes, these detected devices would appear in the rogue devices list and the 802.11 would be one of the details collected. This can be seen throughout the reports generated for FAU_SAA.1.

Whether a device is authorized or unauthorized is determined by the MAC addresses set in the allow-list. The MAC address is checked after the frame is captured to see if it matches a mac address in the configured list. This will then also be displayed in the rogue device report. Both detection of authorized and unauthorized APs and EUDS were demonstrated in WIDS10:FAU_INV_EXT.1-t1. Due to this implementation of determining authorization after a frame is viewed there is no difference in how a TOE would monitor authorized vs unauthorized devices over the air. Therefore all device authorization types are equivalent over the air and therefore don't need to be explicitly tested here.

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.5 Identification and Authentication (FIA)

2.5.1 802.1X Port Access Entity (Authenticator) Authentication (WLANAS10:FIA_8021X_EXT.1)

2.5.1.1 WLANAS10:FIA_8021X_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.1.2 WLANAS10:FIA_8021X_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.1.3 WLANAS10:FIA_8021X_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: In order to show that the TSF implements the 802.1X-2010 standard correctly, the evaluator will ensure that the TSS contains the following information:

The sections (clauses) of the standard that the TOE implements

- For each identified section, any options selected in the implementation allowed by the standards are specified

- For each identified section, any non-conformance is identified and described, including a justification for the non-conformance

Because the connection to the RADIUS server will be contained in an IPsec or RadSec (TLS) tunnel, the security mechanisms detailed in the RFCs identified in the requirement are not relied on to provide protection for these communications. Consequently, no extensive analysis of the RFCs is required. However, the evaluator will ensure that the TSS describes the measures (documentation, testing) that are taken by the product developer to ensure that the TOE conforms to the RFCs listed in this requirement.

Section 6.3 of the **ST** (WLANAS10:FCS_CKM.2/GTK) states the TOE supports cryptographic key distribution for 802.11 PMK key reception from an 802.1X authentication server, AES Key Wrap in EAPOL-Key frame and Group Key Handshake.

The KEY Generation/Establishment portion of Section 6.3 further states in the distributed TOE deployment, AP acts as the authenticator and derives the 802.11 keys. With WPA2/WPA3-enterprise connections via 802.1X, all communications between the AP and wireless client is encapsulated in EAP and all traffic between AP and the controller is secured via SSH. The controller to RADIUS server communication is secured via RADSEC. The TOE supports standard WPA2/WPA3-enterprise with 802.11i where 4 way EAP handshake is between the Authenticator (AP) and the wireless client.

AP sends the EAPOL identity to the wireless client, which responds with a radius access request. Upon successful authentication, radius access accept message is received by the client. A PMK is then generated by the RADIUS server for Authenticator and client after which a PTK is derived. The Authenticator also generates a GTK. If the 4-way handshake fails, the client will not be allowed to access the network or connect to the AP.

Upon a successful 4-way handshake, the Authenticator will allow for WLAN data to pass through the system to the Controller in a tunnel architecture or to intended destination in distributed architecture.

The PTK (total 384 bits) is derived into three parts. The second part is KEK and used to encrypt GTK to be sent as 3rd message in WPA2/WPA3 handshake. Third part is TK, which is actually used to encrypt/decrypt communication between both AP and Client.

The Ruckus implementation of PTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.3 → Pairwise key hierarchy (PTK derivation using the PRF function)

The Ruckus implementation of GTK generation complies to the following sections of IEEE 802.11-2012:

- Section 4.10.3.2 --> AKM operations with AS (WPA2/WPA3 4-way handshake explaining PTK/GTK transfer)
- Section 11.6.1.4 --> Group key hierarchy (GTK derivation using the PRF function)

When the RADIUS protocol is used between the TOE and the authentication server, the MS-MPPE-Recv-Key attribute (vendor-id = 17; see Section 2.4.3 in IETF RFC 2548-1999 [B30]) is used to transport the PMK to TOE. The GTK is distributed by using AES Key Wrap in an EAPOL-Key frame in accordance with the 802.11-2012 standard for the packet format and timing considerations.

Certification testing performed by the Wi-Fi Alliance demonstrates that the TOE implements the IEEE 802.11-2012 standard correctly. Refer to the Wi-Fi Alliance certificates for compliance, https://www.wi-fi.org/certification.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will perform the following tests:

Test 1: The evaluator will demonstrate that a wireless client has no access to the test network. After successfully authenticating with a RADIUS server through the TOE, the evaluator will demonstrate that the wireless client does have access to the test network.

Test 2: The evaluator will demonstrate that a wireless client has no access to the test network. The evaluator will attempt to authenticate using an invalid client certificate, such that the EAP-TLS negotiation fails. This should result in the wireless client still being unable to access the test network.

Test 3: The evaluator will demonstrate that a wireless client has no access to the test network. The evaluator will attempt to authenticate using an invalid RADIUS certificate, such that the EAP-TLS negotiation fails. This should result in the wireless client still being unable to access the test network.

Note: Tests 2 and 3 above are not tests that 'EAP-TLS works,' although that is a by-product of the test. The test is actually that a failed authentication (under two failure modes) results in denial of access to the network, which demonstrates the enforcement of FIA_8021X_EXT.1.3.

Test 1 – The evaluator first verified that the wireless client has no access to the test network and then successfully authenticated with the RADIUS server and confirmed that the wireless client had access to the test network.

Test 2 - The evaluator first verified that the wireless client has no access to the test network. The evaluator then attempted to authenticate with an invalid client cert and found that negotiation failed and that the wireless client was still unable to access the test network.

Test 3 - The evaluator first verified that the wireless client has no access to the test network. The evaluator then attempted to authenticate with an invalid RADIUS cert and found that negotiation failed and that the wireless client was still unable to access the test network.

2.5.2 Authentication Failure Management (NDcPP22e:FIA_AFL.1)

2.5.2.1 NDcPP22e:FIA_AFL.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.2.2 NDcPP22e:FIA_AFL.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.5 in the **ST** states that the controller provides remote administration of the system via secure communication channel (WebGUI via HTTPS and CLI via SSH). Unsuccessful attempts beyond an administrator configured limit will result in access being denied until an administrator configured time duration has elapsed. Local system administrator can login into the TOE while remote administrators are blocked. The local administrator account never gets locked out.

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

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

Section “Locking an Administrator Account” in the **Admin Guide** describes how an administrator can configure administrator accounts to be forcefully locked when there are repeated attempts to access the account by unauthorized users. This is typically applicable in situations when the user name entered is correct but password is wrong. A user is locked out for the account lockout time after the configured number of failed login attempts.

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

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

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

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

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

Test 1 and Test 2 – The evaluator configured the TOE with a maximum number of failed login attempts to 2, and lockout duration to 3 minutes. The evaluator then logged in 3 times with an incorrect password and confirmed that the account was locked as expected. After 3 minutes, the evaluator successfully logged in with the correct password.

Note that time period, not administrator action is the selection for FIA_AFL.1.2.

2.5.3 Password Management (NDcPP22e:FIA_PMG_EXT.1)

2.5.3.1 NDcPP22e:FIA_PMG_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.5 of **ST** states the TOE supports a password enforcement configuration where the minimum password length can be set by an administrator. The default minimum length is 8 and this can set to as high as 64 characters. Passwords can be created using any alphabetic, numeric, and a range of special characters (identified in FIA_PMG_EXT.1).

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

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

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

Section “Password Management” in **Admin Guide** describes how to configure the minimum password length and provides security administrators with an example of the composition of a strong password. An administrator password can change the administrator password from the controller interface and from the command-line interface, including for the AP and vSZ-D. Passwords can be composed of any combination of uppercase and lowercase letters, numbers, and the following special characters: ! @ # $ % ^ & *

( ) - _ = + [ ] {}; : ' " , <> / ?.. (No other special characters are allowed). To cover FIPS and CC password length range, the minimum password length ranges from 8 to 63 and the maximum length is capped at 64. The administrator login password of the AP zones are pushed from the controller. Therefore, the controller validates the admin login passwords length of AP zones before pushing them into APs. The administrator login password of the dataplane is identical to the controller, so it need not be validated.

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

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

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

Test 1 - The evaluator verified that an 8 character long password is successfully accepted. The evaluator also verified that all selected special characters, all lowercase letters, all uppercase letters, and all numbers are supported.

Test 2 - The evaluator found that passwords with less than 8 characters are not accepted. Then the evaluator verified that an unselected special character is not accepted. Finally the evaluator attempted an invalid password length of 65 characters and found that no more than the max of 64 characters could be entered.

2.5.4 Pre-Shared Key Composition (WLANAS10:FIA_PSK_EXT.1)

2.5.4.1 WLANAS10:FIA_PSK_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.4.2 WLANAS10:FIA_PSK_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.4.3 WLANAS10:FIA_PSK_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will verify that the TSS describes

1. the protocols that can use pre-shared keys and that these are consistent with the selections made in FIA_PSK_EXT.1.1.

2. the allowable values for pre-shared keys and that they are consistent with the selections made in FIA_PSK_EXT.1.2.

3. the way bit-based pre-shared keys are procured and that it is consistent with the selections made in FIA_PSK_EXT.1.3.

Section 6.5 of **ST** states The TOE supports use of preshared keys for IPsec, 802.11 WPA2-PSK/ WPA3_SAE SSID (single PSK and dynamic PSK). The TOE accepts both bit based and text based keys for Pre-shared keys. A TOE administrator specifies the PSK in the controller configuration. For IPsec, the PSK includes a combination of upper and lower-case letters, numbers, and supported special characters and must be 44-128 for Hex characters and 8-64 for ASCII. For 802.11 WPA2-PSK/WPA3_SAE SSID text based keys, the TOE accepts 22-63 characters with any combination of upper and lower case letters, numbers, and special characters (that include: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”). For bit based pre-shared keys, 64 Hex characters must be used. The TOE does not generate bit based keys. Text based/ASCII WPA2/ WPA3_SAE passwords are converted to 256 bit based on HMAC SHA1 and IPsec passwords are converted using PRF. The above information is consistent with the selections made in FIA_PSK_EXT.1.

Component Guidance Assurance Activities: The evaluator will examine the operational guidance to determine that it provides guidance to administrators on the composition of strong text-based pre-shared keys, and (if the selection indicates keys of various lengths can be entered) that it provides information on the range of lengths supported. The guidance must specify the allowable characters for pre-shared keys, and that list must be a superset of the list contained in FIA_PSK_EXT.1.2.

The evaluator will confirm the operational guidance contains instructions for either entering bit-based preshared keys for each protocol identified in the requirement or for generating a bit-based pre-shared key (or both).

Section “Configuring System IPsec using Preshared Key” in **Admin Guide** describes how to enable IPsec from the controller to external systems using a pre-shared key and provides security administrators with an example of the composition of a strong text-based pre-shared key. The text-based pre-shared key has a valid length range for ASCII from 8 through 64 and bit-based text is from 44 through 128.

Section “Enabling Trusted Channel Using IEEE 802.11-2012 (WPA2) Standards” in the **Admin Guide** describes how to enable a secure and trusted channel for communication by using IEEE 802.11-2012 (WPA2) standards. This connection is initiated from the beginning by itself with WPA2 four-way handshake. This is per WPA2 standard, and no manual intervention needed. For the preshared key, the Hexadecimal (0 to 9 and A to F) characters are only allowed, no other ascii characters. 64 hexadecimal characters must be used. 22 to 63 text based characters are also supported and an example of the composition of a strong text-based pre-shared key is provided for security administrators. Section “Creating a WLAN WPA3 WLAN2/WPA3 Mixed Profile” describes how to perform the same steps using WPA3-SAE.

Component Testing Assurance Activities: The evaluator will also perform the following tests for each protocol (or instantiation of a protocol, if performed by a different implementation on the TOE). Note that one or more of these tests can be performed with a single test case.

Test 1: The evaluator will compose a pre-shared key of 22 characters that contains a combination of the allowed characters in accordance with the operational guidance and demonstrates that a successful protocol negotiation can be performed with the key.

Test 2: [conditional]: If the TOE supports pre-shared keys of multiple lengths, the evaluator will repeat Test 1 using the minimum length; the maximum length; a length inside the allowable range; and invalid lengths beyond the supported range (both higher and lower). The minimum, maximum, and included length tests should be successful, and the invalid lengths must be rejected by the TOE.

Test 3: [conditional]: If the TOE does not generate bit-based pre-shared keys, the evaluator will obtain a bit-based pre-shared key of the appropriate length and enter it according to the instructions in the operational guidance. The evaluator will then demonstrate that a successful protocol negotiation can be performed with the key.

Test 4: [conditional]: If the TOE does generate bit-based pre-shared keys, the evaluator will generate a bit-based pre-shared key of the appropriate length and use it according to the instructions in the operational guidance. The evaluator will then demonstrate that a successful protocol negotiation can be performed with the key.

Test 1 - the evaluator configured a 22 character pre-shared key on the TOE and performed a successful protocol connection using this key. This test was repeated for WPA3-SAE, WPA2-PSK, WPA3 Enterprise, and WPA2 Enterprise.

Test 2 - The evaluator attempted to configure several different pre-shared keys and ensured that the various conditions are met with the expected outcome.

Test 3 - The evaluator configured the TOE to accept a 256 bit hex key and then performed a successful protocol connection using the key.

Test 4 – Not applicable as the TOE does not generate bit-based keys.

2.5.5 Re-Authenticating (WLANAS10:FIA_UAU.6)

2.5.5.1 WLANAS10:FIA_UAU.6.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: The evaluator will attempt to change their password as directed by the operational guidance. While making this attempt, the evaluator will verify that re-authentication is required. If other re-authentication conditions are specified, the evaluator will cause those conditions to occur and verify that the TSF re-authenticates the authenticated user.

The evaluator changed the administrative user password on the TOE, and was immediately required to re-authenticate.

2.5.6 Protected Authentication Feedback (NDcPP22e:FIA_UAU.7)

2.5.6.1 NDcPP22e:FIA_UAU.7.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

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

Section “Administrating the Controller” of the ST demonstrates the login for both local CLI and remote SSH or HTTPS. The provided screenshots demonstrating the login show that the authentication data is obscured by default.

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

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

Refer to NDcPP21:FIA_UIA_EXT.1-t1. The evaluator found that no password feedback is provided when logging into the local console.

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

2.5.7.1 NDcPP22e:FIA_UAU_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

This assurance activity has been met by NDcPP22E:FIA_UIA_EXT.1.

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

This assurance activity has been met by NDcPP21:FIA_UIA_EXT.1.

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

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

2.5.8 User Identification and Authentication (NDcPP22e:FIA_UIA_EXT.1)

2.5.8.1 NDcPP22e:FIA_UIA_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.8.2 NDcPP22e:FIA_UIA_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

Section 6.5 in the **ST** states that the TOE requires users to be identified and authenticated before they can use functions mediated by the TOE, except to display a message of the day banner without identification or

authentication. The TOE requires all administrators of the system to login via credentials (username & password) before granting access to the system. The Controller provides remote administration of the system via secure communication channel (WebGUI via HTTPS and CLI via SSH). The AP and vSZ-D do not support authentication of Security Administrators and there are no unauthenticated services/services supported by these components.

Section 7 "Requirement Allocation" in the **ST** provides a mapping of the distributed TOE components to the SFRs in this **ST**. This TOE is a distributed TOE consistent with Use Case 3 as defined in the NDcPP22E.

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

Section "Administrating the Controller" in the **Admin Guide** describes how the administrator can log in to and administer the TOE both locally via the local console and remotely via an SSH CLI session or an HTTPS Web UI session. There is no specific configuration needed to access the SSH and WebUI session, it is enabled by default.

Section "Management Channel Between AP/vSZ-D and Controller" in the **Admin Guide** describes how the AP and vSZ-D are SSH clients that communicate to the SSH Server which is the SZ controller.

Section "Joining AP to (v)SZ Controller" in the **Admin Guide** indicates that once an AP is approved an SSH tunnel will be formed across the AP and WLAN controller using public key auth (no password based authentication) and this SSH tunnel will be utilized for all management communications between the AP and Controller.

Section "Joining vSZ-D to the vSZ Controller" in the **Admin Guide** indicates that once a vSZ-D discovers the controller and is approved, communication between the vSZ-D and vSZ controller is through SSH only.

Section "Password Management" in **Admin Guide** describes how to configure the minimum password length and provides security administrators with an example of the composition of a strong password. An administrator password can change the administrator password from the controller interface and from the command-line interface, including for the AP and vSZ-D. Passwords can be composed of any combination of uppercase and lowercase letters, numbers, and the following special characters: ["!", "@", "#", "$", "%", "^", "&", "*", "(", ")". (No other special characters are allowed). To cover FIPS and CC password length range, the minimum password length ranges from 8 to 63 and the maximum length is capped at 64. The administrator login password of the AP zones are pushed from the controller. Therefore, the controller validates the admin login passwords length of AP zones before pushing them into APs. The administrator login password of the dataplane is identical to the controller, so it need not be validated.

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

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

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

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

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

Test 1 – The evaluator first attempted to log into the local console with an incorrect password and found that the login failed. The evaluator then logged in with the correct password successfully. The evaluator then logged out. The evaluator next attempted to log into the SSH CLI with an incorrect password and found that the login failed. The evaluator then logged in with the correct password successfully. The evaluator then logged out. Finally, the evaluator attempted to log into the Web UI with an incorrect password and found that the login failed. The evaluator then logged in with the correct password successfully. The evaluator then logged out.

Test 2 – The evaluator observed during test 1 that the only service available to an administrator prior to login was viewing the warning banner and initiating a login.

Test 3 - The evaluator observed during test 1 that the only service available to an administrator prior to login was viewing the warning banner and initiating a login.

Test 4 – This test was demonstrated in test 1.

2.5.9 X.509 Certificate Validation (NDcPP22e:FIA_X509_EXT.1/ITT)

2.5.9.1 NDcPP22e:FIA_X509_EXT.1.1/ITT

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall demonstrate that checking the validity of a certificate is performed when a certificate is used in an authentication step. It is not sufficient to verify the status of a X.509

certificate only when it is loaded onto the device. The evaluator shall perform the following tests for FIA_X509_EXT.1.1/ITT:

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

a) Test 1a: The evaluator shall present the TOE with a valid chain of certificates (terminating in a trusted CA certificate) as needed to validate the leaf certificate to be used in the function, and shall use this chain to demonstrate that the function succeeds.

Test 1a shall be designed in a way that the chain can be 'broken' in Test 1b by either being able to remove the trust anchor from the TOEs trust store, or by setting up the trust store in a way that at least one intermediate CA certificate needs to be provided, together with the leaf certificate from outside the TOE, to complete the chain (e.g. by storing only the root CA certificate in the trust store).

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

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

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

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

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

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

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

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

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

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

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

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

(TD0527 12/2020 update applied)

Test 1 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 2 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 3 - Revocation checking is disabled for the IPsec data tunnel for ITT communication.

Test 4 - Revocation checking is disabled for the IPsec data tunnel for ITT communication.

Test 5 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 6 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 7 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 8 – The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

2.5.9.2 NDcPP22e:FIA_X509_EXT.1.2/ITT

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall perform the following tests for FIA_X509_EXT.1.2/ITT. The tests described must be performed in conjunction with the other certificate services assurance activities, including the functions in FIA_X509_EXT.2.1/ITT. 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 two certificates: a self-signed root CA certificate and a leaf (node) certificate. The properties of the certificates in the chain are adjusted as described in each individual test below (and this modification shall be the only invalid aspect of the relevant certificate chain).

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

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

Test 1 - The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Test 2- The IPsec data tunnel for ITT communication leverages the same Strongswan code for certificate verification as the external IPsec connection. As both connections leverage the same code it was only necessary to verify the proper functionality for one IPsec case. The testing of the certificate verification function was demonstrated in FIA_X509_EXT.1/Rev.

Component TSS Assurance Activities: The evaluator shall examine the TSS to ensure it 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). If selected, the TSS shall describe how certificate revocation checking is performed. It is not sufficient to verify the status of a X.509 certificate only when it's loaded onto the device.

Section 6.5 in the **ST** states that the TOE provides X.509 v3 based authentication between the components in accordance to RFC 5280. The TOE components perform certificate validation when establishing an IPsec session which includes verifying the extendedKeyUsage field in the certificates that are exchanged. The TOE rejects the connection if certificate validation fails. The TOE does not support revocation checking for certificate validation between the distributed components.

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

Section “Configuring Ruckus GRE and IPsec in the WLAN” in the **Admin Guide** states that Ruckus GRE and IPsec is a configuration of the IPsec tunnel between the AP and the SZ or vSZ-D and provides instructions for configuring the IPsec tunnel between these components using certificates.

The **ST** does not claim any rules for extendedKeyUsage for IPsec or revocation for the distributed TOE channel.

Component Testing Assurance Activities: None Defined

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

2.5.10.1 NDcPP22e:FIA_X509_EXT.1.1/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

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

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

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

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

c) Test 3: The evaluator shall test that the TOE can properly handle revoked certificates - conditional on whether CRL or OCSP is selected; if both are selected, then a test shall be performed for each method. The evaluator shall test revocation of the peer certificate and revocation of the peer intermediate CA certificate i.e. the intermediate CA certificate should be revoked by the root CA. The evaluator shall ensure that a valid certificate is used, and that the validation function succeeds. The evaluator then attempts the test with a certificate that has been revoked (for each method chosen in the selection) to ensure when the certificate is no longer valid that the validation function

fails. Revocation checking is only applied to certificates that are not designated as trust anchors. Therefore the revoked certificate(s) used for testing shall not be a trust anchor.

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

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

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

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

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

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

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

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

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

by a trusted EC root CA. The evaluator shall attempt to load the certificate into the trust store and observe that it is rejected, and not added to the TOE's trust store.

(TD0527 12/2020 update applied)

Test 1 – IPsec: The evaluator demonstrated a successful connection with a valid chain of certificates in FCS_IPSEC_EXT.1.14, test 5. For the broken chain, the evaluator configured the test peer to send a certificate signed by a different CA that was not loaded onto the TOE's truststore. The evaluator attempted to connect the IPsec VPN between the test peer and the TOE and viewed that the connection failed as expected.

RadSec: This successful validation of the chain after the correct CA is added to the truststore was demonstrated in FCS_TLSC_EXT.1.1, test 1. For the broken chain, the evaluator configured the TOE to not have the trusted root CA used by the test server to anchor all of its certificates. The evaluator attempted to connect the TLSC TOE client to the test server and confirmed that the connection failed.

Test 2 – IPsec: The evaluator alternately configured a test peer to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection only succeeded if there were no expired certificates.

RadSec: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that is expired, and 3) issued by an intermediate CA that is expired. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that a connection only succeeded if there were no expired certificates.

Test 3 – IPsec: The evaluator alternately configured a test peer to send an authentication certificate 1) that is valid, 2) that is revoked, and 3) issued by an intermediate CA that is revoked. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that a connection only succeeded if there were no revoked certificates.

Radsec: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that is revoked, and 3) issued by an intermediate CA that is revoked. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that a connection only succeeded if there were no revoked certificates.

Test 4 – IPsec: The evaluator alternately configured a test peer to send an authentication certificate 1) that is valid, 2) that has a root that refers to an OCSP revocation server where the signer lacks OCSPSigning, 3) issued by an intermediate CA whose issuer CA refers to an OCSP revocation server where the signer lacks OCSPSigning, and 3) issued by an intermediate CA referring to an OCSP revocation server where the signer lacks OCSPSigning. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection only succeeded if all retrieved OCSP responses are signed using certificates with OCSPSigning.

RadSec: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that has a root that refers to an OCSP revocation server where the signer lacks OCSPSigning, 3) issued by an

intermediate CA whose issuer CA refers to an OCSP revocation server where the signer lacks OCSPSigning, and 3) issued by an intermediate CA referring to an OCSP revocation server where the signer lacks OCSPSigning. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection succeeded only if all retrieved OCSP responses are signed using certificates with OCSPSigning.

Test 5 – IPsec: The evaluator alternately configured a test peer to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that a connection succeeded only if the certificate is not modified/corrupted.

RadSec: The evaluator alternately configured a test server to send an authentication certificate 1) that is valid, 2) that has one byte in the ASN1 field changed, 3) that has one byte in the certificate signature changed, and 4) that has one byte in the certificate public key changed. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection succeeded only if the certificate is not modified/corrupted.

Test 6 - This test was performed in test 5.

Test 7 – This test was performed in test 5.

Test 8 – The following test was performed for IPsec, see the parts below. The RadSec connection does not support ECDSA certificates and therefore this test not applicable for RadSec.

Test 8a - The evaluator configured the test peer to send a valid chain of ECDSA certificates where only the root CA is located in the trust store, and where the ECDSA certificates have a supported named curve. The TOE accepted the certificate as valid.

Test 8b - The evaluator then configured the test peer to send a valid chain of ECDSA certificates where only the root CA is located in the trust store, and where an intermediate CA's ECDSA certificate has an explicit curve. The TOE rejected the certificate as invalid.

Test 8c - The evaluator attempted to load an intermediate CA ECDSA certificate with a supported named curve into the TOE trust store and observed that the certificate can be loaded. The evaluator attempted to load an intermediate CA ECDSA certificate with an explicit curve into the TOE trust store and observed that the certificate could not be loaded.

2.5.10.2 NDcPP22e:FIA_X509_EXT.1.2/Rev

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: The evaluator shall perform the following tests for FIA_X509_EXT.1.2/Rev. The tests described must be performed in conjunction with the other certificate services assurance activities, including the

functions in FIA_X509_EXT.2.1/Rev. The tests for the extendedKeyUsage rules are performed in conjunction with the uses that require those rules. Where the TSS identifies any of the rules for extendedKeyUsage fields (in FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied) then the associated extendedKeyUsage rule testing may be omitted.

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

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

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

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

Test 1 and Test 2 – IPsec: The evaluator alternately configured a test peer to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and confirmed that the connection was rejected in each case.

RadSec: The evaluator alternately configured a test server to send an authentication certificate issued by a Sub CA with no BasicConstraints and with BasicConstraints but the CA Flag set to false. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection was rejected in each case.

Component TSS Assurance Activities: The evaluator shall ensure the TSS describes where the check of validity of the certificates takes place, and that the TSS identifies any of the rules for extendedKeyUsage fields (in

FIA_X509_EXT.1.1) that are not supported by the TOE (i.e. where the ST is therefore claiming that they are trivially satisfied). It is expected that revocation checking is performed when a certificate is used in an authentication step and when performing trusted updates (if selected). It is not necessary to verify the revocation status of X.509 certificates during power-up self-tests (if the option for using X.509 certificates for self-testing is selected).

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

Section 6.5 in the **ST** states that the TOE provides X.509v3 based authentication for communications with external components via IPsec for the audit server and RADSEC in accordance to RFC 5280. The Controller performs the certificate validation when it tries to join an external component. The Controller verifies the extendedKeyUsage field. The TOE supports OCSP to verify the validity of the certificates presented by the external component, including the leaf cert and any intermediate certs received, regardless of the cert chain length. Administrators can upload the CA chain and map which certificates to use for different external connections. The TOE rejects the connection if the certificate validation fails.

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

Section “Configuring System IPsec using Certificates” in **Admin Guide** describes how to enable IPsec from the controller to external systems using certificates. This includes configuring the certificate to use the remote ID, and the OCSP information. Both RSA and ECDSA private keys are supported. The section further describes how to import the certificates used for IPsec as well as the trusted CA used for certificate verification

The **ST** does not claim any rules for extendedKeyUsage for IPsec.

Section “RadSec (RADIUS over TLS)” in the **Admin Guide** provides instructions for configuring the RadSec (RADIUS over TLS) connection using certificates including enabling OCSP for revocation. The section further describes how to import the trusted CA used for certificate verification. The certificate sent by the Radius Server must contain the Server Authentication purpose in the extendedKeyUsage field.

Component Testing Assurance Activities: None Defined

2.5.11 X.509 Certificate Authentication (NDcPP22e:FIA_X509_EXT.2)

2.5.11.1 NDcPP22e:FIA_X509_EXT.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.11.2 NDcPP22e:FIA_X509_EXT.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.5 in the **ST** states that Administrators can upload the CA chain and map which certificates to use for different external connections. The TOE provides X.509v3 based authentication for communications with external components via IPsec for the audit server and RADSEC in accordance to RFC 5280. The Controller performs the certificate validation when it tries to join an external component. The Controller verifies the extendedKeyUsage field. The TOE supports OCSP to verify the validity of the certificates presented by the external component, including the leaf cert and any intermediate certs received, regardless of the cert chain length. The TOE rejects the connection if the certificate validation fails, which would include the OCSP server being unreachable.

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

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in the **Admin Guide** states that Ruckus GRE and IPsec is a configuration of the IPsec tunnel between the AP and the SZ or vSZ-D and provides instructions for configuring the IPsec tunnel between these components using certificates.

The **ST** does not claim revocation for the distributed TOE channel.

Section “Configuring System IPsec using Certificates” in **Admin Guide** describes how to enable IPsec from the controller to external systems using certificates. This includes configuring the certificate to use the remote ID, and the OCSP information. Both RSA and ECDSA private keys are supported. The section further describes how to import the certificates used for IPsec as well as the trusted CA used for certificate verification.

Section “RadSec (RADIUS over TLS)” in the **Admin Guide** provides instructions for configuring the RadSec (RADIUS over TLS) connection using certificates including enabling OCSP for revocation. The section further describes how to import the trusted CA used for certificate verification. The certificate sent by the Radius Server must contain the Server Authentication purpose in the extendedKeyUsage field. This section also describes some reasons certificate verification can fail and is referenced by the other trusted channel sections.

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

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

IPsec: The evaluator alternately configured a test peer to send an authentication certificate with valid/accessible revocation servers and an authentication certificate with revocation information referring to an inaccessible revocation server. In each case, the evaluator then attempted to connect the IPsec VPN between the test peer and the TOE and confirmed that the connection was successful when the revocation server is accessible and when the revocation server is not accessible the connection was rejected.

RadSec: The evaluator alternately configured a test server to send an authentication certificate with valid/accessible revocation servers and an authentication certificate with revocation information referring to an inaccessible revocation server. In each case, the evaluator then attempted to connect the TLSC TOE client to the test server and confirmed that the connection was successful when the revocation server is accessible and when the revocation server is not accessible the connection was rejected.

2.5.12 X.509 Certificate Requests (NDcPP22e:FIA_X509_EXT.3)

2.5.12.1 NDcPP22e:FIA_X509_EXT.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.5.12.2 NDcPP22e:FIA_X509_EXT.3.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Device-specific information was not claimed for FIA_X509_EXT.3 in the **ST**.

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

Section “Generating Certificate Signing Request (CSR)” in the **Admin Guide** provides instructions for creating a certificate signing request and sending it to an CA to purchase an SSL certificate. The instructions include entering the Common Name and Country.

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

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

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

Test 1 – The evaluator generated a CSR and transferred the CSR to a test server. The evaluator then ensured that the CSR conformed to the format specified and provided the public key and other required information.

Test 2 – The evaluator first attempted to validate a response message to a Certification Request without a valid certification path and verified that the function failed. The evaluator then loaded a certificates as trusted CAs needed to validate the response message, and found that the function succeeds.

2.6 Security Management (FMT)

2.6.1 MANAGEMENT OF SECURITY FUNCTIONS BEHAVIOUR (NDCPP22E:FMT_MOF.1/MANUALUPDATE)

2.6.1.1 NDCPP22E:FMT_MOF.1.1/MANUALUPDATE

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

There are no specific requirements for non-distributed TOEs.

Section 6.6 in the **ST** states that the TOE provides the ability to perform administrative functions as an authorized user. All authorized users are granted the Administrator role. Administrators can control the TOE components (Controller (SZ/vSZ), vSZ-D and AP) via the Controller. Only authenticated administrators are allowed access to the TOE to perform administrative functions via WebUI (HTTPS), CLI (SSH) and Local Console. All security functions are available through all interfaces.

The following administrative functions are supported:

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- Ability to configure the authentication failure parameters for FIA_AFL.1
- Ability to manage the cryptographic keys,
- Ability to configure the cryptographic functionality,
- Ability to configure the lifetime for IPsec SAs;
- Ability to configure the interaction between TOE components;
- Ability to set the time which is used for time-stamps;
- Ability to configure NTP;
- Ability to configure the reference identifier for the peer;
- Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors,

- Ability to import X509v3 certificates to the TOE's trust store,
- Ability to manage the trusted public keys database,
- Configure approval/denial of AP and vSZ-D to communicate/join the controller
- Configure the security policy for each wireless network, including:
  - Security type
  - Authentication protocol
  - Client credentials to be used for authentication
  - Service Set Identifier (SSID)
  - If the SSID is broadcasted Frequency band set to [2.4 GHz, 5 GHz]
  - Transmit power level
- Define an inventory of authorized APs based on [MAC addresses]
- Define an inventory of authorized EUDs based on MAC addresses
- Define rules for monitoring and alerting on the wireless traffic
- Define authorized SSID(s)
- Define authorized WLAN authentication schemes
- Define authorized WLAN encryption schemes
- disable transmission of data by wireless sensor,
- Define attack signatures,
- Define rules for overwriting previous packet captures,
- Define the amount of time sensor monitors a specific channel

Section 7 “Requirement Allocation” in the **ST** provides a mapping of the distributed TOE components to the SFRs in this **ST**. This TOE is a distributed TOE consistent with Use Case 3 as defined in the NDcPP21.

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

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

Section “Upgrading (v)SZ Software” in the **Admin Guide** describes how to download the update image from the Ruckus Customer release web site and then upload it to the Controller. The web interface lists the active and inactive upgrade history. Once uploaded, delayed activation/upgrade can be initiated. Upgrading the (v)SZ software can force a restart and therefore, cause a temporary interruption in all the functionalities.

The “Upgrading the AP Software” section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to AP Software are addressed via AP firmware associated with a firmware version which is bundled as part of the (v) SZ Software upgrade image. This section further provides instructions for manually upgrading the AP firmware version. Upgrading the AP software can force a restart and therefore, cause a temporary interruption in all the functionalities.

The “Upgrading the vSZ-D Software” section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to vSZ-D software are addressed through vSZ-D Patch and provides instructions for upgrading the vSZ-D software. Upgrading the (v)SZ-D software can force a restart and therefore, cause a temporary interruption in all the functionalities.

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

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

As demonstrated in FIA_UIA_EXT.1 test 1, no actions other than viewing the warning banner or initiating authentication are available prior to login. The successful update of the TOE was demonstrated in FPT_TUD_EXT.1.

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

2.6.2.1 NDcPP22e:FMT_MTD.1.1/CoreData

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.5 in the **ST** states that the TOE requires users to be identified and authenticated before they can use functions mediated by the TOE, except to display a message of the day banner; this includes being unable to manage the TOE's truststore prior to login. The Controller provides a local console as well as remote administration of the system via secure communication channel (WebGUI via HTTPS and CLI via SSH). The TOE requires all administrators of the system to login via credentials (username & password) before granting access to the system.

Section 6.6 in the **ST** states that the TOE provides the ability to perform administrative functions as an authorized user. All authorized users are granted the Administrator role. Administrators can control the TOE components (Controller (SZ/vSZ), vSZ-D and AP) via the Controller. Only authenticated administrators are allowed access to the TOE to perform administrative functions via WebUI (HTTPS), CLI (SSH) and Local Console. All security functions are available through all interfaces.

The following administrative functions are supported:

- Ability to administer the TOE locally and remotely;
- Ability to configure the access banner;
- Ability to configure the session inactivity time before session termination or locking;
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- Ability to configure the authentication failure parameters for FIA_AFL.1
- Ability to manage the cryptographic keys,
- Ability to configure the cryptographic functionality,
- Ability to configure the lifetime for IPsec SAs;
- Ability to configure the interaction between TOE components;
- Ability to set the time which is used for time-stamps;
- Ability to configure NTP;
- Ability to configure the reference identifier for the peer;
- Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors,
- Ability to import X509v3 certificates to the TOE's trust store,
- Ability to manage the trusted public keys database,
- Configure approval/denial of AP and vSZ-D to communicate/join the controller
- Configure the security policy for each wireless network, including:
  - Security type
  - Authentication protocol
  - Client credentials to be used for authentication
  - Service Set Identifier (SSID)
  - If the SSID is broadcasted Frequency band set to [2.4 GHz, 5 GHz]

  - Transmit power level
- Define an inventory of authorized APs based on [MAC addresses]
- Define an inventory of authorized EUDs based on MAC addresses
- Define rules for monitoring and alerting on the wireless traffic
- Define authorized SSID(s)
- Define authorized WLAN authentication schemes
- Define authorized WLAN encryption schemes
- disable transmission of data by wireless sensor,
- Define attack signatures,
- Define rules for overwriting previous packet captures,
- Define the amount of time sensor monitors a specific channel

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

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

The TSF data manipulating functions and the corresponding configuration information are identified or referenced throughout this AAR with the requirement to which they apply. Section "Crypto Officer Roles and Responsibilities" **Admin Guide** states, regardless of the component, no administrative action can be performed prior to authentication as an administrator user.

Section "X.509 Certificates" in the **Admin Guide** describes how administrators can generate certificate signing requests, configure certificates on the controller and upload CA certificates to the AP and vSZ-D.

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

No separate testing required.

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

2.6.3.1 NDcPP22e:FMT_MTD.1.1/CryptoKeys

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6 in the **ST** details how each TOE security function including all security management functions are realized for every TOE component. The SZ/vSZ controller component in the TOE manages all TOE components. Section 7 "Requirement Allocation" in the **ST** provides a mapping of the distributed TOE components to the SFRs in this **ST**. This TOE is a distributed TOE consistent with Use Case 3 as defined in the NDcPP22E.

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

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

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

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

Section "Joining vSZ-D to the vSZ Controller" of **Admin Guide** states "After the vDP is joined to the vSZ, all management of the vDP is performed via the vSZ. SSH admin access to the vDP is stopped."

Section "Joining AP to (v)SZ Controller" of **Admin Guide** states the SSH tunnel will be utilized for management communication between AP and controller.

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

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

As demonstrated in FIA_UIA_EXT.1 test 1, no actions other than viewing the warning banner or initiating authentication are available prior to login. The successful management of the crypto keys was performed to successfully configure the trusted channels that were tested for FTP_ITC.1. NDcPP22e:FIA_X509_EXT.3 also specifically demonstrates the creation of a successful certificate request and private key and the importing of the signed certificate along with its trusted root certificate.

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

2.6.4.1 NDcPP22e:FMT_SMF.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

(containing also requirements on Guidance Documentation and Tests)

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

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

For distributed TOEs with the option 'ability to configure the interaction between TOE components' the evaluator shall examine that the ways to configure the interaction between TOE components is detailed in the TSS and

Guidance Documentation. The evaluator shall check that the TOE behaviour observed during testing of the configured SFRs is as described in the TSS and Guidance Documentation.

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

Section 6.6 of the **ST** states the TOE provides the ability to perform administrative functions as an authorized user. All authorized users are granted the Administrator role. Administrators can control the TOE components (Controller (SZ/vSZ), vSZ-D and AP) via the Controller. Only authenticated administrators are allowed access to the TOE to perform administrative functions via WebUI (HTTPS), CLI (SSH) and Local Console. All security functions are available through all interfaces.

Section "Administrating the Controller using CLI Console" of the **AGD** describes how to connect via the local interface. As the local interfaces is distinct no warning is necessary.

Refer to the activities for NDcPP22e:FPT_ITT.1 which address how to configure the interactions between TOE components.

Component Guidance Assurance Activities: See TSS Assurance Activities

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

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

The evaluator tested management functions as part of testing the SFRs identified in the TSS assurance activity above. No separate testing for FMT_SMF.1 is required as none of the management functions in FMT_SMF.1.1 have not already been exercised under any other SFR.

2.6.5 Specification of Management Functions (WLAN Access Systems) (WLANAS10:FMT_SMF.1/AccessSystem)

2.6.5.1 WLANAS10:FMT_SMF.1.1/AccessSystem

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will confirm that the TSS includes which security types (e.g., WPA3), authentication protocol (e.g., SAE), and frequency bands the WLAN AS supports. The evaluator will confirm that the TSS includes how connection attempts from clients that are not operating on an approved security type are handled.

Section 6.6 of **ST** states the TOE supports the following management functions related to WLAN functionality:

- Configure the security policy for each wireless network, including:
  - Security type
  - Authentication protocol
  - Client credentials to be used for authentication
  - Service Set Identifier (SSID)
  - If the SSID is broadcasted Frequency band set to [2.4 GHz, 5 GHz]
  - Transmit power level

Section 6.9 of **ST** states for wireless users, WPA3-Enterprise, WPA2-Enterprise, WPA3-SAE and WPA2-PSK as defined by IEEE 802.11-2020 are used to provide a trusted channel between the TOE and WLAN clients. If the client attempts to use another security type to establish a connection, the authentication attempt will be rejected.

Component Guidance Assurance Activities: The evaluator will confirm that the operational guidance includes instructions for configuring the WLAN AS for each feature listed.

Section "Enabling Trusted Channel Using IEEE 802.11-2012 (WPA2) Standards" in the **Admin Guide** describes how to enable a secure and trusted channel for communication by using IEEE 802.11-2012 (WPA2) standards. This includes setting the security type, authentication protocol, and SSID. Section "Creating a WLAN WPA3 WLAN2/WPA3 Mixed Profile" describes how to perform the same steps using WPA3-SAE.

Section "Creating an AP Zone" includes details on configuring the channels an AP broadcasts on as well as setting a transmit power level.

Component Testing Assurance Activities: Test 1: For each security type specified in the TSS, configure the network to the approved security type and verify that the client can establish a connection. Maintaining the same SSID, change the security type of the client to a non-approved security type and attempt to establish a connection. Verify that the connection was unsuccessful.

Test 2: For each authentication protocol specified in the TSS, configure the network accordingly per the AGD. Verify that the client connection attempt is successful when using the correct client credentials and that the connection is unsuccessful when incorrect authentication credentials are used.

Test 3: Configure the SSID to be broadcasted. Using a network sniffing tool, capture a beacon frame and confirm that the SSID is included. Configure the SSID to be hidden. Using a network sniffing tool, capture a beacon frame and confirm that the SSID is not listed.

Test 4: The evaluator will configure the AS to operate in each of the selected frequency bands and verify using a network sniffing tool.

Test 5: The evaluator will demonstrate that the client can establish a connection to the AS on the default power level. After disconnecting, the power level should be adjusted and then the client should be able to successfully connect to the AS again.

Test 1: For each security type claimed by the TOE, the evaluator first configured the WLAN client to use the approved security type and attempted a connection The evaluator observed these connection attempts were successful. The evaluator then maintained the SSID configured on the TOE, and configured the client for a security type that was not configured/approved on the TOE. The evaluator then caused the client to attempt a connection to the TOE. The evaluator observed that these connection attempts using a non-approved security type were unsuccessful.

Test 2: The evaluator attempted two connections with each authentication protocol (WPA2-PSK, WPA3-SAE, and enterprise for either WPA2 or WPA). The first attempt had the client provide correct authentication credentials, and the connection was successful. The following attempt had the client provide incorrect credentials, and the connection was unsuccessful. The enterprise valid and invalid attempts results were demonstrated as part of WLANAS10:FIA_8021X_EXT.1.

Test 3: The evaluator configured the TOE to broadcast its SSID following Operational Guidance. The evaluator started a sniffing session of wireless traffic, and confirmed that the resulting packet capture included a beacon frame that listed the TOEs SSID. The evaluator then configured the TOE to hide its SSID, and repeated the process. The evaluator confirmed that the resulting packet capture included a beacon frame that did not list the SSID.

Test 4: The evaluator configured the TOE following operational guidance to broadcast only on each of the claimed bands (2.4ghz, 5ghz.) For each band, the evaluator disabled the other two and attempted to connect a client, which was successful. The evaluator then verified that the client connected with the configured band using a command on the TOE.

Test 5: The evaluator first demonstrated a successful connection from a client to the AS using the default power level. The evaluator then disconnected from the AS, then adjusted the power level. With this new power level setting, the evaluator attempted to connect the client again and observed that the connection attempt was successful.

2.6.6 Specification of Management Functions (WIDS) (WIDS10:FMT_SMF.1/WIDS)

2.6.6.1 WIDS10:FMT_SMF.1.1/WIDS

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall review the TSS to verify that it includes information the ability of the TOE to define inventory of authorized APs and EUDs.

The evaluator shall verify that the TSS describes the ability of the TOE to allow authorized administrators to define authorized WLAN authentication schemes.

Section 6.6 of **ST** states the TOE has the following available management functionality related to WIDS:

- Define an inventory of authorized APs based on [MAC addresses]
- Define an inventory of authorized EUDs based on MAC addresses
- Define rules for monitoring and alerting on the wireless traffic
- Define authorized SSID(s)
- Define authorized WLAN authentication schemes
- Define authorized WLAN encryption schemes
- disable transmission of data by wireless sensor,
- Define attack signatures,
- Define rules for overwriting previous packet captures,
- Define the amount of time sensor monitors a specific channel

Component Guidance Assurance Activities: The evaluator shall review the operational guidance for instructions on how to configure and change clasification of APs and EUDs to indicate that they are part of the allowlist.

The evaluator shall review the operational guidance to determine how to configure which SSIDs are permitted on the network.

The evaluator shall examine the operational guidance to verify that it provides instructions on how to define a WLAN authentication scheme as authorized or unauthorized for the purposes of detection.

The evaluator shall examine the operational guidance to verify that it provides instructions on how to define a WLAN encryption scheme as authorized or unauthorized for the purposes of detection.

Section “Allowed Device Profile” of **Admin Guide** defines how to configured a device to be detected as authorized/unauthorized based on the AP/EUDs MAC address. Section “Different Rule types and Classification” contains information about the SSID WIDS rule which can define unauthorized SSIDs to detect, the Unauthorized Auth Scheme rule, and Unauthorized Encryption Scheme. For the unauthorized SSID rule, an SSID is considered authorized only if configured to be used by a managed WLAN. All other SSIDs are detected as unauthorized. The

"Trusted Communication Channels" section of the **Admin Guide** demonstrates how to configure a WLAN with an SSID which in turn determines the authorized SSIDs.

Component Testing Assurance Activities: Test 1: The evaluator shall define an inventory of authorized APs and EUDSs. The ability to detect allowlisted and non-allowlisted APs and EUDs will be tested in FAU_INV_EXT.1 and FAU_SAA.1.

Test 2: The evaluator shall define authorized SSIDs. The ability to detect authorized and unauthorized SSIDs will be tested in FAU_WID_EXT.2.3 and FAU_SAA.1.

Test 3: The evaluator shall configure the TSF with a set of allowed authentication and encryption schemes. The ability to detect violation of this policy will be tested in FAU_SAA.1.

Test 4: (conditional): If 'Define the amount of time sensor monitors a specific frequency or channel' is selected:

Step 1: Deploy an allowlisted AP and connect it to the protected wired infrastructure via wire.

Step 2: Confirm that the TSF can observe and capture traffic and events generated by the AP.

Step 3: Verify that the TSF can be configured to capture traffic on a specific channel for specific interval of time, and assign a specified frequency and time interval.

Step 4: Confirm that the TSF remains on the frequency and channel for the time period specified.

Test 1: The evaluator demonstrated the ability to define an inventory of authorized APs ad EUDs in WIDS10:FAU_INV_EXT.1. The ability to detect allowlisted and non-allowlisted APs and EUDs will be tested in FAU_INV_EXT.1 and FAU_SAA.1.

Test 2: The ability to detect authorized and unauthorized SSIDs was tested in WIDS10:FAU_SAA.1 tests 11 and 12. These tests address how SSIDs are defined as unauthorized or authorized.

Test 3: The ability to detect unauthorized authentication and encryption schemes was tested in WIDS10:FAU_SAA.1 tests 20 and 21. These tests note how authorized and unauthorized encryption and authentication schemes are defined.

Test 4: The evaluator deployed an allowlisted AP and connected it to the protected wire infrastructure via wire. The evaluator confirmed that the TOE could observe and capture traffic and events generated by the AP. The evaluator specified the time interval for the TOE by selecting low scanning frequency and the channels specified for monitoring were selected as well. The evaluator observed no interruption in behavior and the TOE monitored the channel correctly as expected.

2.6.7 Restrictions on Security Roles (NDcPP22e:FMT_SMR.2)

2.6.7.1 NDcPP22e:FMT_SMR.2.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.7.2 NDcPP22e:FMT_SMR.2.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.6.7.3 NDcPP22e:FMT_SMR.2.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.6 of the **ST** states the TOE provides the ability to perform administrative functions as an authorized user. All authorized users are granted the Administrator role. Administrators can control the TOE components (Controller (SZ/vSZ), vSZ-D and AP) via the Controller. Only authenticated administrators are allowed access to the TOE to perform administrative functions via WebUI (HTTPS), CLI (SSH) and Local Console. All security functions are available through all interfaces.

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

Section "Administrating the Controller" in the **Admin Guide** describes how the administrator can log in to and administer the TOE both locally via the local console and remotely via an SSH CLI session or an HTTPS Web UI session. There is no specific configuration needed to access the SSH and WebUI session, it is enabled by default.

Component Testing Assurance Activities: In the course of performing the testing activities for the evaluation, the evaluator shall use all supported interfaces, although it is not necessary to repeat each test involving an

administrative action with each interface. The evaluator shall ensure, however, that each supported method of administering the TOE that conforms to the requirements of this cPP be tested; for instance, if the TOE can be administered through a local hardware interface; SSH; and TLS/HTTPS; then all three methods of administration must be exercised during the evaluation team's test activities.

The different administrative interfaces were used for testing over the course of the evaluation. Access to each of the TOE interfaces was demonstrated in FIA_UIA_EXT.1 test 1.

2.6.8 No Administration from Client (WLANAS10:FMT_SMR_EXT.1)

2.6.8.1 WLANAS10:FMT_SMR_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: The evaluator will review the operational guidance 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 evaluator will confirm that the TOE does not permit remote administration from a wireless client by default.

Section “Administrating the Controller” in the **Admin Guide** describes how the administrator can log in to and administer the TOE both locally via the local console and remotely via an SSH CLI session or an HTTPS Web UI session. There is no specific configuration needed to access the SSH and WebUI session, it is enabled by default. Wireless clients connecting to a managed AP are communicating on the TOE's data channel and not via the management interface and therefore, cannot perform remote administration of the SZ.

Component Testing Assurance Activities: The evaluator will demonstrate that after configuring the TOE for first use from the operational guidance, it is possible to establish an administrative session with the TOE on the 'wired' portion of the device. They will then demonstrate that an identically configured wireless client that can successfully connect to the TOE cannot be used to perform administration.

Refer to NDcPP22e:FIA_UIA_EXT.1-t1 where the evaluator demonstrated that it is possible to establish an administrative session with the TOE on the 'wired' portion of the device. The evaluator connected to the TOE from a wireless client and attempted to access the web UI for the SZ144, SZ300 and VSZ-H, and found that the connection failed each time.

2.7 Protection of the TSF (FPT)

2.7.1 Protection of Administrator Passwords (NDcPP22e:FPT_APW_EXT.1)

2.7.1.1 NDcPP22e:FPT_APW_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.1.2 NDcPP22e:FPT_APW_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.7 in the **ST** states the TOE is designed to protect critical security parameters. The TOE provides no mechanism to read or disclose passwords stored in TOE. Passwords are stored in non-plaintext form and are obscured. Passwords are encrypted in storage.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.2 Failure with Preservation of Secure State (WLANAS10:FPT_FLS.1)

2.7.2.1 WLANAS10:FPT_FLS.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will examine the TSS to determine that the TOE's implementation of the fail secure functionality is documented. The evaluator will examine the TSS to ensure that it describes all failure conditions and how a secure state is preserved if any of these failures occur. The evaluator will ensure that the definition of a secure state is suitable to ensure the continued protection of any key material and user data.

Section 6.7 in the **ST** states that when the TOE boots up, each component of the TOE performs POST (Power On Self-Test) for all cryptographic modules. The tests include:

- AES Known Answer Test
- SHA Known Answer Test
- HMAC Known Answer Test
- CCM Known Answer Test
- GCM Known Answer Test
- DRBG Known Answer Test
- Software Integrity test (RSA with SHA384)
- CMAC Known Answer Test
- RSA Known Answer Test
- ECDSA Pairwise Consistency Test
- ECC CDH shared secret computation

When any one of the known answer tests fails, the system will enter the quarantine state. Upon critical failure, the TOE component either goes into a quarantine state and can be recovered only by a security administrator or it reboots and disables access to its interfaces therefore restricting access to key material and user data.

Component Guidance Assurance Activities: The evaluator will examine the operational guidance to verify that it describes applicable recovery instructions for each TSF failure state.

Section “Quarantine state” in **Admin Guide** states when a power up self test fails, the system (controller or dataplane) moves to the quarantine state. In the quarantine state, only the CO (admin) can log into the CLI and recover the system. In the quarantine state, limited CLI commands are available for system recovery. All communication toward external nodes is disabled, and network interfaces are down.

Section “Quarantine state for AP” in **Admin Guide** states AP goes into the quarantine state in either of the following scenarios:

- when AP is zeroized
- when an AP self-test has failed due to an error in the firmware.

In zeroized APs, the crypto officer is unable to access the AP CLI. The only way to recover the CO login is through a hard reset. This allows the CO to log into the AP CLI; however, zeroization causes the AP to lose the web, user, and SSH certifications and keys permanently. However, in APs that fail the self-test, network connectivity is down and

even hard reset cannot recover the AP; it has to be sent back to the factory. The failure of the AP self-test can only be determined by physically examining the device.

Component Testing Assurance Activities: For each failure mode specified in the ST, the evaluator will ensure that the TOE attains a secure state (e.g., shutdown) after initiating each failure mode type.

The evaluator used a special test build on each TOE type so that known answer test failures can be initialized at boot. The evaluator viewed that the TOE entered a secure state once the test failure occurred.

2.7.3 Basic internal TSF data transfer protection - per TD0639 (NDcPP22e:FPT_ITT.1)

2.7.3.1 NDcPP22e:FPT_ITT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: If the TOE is not a distributed TOE, then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall examine the TSS to determine that, for all communications between components of a distributed TOE, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator shall also confirm that all protocols listed in the TSS for these inter-component communications are specified and included in the requirements in the ST.

Section 6.7 in the **ST** states that the TOE has 3 components, a controller, an AP, and in virtual deployments, a vSZ-D. All management communication between the controller and AP and between the controller and vSZ-D are secured via SSH. Data traffic between AP and vSZ-D or controller is secured via IPsec. The distributed communications are encrypted the same regardless of the AP being in a wired or mesh configuration. Section 6.3 describes the protocols and their options in the evaluated configuration. The descriptions of the protocols found in the TSS are described in assurance activities for the corresponding SFRs found previously in this AAR.

Component Guidance Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall confirm that the guidance documentation contains instructions for establishing the relevant allowed communication channels and protocols between each pair of authorized TOE components, and that it contains recovery instructions should a connection be unintentionally broken.

Section “AP Models that Support FIPS Mode” in the **Admin Guide** indicates that while TLS is used to perform the initial discovery of the SZ controller, this TLS is not claimed or tested in the evaluation due to limited certificate verification capabilities during the registration of the TOE components. The TOE requires the use of a dedicated channel for the AP and vSZ-D to register with the Controller. The administrator must perform the registration of TOE components in a controlled environment in which there is a segregated network with only TOE components present.

Section “Joining vSZ-D to the vSZ Controller” in the **Admin Guide** indicates the IP address of the vSZ controller is configured on the vSZ-D which then initiates the attempt to connect to the controller. The administrator then selects and approves the vSZ-D via the controller’s Web UI. Once the registration process is approved, an SSH connection is automatically established and all communication between the vSZ-D and the vSZ Controller is over SSH. If the connection between vSZ-D and vSZ is broken then it resumes back automatically and no manual intervention is required.

Section “Joining AP to the (v)SZ Controller” in the **Admin Guide** describes the steps for joining of the AP to the SZ Controller. The IP address of the SZ controller is configured on the AP which then initiates the attempt to connect to the controller. The administrator then selects and approves the AP via the controller’s Web UI. Once the registration process is approved, an SSH connection is automatically established between the AP and the SZ Controller. The connection with the AP can similarly be disabled if the administrator selects the AP and then clicks the ‘Delete’ button. If the connection is broken it will be resumed/reattempted without any user intervention.

Section “Configuring Regular Mesh” in the **Admin Guide** describes how to setup a wireless mesh configuration with a Root AP and a Mesh AP. This configuration does not change the distributed TOE channel’s encryption.

Section “Configuring Ruckus GRE and IPsec in WLAN-Concept” in the **Admin Guide** indicates that the Ruckus GRE and IPsec is the IPsec tunnel between the AP and the SZ or vSZ-D and describes how to enable IPsec between the AP and SZ or vSZ-D. The IPSec connection between AP and vSZ-D is recovered automatically and manual intervention is not required.

Component Testing Assurance Activities: If the TOE is not a distributed TOE then no evaluator action is necessary. For a distributed TOE the evaluator carries out the activities below.

The evaluator shall perform the following tests:

a) Test 1: The evaluator shall ensure that communications using each protocol between each pair of authorized TOE components is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

b) Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

c) Test3: Objective: The objective of this test is to ensure that the TOE reacts appropriately to any connection outage or interruption of the route between distributed components.

The evaluator shall ensure that, for each different pair of non-equivalent component types, the connection is physically interrupted for the following durations: i) a duration that exceeds the TOE's application layer timeout setting, ii) a duration that is shorter than the application layer timeout but is of sufficient length to interrupt the network link layer.

The evaluator shall ensure that when physical connectivity is restored, either communications are appropriately protected, or the secure channel is terminated and the registration process (as described in the FTP_TRP.1/Join) re-initiated, with the TOE generating adequate warnings to alert the Security Administrator.

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

Further assurance activities are associated with the specific protocols.

Test 1 - The SSH communication between the AP and SZ Controller, or in a virtual instance, the vSZ-D and SZ controller, is enabled by default after the registration process completes. The registration process was demonstrated as part of FCO_CPC_EXT.1 and FTP_TRP.1/Join. The IPsec for the WLAN data channel is not enabled by default, so the evaluator demonstrated the setup of the IPsec for the WLAN data tunnel. The successful connections for both the SSH channel and the IPsec channel are demonstrated in test 3 below. The evaluator did demonstrate the setup for a mesh configuration.

Test 2 – This test was demonstrated as part of test 3 below however to ensure that a mesh configuration did not affect the encrypted distributed TOE channels, the evaluator captured the traffic between the SZ controller and the wireless mesh AP and viewed the management traffic still used SSH and the data traffic still used IPsec.

Test 3 – SSH: The evaluator registered the AP to the SZ controller so that the SSH ITT communication was enabled. The evaluator then physically disrupted the SSH connections by temporarily unplugging a cable between two in-line switches. The evaluator observed that the channels remained secure during and after the disruption. The evaluator physically disconnected a switch between the AP and SZ controller for approximately 1 minute. The TOE continued to use the existing connection. The evaluator physically disconnected a switch between the AP and SZ controller for approximately 10 minutes. The TOE then renegotiated a new connection.

IPsec: The evaluator registered the AP to the SZ controller and then protected the connection with IPsec. The evaluator physically disrupted the IPsec connection by temporarily unplugging a cable between two in-line switches. The evaluator observed that the channel remained secure during and after the disruption. The evaluator physically disconnected a switch between the AP and SZ controller for approximately 30 seconds. The TOE continued to use the existing connection. The evaluator physically disconnected a switch between the AP and SZ controller for approximately 10 minutes. The TOE then renegotiated a new connection. This test was repeated for the connection between an AP and the vSZ-D.

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

2.7.4.1 NDcPP22e:FPT_SKP_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.7 in the **ST** states that the TOE is designed to protect critical security parameters. The TOE provides no mechanism to read or disclose private keys, preshared keys, passwords or symmetric keys stored in TOE. Keys are stored in internal storage. Passwords are stored in non-plaintext form and are obscured. Passwords are encrypted in storage.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.7.5 Reliable Time Stamps - per TD0632 (NDcPP22e:FPT_STM_EXT.1)

2.7.5.1 NDcPP22e:FPT_STM_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.5.2 NDcPP22e:FPT_STM_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.7 in the **ST** states that the TOE provides reliable timestamp by synchronizing time via NTP securely. The timestamps are used in audit records, certificate validation, session monitoring activity and Wireless client access management.

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

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

Section “Configuring System Time” in **Admin Guide** describes how the controller uses an external Network Time Protocol (NTP) server to synchronize the times across cluster nodes and managed access points.

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

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

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

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

c) Test 3: [conditional] If the TOE obtains time from the underlying VS, the evaluator shall record the time on the TOE, modify the time on the underlying VS, and verify the modified time is reflected by the TOE. If there is a delay

between the setting the time on the VS and when the time is reflected on the TOE, the evaluator shall ensure this delay is consistent with the TSS and Guidance.

Test 1 – Not applicable. The TOE does not support direct setting of the time.

Test 2 - The evaluator configured the NTP client on the controller devices and observed that each device successfully synchronized their time with the time on the NTP server.

Test 3- Not applicable. The TOE does not obtain time from the underlying VS.

2.7.6 TSF TESTING (NDCPP22E:FPT_TST_EXT.1)

2.7.6.1 NDCPP22E:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

Section 6.7 in the **ST** states that when the TOE boots up, each component of the TOE performs POST (Power On Self-Test) for all cryptographic modules. The tests include:

- AES Known Answer Test
- SHA Known Answer Test
- HMAC Known Answer Test
- CCM Known Answer Test
- GCM Known Answer Test
- DRBG Known Answer Test
- Software Integrity test (RSA with SHA384)

- CMAC Known Answer Test
- RSA Known Answer Test
- ECDSA Pairwise Consistency Test
- ECC CDH shared secret computation

When any one of the known answer tests fails, the system will enter the quarantine state. Upon critical failure, the TOE component either goes into a quarantine state and can be recovered only by a security administrator or it reboots and disables access to its interfaces.

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

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

Section “Quarantine state” in **Admin Guide** states when a power up self test fails, the system (controller or dataplane) moves to the quarantine state. In the quarantine state, only the CO (admin) can log into the CLI and recover the system. In the quarantine state, limited CLI commands are available for system recovery. All communication toward external nodes is disabled, and network interfaces are down.

Section “Quarantine state for AP” in **Admin Guide** states AP goes into the quarantine state in either of the following scenarios:

- when AP is zeroized
- when an AP self-test has failed due to an error in the firmware.

In zeroized APs, the crypto officer is unable to access the AP CLI. The only way to recover the CO login is through a hard reset. This allows the CO to log into the AP CLI; however, zeroization causes the AP to lose the web, user, and SSH certifications and keys permanently. However, in APs that fail the self-test, network connectivity is down and even hard reset cannot recover the AP; it has to be sent back to the factory. The failure of the AP self-test can only be determined by physically examining the device.

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

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

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

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

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

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

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

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

Test 1 – The evaluator rebooted the TOE and confirmed via boot transcript and event logs on all four devices that software/firmware integrity tests and cryptographic algorithm tests are performed.

2.7.7 TSF Testing (WLANAS10:FPT_TST_EXT.1)

2.7.7.1 WLANAS10:FPT_TST_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will examine the TSS to ensure that it describes how to verify the integrity of stored TSF executable code when it is loaded for execution, which includes the generation and protection of the 'check value' used to ensure integrity as well as the verification step. This description will also cover the digital signature service used in performing these functions. The evaluator also checks the operational guidance to ensure that any actions required by the administrator to initialize or operate this functionality are present.

The evaluator will also ensure that the TSS or operational guidance describes the actions that take place for successful and unsuccessful execution of the integrity test.

Section 6.7 in the **ST** states that when the TOE boots up, each component of the TOE performs POST (Power On Self-Test) for all cryptographic modules. The tests include:

- AES Known Answer Test

- SHA Known Answer Test
- HMAC Known Answer Test
- CCM Known Answer Test
- GCM Known Answer Test
- DRBG Known Answer Test
- Software Integrity test (RSA with SHA384)
- CMAC Known Answer Test
- RSA Known Answer Test
- ECDSA Pairwise Consistency Test
- ECC CDH shared secret computation

When any one of the known answer tests fails, the system will enter the quarantine state. Upon critical failure, the TOE component either goes into a quarantine state and can be recovered only by a security administrator or it reboots and disables access to its interfaces.

Component Guidance Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will ensure that the TSS or operational guidance describes the actions that take place for successful and unsuccessful execution of the integrity test.

Section “Quarantine state” in **Admin Guide** states when a power up self test fails, the system (controller or dataplane) moves to the quarantine state. In the quarantine state, only the CO (admin) can log into the CLI and recover the system. In the quarantine state, limited CLI commands are available for system recovery. All communication toward external nodes is disabled, and network interfaces are down.

Section “Quarantine state for AP” in **Admin Guide** states AP goes into the quarantine state in either of the following scenarios:

- when AP is zeroized
- when an AP self-test has failed due to an error in the firmware.

In zeroized APs, the crypto officer is unable to access the AP CLI. The only way to recover the CO login is through a hard reset. This allows the CO to log into the AP CLI; however, zeroization causes the AP to lose the web, user, and SSH certifications and keys permanently. However, in APs that fail the self-test, network connectivity is down and even hard reset cannot recover the AP; it has to be sent back to the factory. The failure of the AP self-test can only be determined by physically examining the device.

Component Testing Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will perform the following tests:

Test 1: Following the operational guidance, the evaluator will initialize the integrity protection system. The evaluator will perform actions to cause TSF software to load and observe that the integrity mechanism does not flag any executables as containing integrity errors.

Test 2: The evaluator will modify the TSF executable and cause that executable to be loaded by the TSF. The evaluator will observe that an integrity violation is triggered (care must be taken so that the integrity violation is determined to be the cause of the failure to load the module and not the fact that the module was modified so that it was rendered unable to run because its format was corrupt).

Test 1: This test was successfully demonstrated in NDcPP22e:FPT_TST_EXT.1.

Test 2: Integrity checks on images are checked at time of upgrade. Failed integrity images are rejected. This was tested as part of NDcPP22e:FPT_TUD_EXT.1.

2.7.8 Trusted update (NDcPP22e:FPT_TUD_EXT.1)

2.7.8.1 NDcPP22e:FPT_TUD_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.8.2 NDcPP22e:FPT_TUD_EXT.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.7.8.3 NDcPP22e:FPT_TUD_EXT.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

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

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

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

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

Section 6.7 in the **ST** states that the TOE provides the ability to query a currently running firmware version via both the WebUI and the CLI after logging in. The TOE also provides the ability to obtain and apply a new software version via the Web UI. The new software update is first uploaded and then activated later when the administrator selects the upgrade option. The Controller in turn will upgrade the APs and vSZ-D once it has applied the software on itself securely via SSH. All components of the TOE are updated via the controller. All controller, AP and vSZ-D updates are verified via digital signatures. Software installation will fail if the signature verification fails. All software updates are hosted on Ruckus support portal and can be downloaded securely via HTTPS after authenticating to the server.

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

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

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

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

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

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

Section “Upgrading the Software” in the **Admin Guide** provides instructions for updating the software.

Section “Upgrading (v)SZ Software” in the **Admin Guide** provides instructions for updating the software and for querying the currently active version. This includes downloading the update image from the Ruckus Customer release web site and then uploading it to the Controller. The web interface lists the active and inactive upgrade history and includes a screenshot of where the inactive upgrade can be seen. Once uploaded, delayed activation/upgrade can be initiated. The upgrade package contains the upgrade software/firmware, signatures and the certificates of the signature signers. When the upgrade package is uploaded to the controller, the controller will validate the certificate chain first. If the certificate of the signature signer passes the chain validation, the controller then verifies the signatures of the upgrade software/firmware. When the upgrade package signature signer certificate chain validation error or signature verification error occur, the GUI shows a package decryption error. In such cases, a valid upgrade package should be used to continue system upgrading.

The “Upgrading the AP Software” section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to AP Software are addressed via AP firmware associated with a firmware version which is bundled as part of the (v) SZ Software upgrade image. This section further provides instructions for manually upgrading the AP firmware version. The Change AP Firmware option displays the available firmware updates as seen in the screenshots provided. The Firmware software contains the upgrade software, Signatures and certificates of the signature signers. When the Firmware is pushed to the AP from the (v)SZ, the AP will validate the Certificate Chain first once the Chain validation goes through then AP validates the Signatures of the upgrade firmware. If any of this validation fails first, the upgrade will fail and the corresponding status will be shown on UI and detailed info can be viewed through logs.

The “Upgrading the vSZ-D Software” section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to vSZ-D software are addressed through vSZ-D Patch and provides instructions for upgrading the vSZ-D software. The provided screenshot shows how the inactive vSZ-D patch is viewed. The upgrade patch contains the upgrade software/firmware, signatures and the certificates of the signature signers. When the upgrade package is uploaded to the (v)SZ, (v)SZ will validate the certificate chain first. If the certificate of the signature signer passes the chain validation, the (v)SZ then verifies the signatures of the upgrade software/firmware. When the upgrade package signature signer certificate chain validation error or signature verification error occur, the GUI shows a package decryption error. In such cases, a valid upgrade package should be used to continue system upgrading.

Section “Self-Service Resources” indicates that software downloads and release notes can be obtained via the Ruckus Support Portal at https://support.ruckuswireless.com.

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

a) Test 1: The evaluator performs the version verification activity to determine the current version of the product. If a trusted update can be installed on the TOE with a delayed activation, the evaluator shall also query the most recently installed version (for this test the TOE shall be in a state where these two versions match). The evaluator obtains a legitimate update using procedures described in the guidance documentation and verifies that it is successfully installed on the TOE. For some TOEs loading the update onto the TOE and activation of the update are separate steps ('activation' could be performed e.g. by a distinct activation step or by rebooting the device). In that case the evaluator verifies after loading the update onto the TOE but before activation of the update that the current version of the product did not change but the most recently installed version has changed to the new product version. After the update, the evaluator performs the version verification activity again to verify the version correctly corresponds to that of the update and that current version of the product and most recently installed version match again.

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

The evaluator verifies that the TOE rejects all of the illegitimate updates. The evaluator performs this test using all of the following forms of illegitimate updates:

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

2) An image that has not been signed

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

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

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

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

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

to install the update on the TOE (in spite of the unsuccessful hash verification) and confirms that this fails. Depending on the implementation of the TOE, the TOE might not allow to even attempt updating the TOE after the verification of the hash value fails. In that case the verification that the hash comparison fails is regarded as sufficient verification of the correct behaviour of the TOE.

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

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

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

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

Test 1- On each of the TOE components, the evaluator verified the version on each device prior to performing the update, performed the update and then verified that the device version was successfully updated.

Test 2 - Since all components of the TOE are updated via the controller, the invalid updates were tested by attempting to upload them to the SZ controller and verifying the images were rejected. The evaluator modified the update image file, and obtained an unsigned image and an image with an invalid signature from the vendor. The evaluator then noted the version information and attempted to install each invalid image and found that the process failed, and that the version information did not change.

Test 3 – Not applicable. The TOE does not support published hash.

2.8 TOE ACCESS (FTA)

2.8.1 TSF-INITIATED TERMINATION (NDCPP22E:FTA_SSL.3)

2.8.1.1 NDCPP22E:FTA_SSL.3.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.8 of the **ST** states The TOE security administrator can configure the session inactivity timeout which terminates the sessions of administrators once the session inactivity timeout is reached for both local and remote interactive sessions.

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

Section "Terminating Sessions" section in the **Admin Guide** describes how to configure the controller to terminate a remote interactive session after it has exceeded the session timeout value configured by the security administrator. The configuration also gets applied to the virtual dataplane and access points. The session idle timeout value can be set between 1 to 1440 minutes. For the CLI session the default session timeout is 30 minutes. For the GUI session the default session idle timeout is 15 minutes.

Component Testing Assurance Activities: For each method of remote administration, the evaluator shall perform the following test:

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

For both the SSH CLI and the Web UI, the evaluator configured a 1 minute timeout and then observed that the session timed out at approximately 1 minute. This test was then repeated on both TOE administrative interfaces using time value of 5 minutes.

2.8.2 User-initiated Termination (NDcPP22e:FTA_SSL.4)

2.8.2.1 NDcPP22e:FTA_SSL.4.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

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

Section 6.8 of the **ST** states Administrators also have the ability to terminate their own session by clicking logout on the WebUI and typing exit on the CLI.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation states how to terminate a local or remote interactive session.

Section "Terminating Administrator Sessions" provides instructions for logging out of the Web UI by selecting the 'Log off' option and logging out of an SSH session or the local console using the 'exit' command.

Component Testing Assurance Activities: For each method of remote administration, the evaluator shall perform the following tests:

a) Test 1: The evaluator initiates an interactive local session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

b) Test 2: The evaluator initiates an interactive remote session with the TOE. The evaluator then follows the guidance documentation to exit or log off the session and observes that the session has been terminated.

Test 1 – This test was performed during the testing of NDcPP21:FIA_UIA_EXT.1, test 1. The evaluator logged out of a local interactive session and observed that the session was terminated.

Test 2 - This test was performed during the testing of NDcPP21:FIA_UIA_EXT.1, test 1. The evaluator logged out of the remote interactive sessions via both the Web UI and SSH CLI and observed that the session was terminated.

2.8.3 TSF-initiated Session Locking (NDcPP22e:FTA_SSL_EXT.1)

2.8.3.1 NDcPP22e:FTA_SSL_EXT.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that it details whether local administrative session locking or termination is supported and the related inactivity time period settings.

Section 6.8 of the **ST** states The TOE security administrator can configure the session inactivity timeout which terminates the sessions of administrators once the session inactivity timeout is reached for both local and remote interactive sessions.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation states whether local administrative session locking or termination is supported and instructions for configuring the inactivity time period.

Section "Terminating Sessions" in **Admin Guide** describes how to configure the controller to terminate a remote interactive session after it has exceeded the session timeout value configured by the security administrator. This configuration also gets applied to the virtual dataplane and access points. The local CLI timeout of the controller is set through the CLI. The session idle timeout value can be set between 1 to 1440 minutes. For the CLI session the default session timeout is 30 minutes.

Component Testing Assurance Activities: The evaluator shall perform the following test:

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

The evaluator set the console inactivity time period to 1 minute and verified that the devices timed out after 1 minute as expected, and then repeated this with an inactivity time period of 5 minutes.

2.8.4 Default TOE Access Banners (NDcPP22e:FTA_TAB.1)

2.8.4.1 NDcPP22e:FTA_TAB.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall check the TSS to ensure that it details each administrative method of access (local and remote) available to the Security Administrator (e.g., serial port, SSH, HTTPS). The evaluator shall check the TSS to ensure that all administrative methods of access available to the Security Administrator are listed and that the TSS states that the TOE is displaying an advisory notice and a consent warning message for each administrative method of access. The advisory notice and the consent warning message might be different for different administrative methods of access, and might be configured during initial configuration (e.g. via configuration file).

Section 6.5 in the **ST** states that the TOE requires users to be identified and authenticated before they can use functions mediated by the TOE, except to display a message of the day banner without identification or authentication. The TOE requires all administrators of the system to login via credentials (username & password)

before granting access to the system. The Controller provides a local console as well as remote administration of the system via secure communication channel (WebGUI via HTTPS and CLI via SSH).

Section 6.8 in the **ST** states the TOE provides the ability for administrators to configure a login banner which will be visible to users when they try to access the controller prior to log in via WebUI or CLI.

Component Guidance Assurance Activities: The evaluator shall check the guidance documentation to ensure that it describes how to configure the banner message.

Section "Setting Up the Login Banner" in **Admin Guide** describes how an administrator can customize the message that appears in the login banner of the controller web interface.

Component Testing Assurance Activities: The evaluator shall also perform the following test:

a) Test 1: The evaluator follows the guidance documentation to configure a notice and consent warning message. The evaluator shall then, for each method of access specified in the TSS, establish a session with the TOE. The evaluator shall verify that the notice and consent warning message is displayed in each instance.

The evaluator followed the guidance documentation to set the login banner. Refer to NDcPP21:FIA_UIA_EXT.1-t1 where the login banner was displayed at login for the console (local and SSH) and Web UI.

2.8.5 TOE Session Establishment - per TD0679 (WLANAS10:FTA_TSE.1)

2.8.5.1 WLANAS10:FTA_TSE.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that all of the attributes on which a client session can be denied are specifically defined.

Section 6.8 in the **ST** states that the TOE provides ability to deny establishment of a wireless client session based on TOE interface, time, and day.

Component Guidance Assurance Activities: The evaluator shall examine the operational guidance to determine that it contains guidance for configuring each of the attributes identified in the TSS.

Section "Configuring the WLAN Scheduler" in **Admin Guide** states an administrator can control a client's access to the network by providing a time schedule within which the device can access the network. By configuring the WLAN scheduler, the controller can deny establishment of a wireless client session based on wlan, time, day etc.

The controller can also control client access to the network by providing a time schedule within which the device can access the network. When the WLAN scheduler is disabled, SSID broadcasts is disabled and therefore, client connection is lost including all clients that were earlier connected when the scheduler was enabled.

Component Testing Assurance Activities: The evaluator shall also perform the following test for each attribute:

Test 1: The evaluator successfully establishes a client session with a wireless client. The evaluator then follows the operational guidance to configure the system so that that client's access is denied based on a specific value of the attribute. The evaluator shall then attempt to establish a session in contravention to the attribute setting (for instance, the client is denied WLAN access based upon the TOE interface (e.g. WLAN access point) it is connecting to or the client is denied access based upon the time-of-day or day-of-week it is attempting connection on). The evaluator shall observe that the access attempt fails.

The evaluator successfully established a client session with a wireless client. The evaluator then configured the system so that that client's access is denied at all times except for 1am, and found that it was no longer possible to log into the SZ controller. The evaluator then configured the system so that that client's access is denied on all days except for Sunday, and found that it was no longer possible to log into the SZ controller. The evaluator then configured the system so that the AP, which was the only access point managed by the SZ controller, is no longer linked with the SZ controller, and found that it was no longer possible to log into the SZ controller.

2.9 Trusted path/channels (FTP)

2.9.1 Inter-TSF trusted channel - per TD0639 (NDcPP22e:FTP_ITC.1)

2.9.1.1 NDcPP22e:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.1.2 NDcPP22e:FTP_ITC.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.1.3 NDcPP22E:FTP_ITC.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each secure communication mechanism is identified in terms of the allowed protocols for that IT entity, whether the TOE acts as a server or a client, and the method of assured identification of the non-TSF endpoint. The evaluator shall also confirm that all secure communication mechanisms are described in sufficient detail to allow the evaluator to match them to the cryptographic protocol Security Functional Requirements listed in the ST.

Section 6.9 in the **ST** states that the TOE provides secure communication channels for external communications and remote administration. The controller uses secure channel to communicate to external components.

- Audit server via IPSec (TOE can acts as initiator)
- RADIUS via RADSEC (TOE acts as TLS client)
- Wireless client connecting to TOE AP via WPA3 or WPA2

The Audit server via IPsec communication has been addressed further in the assurance activities for NDcPP22E:FCS_IPSEC_EXT.1. Specifically the activities for NDcPP22E:FCS_IPSEC_EXT.1.13 address the TOE properly identifying and authenticating the IPsec peer using PSKs or certificates.

Radius via RADSEC communication has been addressed in the assurance activities for WLANAS10:FCS_RADSEC_EXT.1. This addresses how the TOE is identified and authenticated using X509 certificates with reference identifiers.

Wireless client via WPA2 or WPA3 communication has been addressed in the assurance activities for WLANAS10:FCS_CKM.1/WPA. This describes how radius is leveraged to identify and authenticate the wireless clients.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation contains instructions for establishing the allowed protocols with each authorized IT entity, and that it contains recovery instructions should a connection be unintentionally broken.

Sections “Configuring System IPsec using Preshared Key” and “Configuring System IPsec using Certificates” in **Admin Guide** provide the instructions for enabling IPsec from the controller to external systems for audit data. Section “System IPsec” in the **Admin Guide** describes the IPsec configuration for communication between the SZ controller and an external syslog server. It states that if the connection between the SZ controller and the IPsec gateway is broken after the IKE rekey timeout period, the IPsec tunnel will go down and a system event will be

triggered to notify users. If the connection is broken before the IKE rekey timeout, the system IPsec sends a re-transmission request to the gateway every 10 seconds until the IKE rekey timeout or 360 re-transmission attempts.

Section “Configuring RadSec” in the **Admin Guide** provides instructions for configuring communication to a remote AAA server using Radius over TLS. Section “RadSec (RADIUS over TLS)” in the **Admin Guide** states that the connection between the SZ controller and Radsec Server will last for a maximum of 30 seconds. As soon as the SZ receives a new authentication request, it will initiate a TLS handshake with the Radsec Server. If the network is down or the RadSec server process itself is down, then the wireless client authentication will fail. If the connection is broken, it will resume by default when the next radius message is received from the client.

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

The evaluator shall perform the following tests:

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

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

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

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

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

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

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

Further assurance activities are associated with the specific protocols.

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

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

Test 1 - The evaluator configured the TOE for external authentication using RADIUS over TLS and applied the RadSec configuration to the WLAN. The evaluator also configured the TOE for syslog over IPsec. The successful connections for both RadSec and syslog IPsec were demonstrated in test 4.

Test 2 - This test case was demonstrated as part of test 4.

Test 3 - This test case was demonstrated as part of test 4.

Test 4 – IPsec: The evaluator configured the syslog server and the SZ controller so that a IPsec protected connection should be accepted. The evaluator then physically disrupted the connection by temporarily unplugging a cable between two in-line switches. The evaluator observed that the channel remained secure during and after the disruption. The evaluator then physically disconnected a switch between the syslog server and the SZ controller for approximately 30 seconds. The TOE continued to use the existing IPsec connections. Lastly, the evaluator physically disconnected a switch between the syslog server and the SZ controller for approximately 5 minutes during that period. The TOE then renegotiated new IPsec connections.

Radsec: The evaluator configured the RADIUS server and SZ controller device so that a TLS protected connection should be accepted. The evaluator then physically disrupted the connection by temporarily unplugging a cable between two in-line switches. The evaluator observed that the channel remained secure during and after the disruption. Since the RADIUS TLS connection is atomic, the evaluator used the channel both before, during and after the disruption. The evaluator then physically disconnected a switch between the RADIUS server and the SZ controller for approximately 30 seconds. The TOE continued to use the existing TLS connections. Lastly, the evaluator physically disconnected a switch between the RADIUS server and the SZ controller for approximately 5 minutes. The TOE then renegotiated new TLS connections.

2.9.2 INTER-TSF TRUSTED CHANNEL (WIDS10:FTP_ITC.1)

2.9.2.1 WIDS10:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: None Defined

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.3 Inter-TSF Trusted Channel (WLANAS10:FTP_ITC.1)

2.9.3.1 WLANAS10:FTP_ITC.1.1

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.3.2 WLANAS10:FTP_ITC.1.2

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.3.3 WLANAS10:FTP_ITC.1.3

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will examine the TSS to determine that, for all communications with authorized IT entities identified in the requirement, each communications mechanism is identified in terms of the allowed protocols for that IT entity. The evaluator will also confirm that all protocols listed in the TSS are specified and included in the requirements in the ST.

Section 6.9 in the **ST** states that the TOE provides secure communication channels for external communications and remote administration. The controller uses secure channel to communicate to external components.

- Audit server via IPSec (TOE can acts as initiator)
- RADIUS via RADSEC (TOE acts as TLS client)
- Wireless client connecting to TOE AP via WPA3 or WPA2

The Audit server via IPsec communication has been addressed further in the assurance activities for NDcPP22E:FCS_IPSEC_EXT.1. Specifically the activities for NDcPP22E:FCS_IPSEC_EXT.1.13 address the TOE properly identifying and authenticating the IPsec peer using PSKs or certificates.

Radius via RADSEC communication has been addressed in the assurance activities for WLANAS10:FCS_RADSEC_EXT.1. This addresses how the TOE is identified and authenticated using X509 certificates with reference identifiers.

Wireless client via WPA2 or WPA3 communication has been addressed in the assurance activities for WLANAS10:FCS_CKM.1/WPA. This describes how radius is leveraged to identify and authenticate the wireless clients.

Component Guidance Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will 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.

Sections “Configuring System IPsec using Preshared Key” and “Configuring System IPsec using Certificates” in **Admin Guide** provide the instructions for enabling IPsec from the controller to external systems for audit data. Section “System IPsec” in the **Admin Guide** describes the IPsec configuration for communication between the SZ controller and an external syslog server. It states that if the connection between the SZ controller and the IPsec gateway is broken after the IKE rekey timeout period, the IPsec tunnel will go down and a system event will be triggered to notify users. If the connection is broken before the IKE rekey timeout, the system IPsec sends a re-transmission request to the gateway every 10 seconds until the IKE rekey timeout or 360 re-transmission attempts. he user may need to re-connect using the Re-connect button from GUI to re-establish the connection.

Section “Configuring RadSec” in the **Admin Guide** provides instructions for configuring communication to a remote AAA server using Radius over TLS. Section “RadSec (RADIUS over TLS)” in the **Admin Guide** states that the connection between the SZ controller and Radsec Server will last for a maximum of 30 seconds. As soon as the SZ receives a new authentication request, it will initiate a TLS handshake with the Radsec Server. If the network is down or the RadSec server process itself is down, then the wireless client authentication will fail. If the connection between controller and RadSec is broken, by default the connection is resumed when the next message is received from the client. If the connection is not reestablished, the administrator should check if the network is down or RadSec server is down as

noted above.Section “Enabling Trusted Channel via IEEE 802.11-2012 (WPA2) standards” in **Admin Guide** describes how to enable a secure and trusted channel for communication by using IEEE 802.11-2012 (WPA2) standards. If the wireless communication is interrupted/broken, the user will need to reauthenticate via the wireless device to reestablish the connection.

Component Testing Assurance Activities: The evaluator will perform the following activities in addition to those required by the NDcPP:

The evaluator will perform the following tests:

Test 1: The evaluator will 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.

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

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

Test 4: The evaluator will, for each protocol associated with each authorized IT entity tested during test 1, physically interrupt an established connection. The evaluator will ensure that when physical connectivity is restored, communications are appropriately protected.

Test 1 - The configuration of the trusted channels was demonstrated in NDcPP21: FTP_ITC.1 test 1.

Test 2 - IPsec and RadSec were tested as part of NDcPP21:FTP_ITC.1 test 4. IEEE 802.11-2012 (WPA2) is demonstrated in WLANESP10:FTP_ITC.1 test 4.

Test 3 - IPsec and RadSec were tested as part of NDcPP21:FTP_ITC.1 test 4. IEEE 802.11-2012 (WPA2) is demonstrated in WLANESP10:FTP_ITC.1 test 4.

Test 4 - IPsec and RadSec were tested as part of NDcPP21:FTP_ITC.1 test 4. The evaluator configured the TOE to broadcast a wireless network protected by a WPA2 PSK (password) and started a sniffing session of wireless traffic. Using a wireless client from previous tests, the evaluator connected the client to the TOE network and confirmed that it had access to the network. Once the client was connected, the evaluator disconnected the wireless client by disabling the wifi radio on the client for 1 minute. After at least 1 minute had passed, the evaluator reconnected the client to the TOE network. The evaluator observed that the client needed to reauthenticate with the saved WPA2 PSK and transmit new EAPOL messages in order to get access with the TOE network.

Test 5 - A valid WPA2 (AES) connection was made to the TOE from a client device. This successful connection created a default profile in /etc/networkmanager/system-connections. The information from this profile was used to attempt a new connection, except the group/pairwire values were modfiied to TKIP. The connection was unsuccessful.

2.9.4 Inter-TSF Trusted Channel (WLAN Client Communications) (WLANAS10:FTP_ITC.1/Client)

2.9.4.1 WLANAS10:FTP_ITC.1.1/Client

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.4.2 WLANAS10:FTP_ITC.1.2/Client

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.4.3 WLANAS10:FTP_ITC.1.3/Client

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: This component is adequately evaluated when performing the evaluation activities for FTP_ITC.1 in the Network Device, version 2.2e base-PP.

Section 6.9 of **ST** states for wireless users, WPA3-Enterprise, WPA2-Enterprise, WPA3-SAE and WPA2-PSKas defined by IEEE 802.11-2020 are used to provide a trusted channel between the TOE and WLAN clients. If the client attempts to use another security type to establish a connection, the authentication attempt will be rejected.

Component Guidance Assurance Activities: None Defined

Component Testing Assurance Activities: None Defined

2.9.5 Trusted Path - per TD0639 (NDcPP22e:FTP_TRP.1/Admin)

2.9.5.1 NDcPP22e:FTP_TRP.1.1/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.5.2 NDcPP22e:FTP_TRP.1.2/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.5.3 NDcPP22e:FTP_TRP.1.3/Admin

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that the methods of remote TOE administration are indicated, along with how those communications are protected. The evaluator shall also confirm that all protocols listed in the TSS in support of TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

Section 6.9 in the **ST** states the controller supports all remote administrative functions via secure channels - WebUI (HTTPS) and CLI (SSH). After registration, SSH is used for all management of the distributed TOE components (AP and vSZ-D) by the SmartZone controller and IPsec is used for the data tunnel.

The WebUI (HTTPS) communication is addressed in the assurance activities for NDcPP22E:FCS_TLSS_EXT.1.

The SSH communication is addressed in the assurance activities for NDcPP22E:FCS_SSHS_EXT.1.

Component Guidance Assurance Activities: The evaluator shall confirm that the guidance documentation contains instructions for establishing the remote administrative sessions for each supported method.

Section “Administrating the Controller” in the **Admin Guide** describes in its subsections how the administrator can log in to and administer the TOE both locally via the local console and remotely via an SSH CLI session or an HTTPS

Web UI session. There is no specific configuration needed to access the SSH and WebUI session, it is enabled by default.

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

a) Test 1: The evaluators shall ensure that communications using each specified (in the guidance documentation) remote administration method is tested during the course of the evaluation, setting up the connections as described in the guidance documentation and ensuring that communication is successful.

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

Further assurance activities are associated with the specific protocols.

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

Test 1 – The evaluator used both remote interfaces, SSH and WebUI throughout the course of testing.

Test 2 - FCS_SSHS_EXT.1 testing demonstrates that the SSH management connection is not in plaintext. FCS_TLSS_EXT.1 testing demonstrates that the TLS/HTTPS management connection is not in plaintext.

2.9.6 Trusted Path - per TD0639 (NDcPP22e:FTP_TRP.1/Join)

2.9.6.1 NDcPP22e:FTP_TRP.1.1/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.6.2 NDcPP22e:FTP_TRP.1.2/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

2.9.6.3 NDcPP22e:FTP_TRP.1.3/Join

TSS Assurance Activities: None Defined

Guidance Assurance Activities: None Defined

Testing Assurance Activities: None Defined

Component TSS Assurance Activities: The evaluator shall examine the TSS to determine that the methods of joining components to the TOE are identified, along with how those communications are protected, including identification of whether the environment is required to provide confidentiality of the communications or whether the registration data exchanged does not require confidentiality. If the TSS asserts that registration data does not require confidentiality protection then the evaluator shall examine the justification provided to confirm that.

The evaluator shall also check that all protocols listed in the TSS in support of this process are included in the SFRs in the ST, and that if the ST uses FTP_TRP.1/Join for the registration channel then this channel cannot be reused as the normal inter-component communication channel (the latter channel must meet FTP_ITC.1 or FPT_ITT.1).

The evaluator shall examine the TSS to confirm that sufficient information is provided to determine the TOE actions in the case that the initial component joining attempt fails.

Section 6.2 of the **ST** states that the TOE requires the use of a dedicated channel for the AP and vSZ-D to register with a Controller. When an AP and vSZ-D discover a Controller via manual configuration or optionally via DHCP options, they will attempt to connect. Upon successful connection to the Controller, no data transfer is allowed between the Controller and AP/vSZD. An administrator must manually approve the AP/vSZ-D which enables data transfer between the Controller and AP/vSZ-D. An administrator also has the ability to manually revoke the connection between Controller and AP/vSZ-D.

Section 6.9 of the **ST** states that the TOE supports distributed TOE communication. The SmartZone controller is configured first, then other components, AP and vSZ-D are configured to join to the controller by assigning the IP address of the controller. After approval of the join request those appliances are managed by the SmartZone Controller. This initial registration is performed over a dedicated channel so confidentiality is not required. After registration, confidentiality is enforced as SSH is used for all management of the distributed TOE components (AP and vSZ-D) by the SmartZone controller and IPsec is used for the data tunnel. If the initial joining of any component (AP or vSZ-D) to the SmartZone controller fails, no communication between the components is possible, other than an attempt at rejoining. In this case the admin should verify the registration steps were performed correctly and the network setup is correct.

Component Guidance Assurance Activities: The evaluator shall examine the guidance documentation to confirm that it contains instructions for establishing and using the enablement and registration channel. The evaluator shall confirm that the guidance documentation makes clear which component initiates the communication. The evaluator shall confirm that the guidance documentation contains recovery instructions should a connection be unintentionally broken during the registration process.

In the case of a distributed TOE that relies on the operational environment to provide security for some aspects of the registration channel security then there are particular requirements on the Preparative Procedures as listed

below. (Reliance on the operational environment in this way is indicated in an ST by a reference to operational guidance in the assignment in FTP_TRP.1.3/Join.) In this case the evaluator shall examine the Preparative Procedures to confirm that they:

a) clearly state the strength of the authentication and encryption provided by the registration channel itself and the specific requirements on the environment used for joining components to the TOE (e.g. where the environment is relied upon to prevent interception of sensitive messages, IP spoofing attempts, man-in-the-middle attacks, or race conditions)

b) identify what confidential values are transmitted over the enablement channel (e.g. any keys, their lengths, and their purposes), use of any non-confidential keys (e.g. where a developer uses the same key for more than one device or across all devices of a type or family), and use of any unauthenticated identification data (e.g. IP addresses, self-signed certificates)

c) highlight any situation in which a secret value/key may be transmitted over a channel that uses a key of lower comparable strength than the transmitted value/key. Comparable strength is defined as the amount of work required to compromise the algorithm or key and is typically expressed as 'bits' of security. The ST author and evaluator should consult NIST 800-57 Table 2 for further guidance on comparable algorithm strength.

Section “Joining vSZ-D to the vSZ Controller” in the **Admin Guide** indicates that while TLS is used to perform the initial discovery of the SZ controller, this TLS is not claimed or tested in the evaluation due to limited certificate verification capabilities during the registration of the TOE components. The TOE requires the use of a dedicated channel for the AP and vSZ-D to register with the Controller. The administrator must perform the registration of TOE components in a controlled environment in which there is a segregated network with only TOE components present. The IP address of the vSZ controller is configured on the vSZ-D which then initiates the attempt to connect to the controller. The administrator then selects and approves the vSZ-D via the controller’s Web UI. Once the registration process is approved, an SSH connection is automatically established and all communication between the vSZ-D and the vSZ Controller is over SSH. If the connection between vSZ-D and vSZ is broken then it resumes back automatically and no manual intervention is required.

Section “AP Models that Support FIPS Mode” in the **Admin Guide** describes the joining of the AP to the SZ Controller. While TLS is used to perform the initial discovery of the SZ controller, this TLS is not claimed or tested in the evaluation due to limited certificate verification capabilities during the registration of the TOE components. The TOE requires the use of a dedicated channel for the AP and vSZ-D to register with the Controller. The administrator must perform the registration of TOE components in a controlled environment in which there is a segregated network with only TOE components present. Section “Joining AP to (v)SZ Controller” in the **Admin Guide** describes the steps for joining of the AP to the SZ Controller. The IP address of the SZ controller is configured on the AP which then initiates the attempt to connect to the controller. The administrator then selects and approves the AP via the controller’s Web UI. Once the registration process is approved, an SSH connection is automatically established between the AP and the SZ Controller. The connection with the AP can similarly be disabled if the administrator selects the AP and then clicks the ‘Delete’ button. If the connection is broken it will be resumed/reattempted without any user intervention.

Section “Management Channel Between AP/vSZ-D and Controller” in the **Admin Guide** further describes the SSH communication between the AP and SZ controller and between the vSZ-D and the vSZ controller. The SSH algorithms, parameters and rekey limits are not configurable. The communication is only through public key auth (no password based authentication).

Section “Administrating the controller remotely” in the **Admin Guide** provides all of the SSH algorithms and key exchange methods supported by the TOE in its implementations as both client and server for remote authentication and for communication between the distributed TOE components.

Before any communication with a wireless client occurs, an IPsec tunnel is formed between the AP and SZ or the vSZ-D. Once the wireless client is authenticated, all user data traffic is protected using IPsec. Section “System IPsec” in the **Admin Guide** states that Ruckus GRE and IPsec is a configuration of the IPsec tunnel between the AP and the SZ or vSZ-D. Section “Configuring Ruckus GRE and IPsec in the WLAN” in the **Admin Guide** provides instructions for configuring the IPsec tunnel between these components.

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

a) Test 1: The evaluator shall ensure that the communications path for joining components to the TSF is tested for each distinct (non-equivalent) component type [The intention here is to cover all different software sections involved. For example, a single software image may be installed on different TOE components, but with different sections of the image executed according to the hardware platform or communications stack. In such as case tests should be carried out for each different software section.], setting up the connections as described in the guidance documentation and ensuring that communication is successful. In particular the evaluator shall confirm that requirements on environment protection for the registration process are consistent with observations made on the test configuration (for example, a requirement to isolate the components from the Internet during registration might be inconsistent with the need for a component to contact a license server). If no requirements on the registration environment are identified as necessary to protect confidentiality, then the evaluator shall confirm that the key used for registration can be configured (following the instructions in the guidance documentation) to be at least the same length as the key used for the internal TSF channel that is being enabled. The evaluator shall confirm that the key used for the channel is unique to the pair of components (this is done by identifying the relevant key during the registration test: it is not necessary to examine the key value).

b) Test 2: The evaluator shall follow the guidance documentation to ensure that in fact the communication channel can be enabled by a Security Administrator for all the TOE components identified in the guidance documentation as capable of initiation.

c) Test 3: The evaluator shall ensure that if the guidance documentation states that the channel data is encrypted then the data observed on the channel is not plaintext.

Further assurance activities are associated with the specific protocols.

Test 1 - The TOE uses a dedicated channel to ensure the registration channel is protected from modification and ensures that the desired TOE component is the one being registered. As part of the pre setup process of the AP the evaluator configured the AP to point to the SZ controller for management. The evaluator viewed that the AP

appeared in the SZ controller's AP list with all the required identifying information to ensure that it was the intended AP. The evaluator chose to Approve the AP based on the presented information. The evaluator verified that the AP was now being managed by the SZ controller.

As part of the pre setup process of the vSZ-D the evaluator configured the vSZ-D to point to the SZ controller for management. The evaluator viewed that the vSZ-D appeared in the SZ controller's dataplane list with all the required identifying information to ensure that it was the intended vSZ-D. The evaluator chose to Approve the vSZ-D based on the presented information. The evaluator verified that the TOE was now being managed by the SZ controller.

Test 2 – see Test 1

Test 3 – The TOE uses a dedicated channel to ensure that the registration data cannot be modified.

3. Protection Profile SAR Assurance Activities

The following sections address assurance activities specifically defined in the claimed Protection Profile that correspond with Security Assurance Requirements.

3.1 Development (ADV)

3.1.1 Basic Functional Specification (ADV_FSP.1)

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

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

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

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

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

In this context, TSFI are deemed security relevant if they are used by the administrator to configure the TOE, or to perform other administrative functions (e.g. audit review or performing updates). Additionally, those interfaces that are identified in the ST, or guidance documentation, as adhering to the security policies (as presented in the SFRs), are also considered security relevant. The intent is that these interfaces will be adequately tested, and having an understanding of how these interfaces are used in the TOE is necessary to ensure proper test coverage is applied.

The set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

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

The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.

The evaluator uses the provided documentation and first identifies, and then examines a representative set of interfaces to perform the EAs presented in Section 2, including the EAs associated with testing of the interfaces.

It should be noted that there may be some SFRs that do not have an interface that is explicitly 'mapped' to invoke the desired functionality. For example, generating a random bit string, destroying a cryptographic key that is no longer needed, or the TSF failing to a secure state, are capabilities that may be specified in SFRs, but are not invoked by an interface.

However, if the evaluator is unable to perform some other required EA because there is insufficient design and interface information, then the evaluator is entitled to conclude that an adequate functional specification has not been provided, and hence that the verdict for the ADV_FSP.1 assurance component is a 'fail'.

For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No additional 'functional specification' documentation is necessary to satisfy the Evaluation Activities specified in the Supporting Document [SD].

3.2 Guidance documents (AGD)

3.2.1 Operational User Guidance (AGD_OPE.1)

Assurance Activities: The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps (per TD0536):

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR.

In addition, the evaluator performs the EAs specified below.

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

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

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

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

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

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

b) The documentation must describe the process for verifying updates to the TOE by verifying a digital signature. The evaluator shall verify that this process includes the following steps:

1) Instructions for obtaining the update itself. This should include instructions for making the update accessible to the TOE (e.g., placement in a specific directory).

2) Instructions for initiating the update process, as well as discerning whether the process was successful or unsuccessful. This includes instructions that describe at least one method of validating the hash/digital signature.

c) The TOE will likely contain security functionality that does not fall in the scope of evaluation under this cPP. The guidance documentation shall make it clear to an administrator which security functionality is covered by the Evaluation Activities.

The **Admin Guide** identifies the evaluated hardware and software in the following sections: “vSZ Installation with FIPS Image”, “Hardware Configuration with FIPS Image”, “vSZ-D FIPS Installation with FIPS Image” and the “AP Configuration in FIPS Mode”.

Throughout the **Admin Guide**, there are notes clarifying the boundaries of the TOE and anything that is excluded in the evaluated configuration making it clear to an administrator which security functionality and interfaces have been assessed and tested in the evaluated configuration. Section "AP Features Not Supported in FIPS Mode" identifies all of the features that are not available in FIPS mode which is enabled in the evaluated configuration.

Section "FIPS Mode Overview" in the **Admin Guide** states that enabling FIPS mode ensures that only CC compliant cryptographic algorithms and processes are allowed in the TOE.

Section "Using FIPS-related CLI commands" of the **Admin Guide** describes the steps to enable FIPS mode which ensures that cryptographic algorithms and key sizes are restricted to those supported by the TOE.

Section "Upgrading the Software" in the **Admin Guide** provides instructions for updating the software.

Section "Upgrading (v)SZ Software" in the **Admin Guide** provides instructions for updating the software and for querying the currently active version. This includes downloading the update image from the Ruckus Customer release web site and then uploading it to the Controller. The web interface lists the active and inactive upgrade history. Once uploaded, delayed activation/upgrade can be initiated. The upgrade package contains the upgrade software/firmware, signatures and the certificates of the signature signers. When the upgrade package is uploaded to the controller, the controller will validate the certificate chain first. If the certificate of the signature signer passes the chain validation, the controller then verifies the signatures of the upgrade software/firmware. When the upgrade package signature signer certificate chain validation error or signature verification error occur, the GUI shows a package decryption error. In such cases, a valid upgrade package should be used to continue system upgrading.

The "Upgrading the AP Software" section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to AP Software are addressed via AP firmware associated with a firmware version which is bundled as part of the (v) SZ Software upgrade image. This section further provides instructions for manually upgrading the AP firmware version. The Firmware software contains the upgrade software, Signatures and certificates of the signature signers. When the Firmware is pushed to the AP from the (v)SZ, the AP will validate the Certificate Chain first once the Chain validation goes through then AP validates the Signatures of the upgrade firmware. If any of this validation fails first, the upgrade will fail and the corresponding status will be shown on UI and detailed info can be viewed through logs.

The "Upgrading the vSZ-D Software" section in the **Admin Guide** states that feature enhancements or fixes or known issues pertaining to vSZ-D software are addressed through vSZ-D Patch and provides instructions for upgrading the vSZ-D software. The upgrade patch contains the upgrade software/firmware, signatures and the certificates of the signature signers. When the upgrade package is uploaded to the (v)SZ, (v)SZ will validate the certificate chain first. If the certificate of the signature signer passes the chain validation, the (v)SZ then verifies the signatures of the upgrade software/firmware. When the upgrade package signature signer certificate chain validation error or signature verification error occur, the GUI shows a package decryption error. In such cases, a valid upgrade package should be used to continue system upgrading.

Section "Self-Service Resources" indicates that software downloads and release notes can be obtained via the Ruckus Support Portal at https://support.ruckuswireless.com.

3.2.2 PREPARATIVE PROCEDURES (AGD_PRE.1)

Assurance Activities: As with the operational guidance, the developer should look to the Evaluation Activities to determine the required content with respect to preparative procedures.

It is noted that specific requirements for Preparative Procedures are defined in [SD] for distributed TOEs as part of the Evaluation Activities for FCO_CPC_EXT.1 and FTP_TRP.1(2)/Join.

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

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

In addition, the evaluator performs the EAs specified below.

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

The documentation should be in an informal style and should be written with sufficient detail and explanation that they can be understood and used by the target audience (which will typically include IT staff who have general IT experience but not necessarily experience with the TOE product itself).

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

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

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

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

The preparative procedures must

a) include instructions to provide a protected administrative capability; and

b) identify TOE passwords that have default values associated with them and instructions shall be provided for how these can be changed.

The evaluation team had the following documents to use when configuring the TOE:

- RUCKUS FIPS and Common Criteria Configuration Guide for SmartZone and APs, 5.2.1.3, Part Number: 800-72735-001 Rev D, June 2023 **(Admin Guide)**

In some instances, the document referenced general Ruckus manuals which the evaluation team could find on the Ruckus web site. The completeness of the documentation is addressed by their use in the AA's carried out in the evaluation.

3.3 Life-cycle support (ALC)

3.3.1 Labelling of the TOE (ALC_CMC.1)

Assurance Activities: This component is targeted at identifying the TOE such that it can be distinguished from other products or versions from the same vendor and can be easily specified when being procured by an end user. A label could consist of a 'hard label' (e.g., stamped into the metal, paper label) or a 'soft label' (e.g., electronically presented when queried).

The evaluator performs the CEM work units associated with ALC_CMC.1.

When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

The evaluator verified that the **ST**, TOE and Guidance are all labeled with the same hardware versions and software. The information is specific enough to procure the TOE and it includes hardware models and software versions. The evaluator checked the TOE software version and hardware identifiers during testing by examining the actual machines used for testing.

3.3.2 TOE CM Coverage (ALC_CMS.1)

Assurance Activities: Given the scope of the TOE and its associated evaluation evidence requirements, the evaluator performs the CEM work units associated with ALC_CMS.1.

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

See section 3.3.1 above for an explanation of how all CM items are addressed.

3.4 Tests (ATE)

3.4.1 Independent Testing - Conformance (ATE_IND.1)

Assurance Activities: Testing is performed to confirm the functionality described in the TSS as well as the guidance documentation (includes 'evaluated configuration' instructions). The focus of the testing is to confirm that the requirements specified in Section 5.1.7 are being met. The Evaluation Activities in [SD] identify the specific testing activities necessary to verify compliance with the SFRs. The evaluator produces a test report documenting the plan for and results of testing, as well as coverage arguments focused on the platform/TOE combinations that are claiming conformance to this cPP.

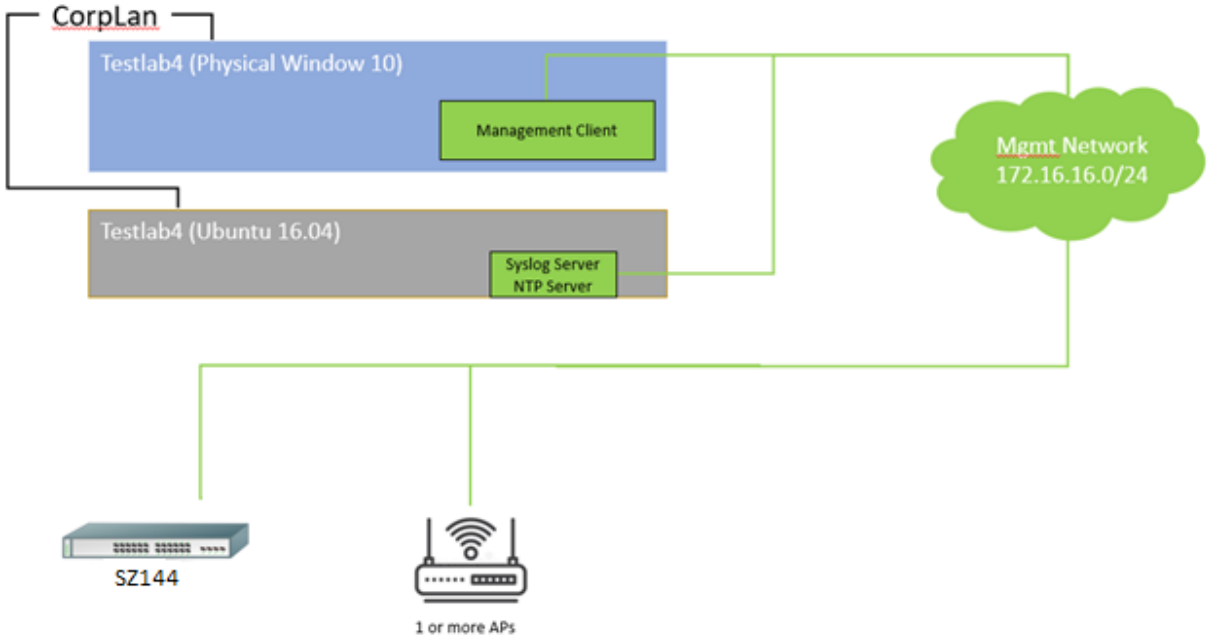
The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

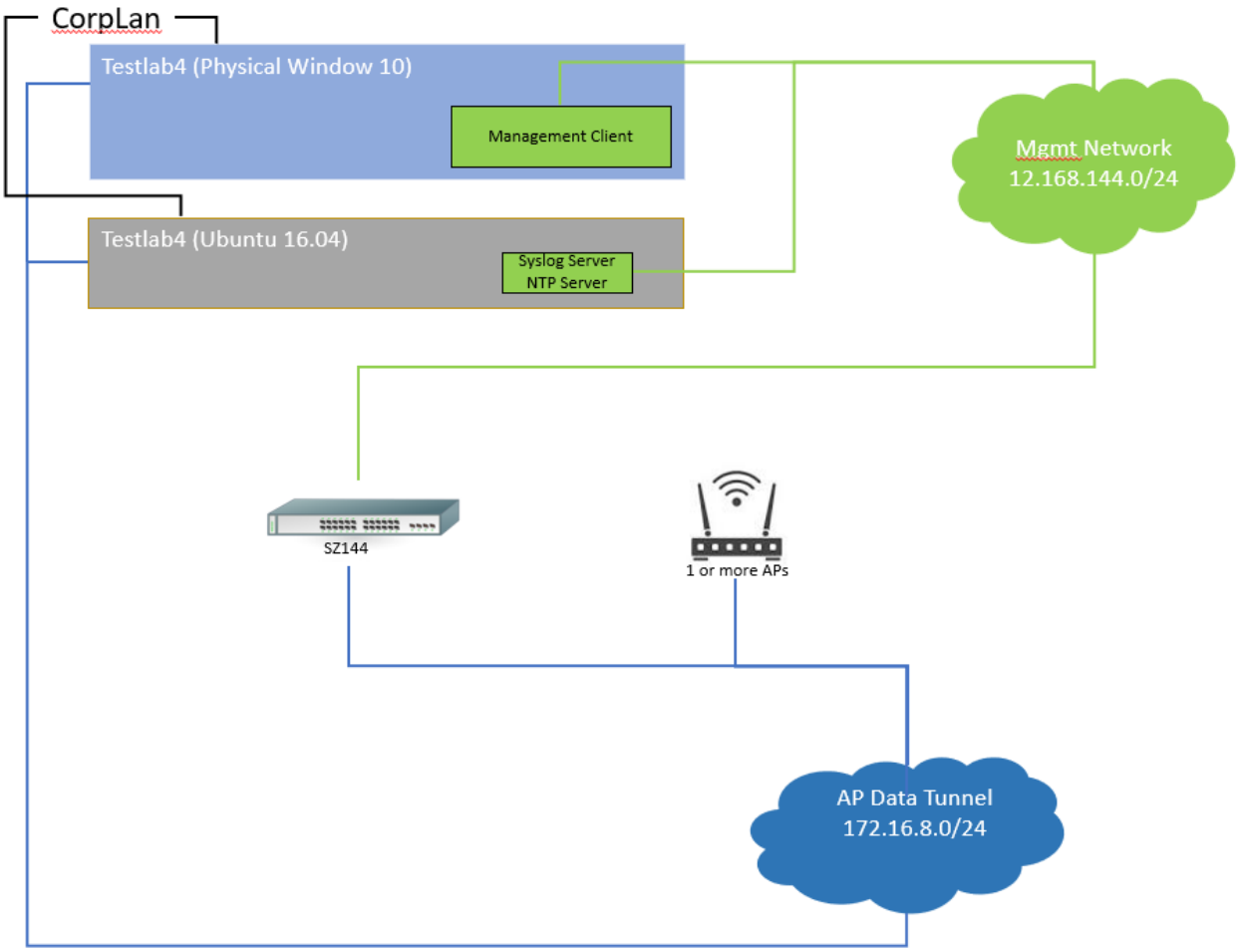
The evaluator should consult Appendix B when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section B.4.3.1.

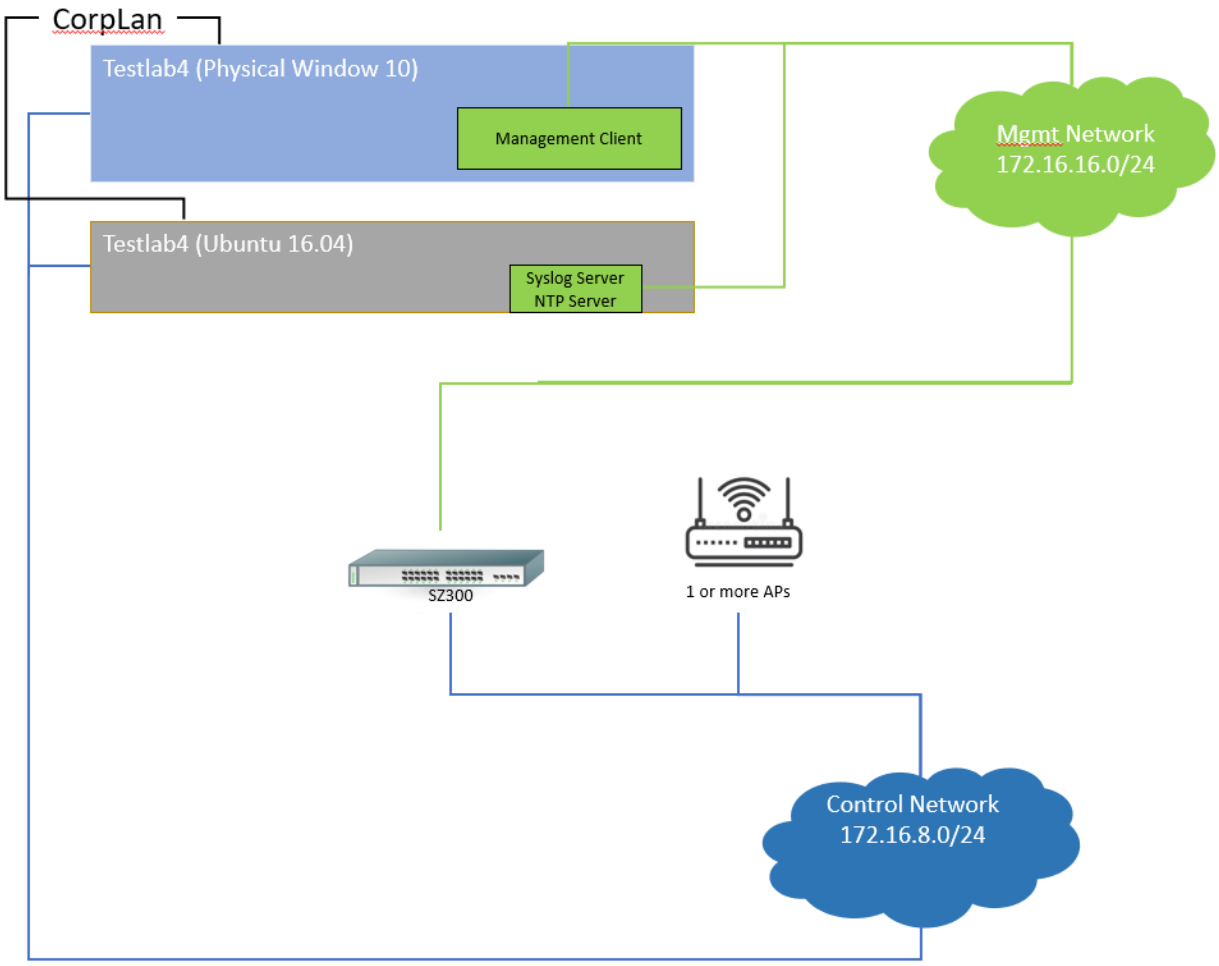
The evaluator created a Detailed Test Report (DTR) to address all aspects of this requirement. The DTR discusses the test configuration, test cases, expected results, and test results. The following diagrams indicate the test environment:



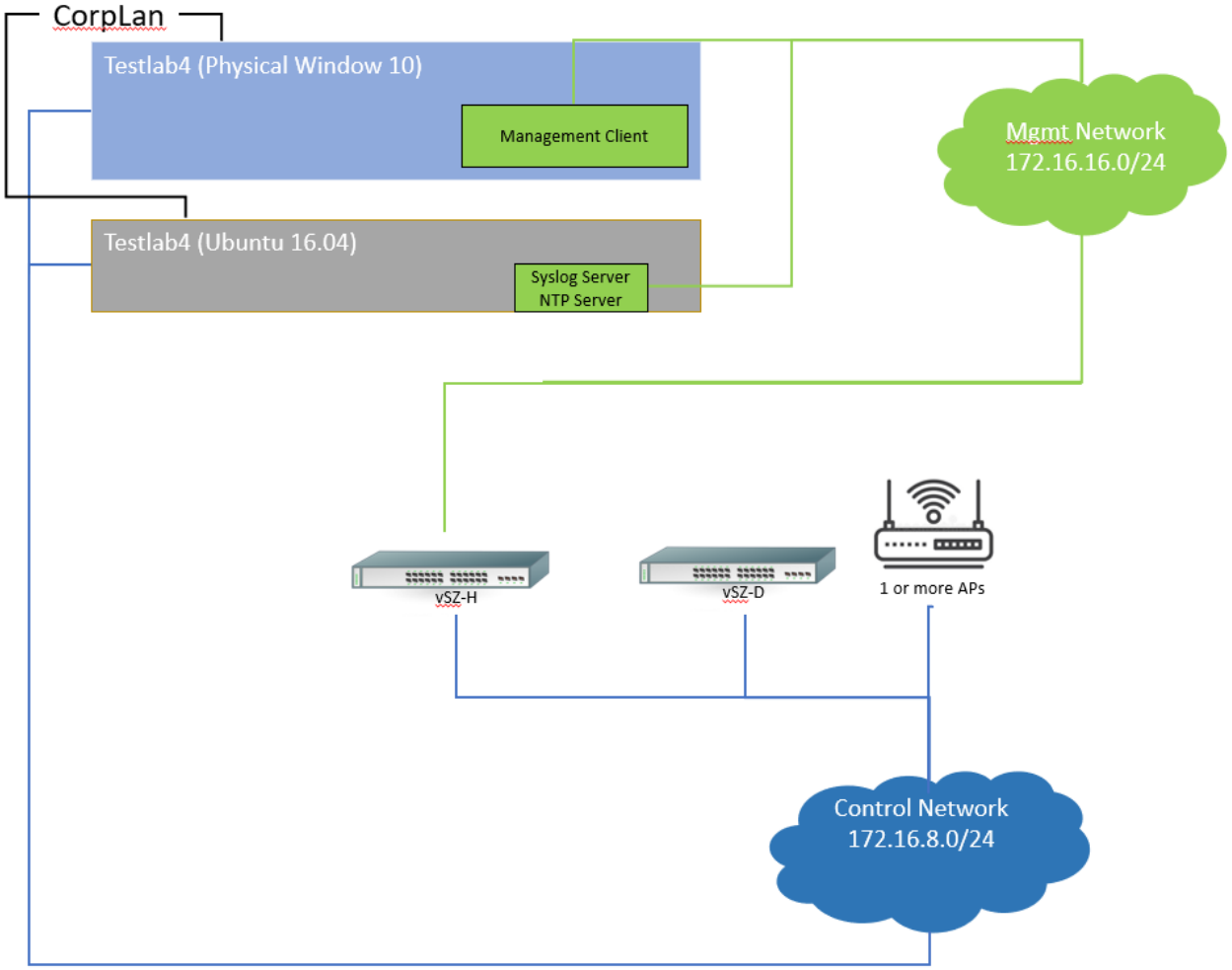
SZ144 Test Setup



SZ144 RGRE+IPsec Setup



SZ300 Test Setup



vSZ-H on ESXi Test Setup

Figure 1 Test Setup

TOE Platforms:

- SZ144
- SZ300
- vSZ-H running on ESXi 7.0
- vSZ-D running on ESXi 7.0
- R650 AP
- R750 AP

- R850 AP

Supporting Products and Software:

- Windows 10, 64-bit
  - Standard Windows utilities (e.g., notepad, snip tool)
  - Putty release 0.73
  - HxD (Hexeditor) version 1.7.7.0
  - Wireshark version 3.2.2
  - Microsoft Hyper-V (part of Windows)
- Ubuntu version 16.04, 64-bit
  - Standard Linux commands (e.g., cat, grep, awk)
  - OpenSSL version 1.0.2g-fips
  - Strongswan version U5.3.5
  - Stunnel version 5.30
  - FreeRADIUS Version 3.0.15
  - tcpdump
  - rsyslog version 8.16.0
  - ntpd version ntpd 4.2.8p4
  - Evaluator developed test scripts
- Kali Linux
- Wireshark version 1.10.0
- Nmap version 6.25

3.5 Vulnerability assessment (AVA)

3.5.1 Vulnerability Survey (AVA_VAN.1)

Assurance Activities: While vulnerability analysis is inherently a subjective activity, a minimum level of analysis can be defined and some measure of objectivity and repeatability (or at least comparability) can be imposed on the vulnerability analysis process. In order to achieve such objectivity and repeatability it is important that the evaluator follows a set of well-defined activities, and documents their findings so others can follow their arguments and come to the same conclusions as the evaluator. While this does not guarantee that different evaluation facilities will identify exactly the same type of vulnerabilities or come to exactly the same conclusions, the approach defines the minimum level of analysis and the scope of that analysis, and provides Certification Bodies a measure of assurance that the minimum level of analysis is being performed by the evaluation facilities.

In order to meet these goals some refinement of the AVA_VAN.1 CEM work units is needed. The following table indicates, for each work unit in AVA_VAN.1, whether the CEM work unit is to be performed as written, or if it has been clarified by an Evaluation Activity. If clarification has been provided, a reference to this clarification is provided in the table.

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

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

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

The developer shall provide documentation identifying the list of software and hardware components7 that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside the TOE) such as a web server and protocol or cryptographic libraries. This additional documentation is merely a list of the name and version number of the components, and will be used by the evaluators in formulating hypotheses during their analysis.

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

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

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

c) additional information in the Preparative Procedures as identified in the refinement of AGD_PRE.1 in additional information in the Preparative Procedures as identified in 3.5.1.2 and 3.6.1.2.

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

The vulnerability analysis is in the Detailed Test Report (DTR) prepared by the evaluator. The vulnerability analysis includes a public search for vulnerabilities and fuzz testing. None of the public search for vulnerabilities, or the fuzz testing uncovered any residual vulnerability.

The evaluation team performed a public search for vulnerabilities in order to ensure there are no publicly known and exploitable vulnerabilities in the TOE from the following sources:

- National Vulnerability Database (https://web.nvd.nist.gov/vuln/search)
- Vulnerability Notes Database (http://www.kb.cert.org/vuls/)
- Rapid7 Vulnerability Database (https://www.rapid7.com/db/vulnerabilities)
- Tipping Point Zero Day Initiative (http://www.zerodayinitiative.com/advisories)

- Exploit / Vulnerability Search Engine (http://www.exploitsearch.net)
- SecurITeam Exploit Search (http://www.securiteam.com)
- Tenable Network Security (http://nessus.org/plugins/index.php?view=search)
- Offensive Security Exploit Database (https://www.exploit-db.com/)

The search was performed on 9/7/2023 with the following search terms: "802.1X", "Ruckus", "SmartZone", "ESXi", "Xeon Processor", "Qualcom IPQ", "openssl", "openssh".

Assurance Activities: The following additional tests shall be performed:1.) [Conditional]: If the TOE is a TLS server and supports ciphersuites that use RSA transport (e.g. supporting TLS_RSA_WITH_* ciphers) the following test shall be performed. Where RSA Key Establishment schemes are claimed and especially when PKCS#1 v1.5* padding is used, the evaluators shall test for implementation flaws allowing Bleichenbacher and Klima et al. style attacks, including Bock et al's ROBOT attacks of 2017 in the flaw analysis. Even though Bleichenbacher's original paper is two decades old, Bock et al. found these attacks to still be effective in weakening the security of RSA key establishment in current products. Bleichenbacher and Klima et al. style attacks are complex and may be difficult to detect, but a number of software testing tools have been created to assist in that process. The iTC strongly recommends that at least one of the tools mentioned in Bock et al's ROBOT attacks of 2017 webpage or paper, as effective to detect padding oracle attacks, be used to test TOE communications channels using RSA based Key Establishment (related sources: http://archiv.infsec.ethz.ch/education/fs08/secsem/bleichenbacher98.pdf, https://eprint.iacr.org/2003/052, https://robotattack.org/). Network Device Equivalency Considerations

Not applicable as the TOE does not support TLS_RSA ciphersuites.