Aruba Remote Access Points and Aruba Mobility Controllers with ArubaOS 8.10

Assurance Activity Report

Version 1.3

November 14, 2023

Document prepared by

www.lightshipsec.com

Document History

Version	Date	Author	Reviewer	Description
1.0	29-Sept-2023	G. McLearn	C. Cantlon	Initial release
1.1	5-Oct-2023	G. McLearn		Updated ST and AGD references
1.2	8-Nov-2023	G. McLearn	C. Cantlon	Updated for ECR
1.3	14-Nov-2023	G. McLearn		Updated for ECR

Table of Contents

1 Introduction

1 This Assurance Activity Report (AAR) documents the evaluation activities performed by Lightship Security for the evaluation identified in Table 1. The AAR is produced in accordance with National Information Assurance Program (NIAP) reporting guidelines.

1.1 Evaluation Identifiers

Table 1: Evaluation Identifiers

Scheme	US Common Criteria Scheme
Evaluation Facility	Lightship Security USA
Developer/Sponsor	Aruba, a Hewlett Packard Enterprise Company 6280 America Center Dr. San Jose, CA 95002
TOE	Aruba Remote Access Points and Aruba Mobility Controllers with ArubaOS 8.10.0.8-FIPS
Security Target	Aruba Remote Access Points and Aruba Mobility Controllers with ArubaOS 8.10 Security Target, v1.6
Protection Profile	collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020 [NDcPP] PP-Module for VPN Gateways, Version: 1.2, 2022-03-31 [VPNMOD] PP-Configuration for Network Device and Virtual Private Network (VPN) Gateways, Version 1.2, 31 March 2022 [CFG_NDcPP-VPNGW_V1.2]

1.2 Evaluation Methods

2 The evaluation was performed using the methods and standards identified in Table 2.

Table 2: Evaluation Methods

Evaluation Criteria	CC v3.1R5
Evaluation Methodology	CEM v3.1R5
Supporting Documents	Evaluation Activities for Network Device cPP, December-2019, Version 2.2 [NDSD] Supporting Document Mandatory Technical Document PP-Module for VPN Gateways, Version: 1.2, 2022-03-31 [VPNSD]

Interpretations	TD #	Name	Source	Exclusion Rationale
	TD0527	Updates to Certificate Revocation Testing (FIA_X509_EXT.1)	[NDcPP]	
	TD0528	NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	[NDcPP]	
	TD0536	NIT Technical Decision for Update Verification Inconsistency	[NDcPP]	
	TD0537	NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	[NDcPP]	
	TD0546	NIT Technical Decision for DTLS – clarification of Application Note 63	[NDcPP]	Not applicable: DTLS is not claimed.
	TD0547	NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	[NDcPP]	
	TD0555	NIT Technical Decision for RFC Reference incorrect in TLSS Test	[NDcPP]	
	TD0556	NIT Technical Decision for RFC 5077 question	[NDcPP]	
	TD0563	NiT Technical Decision for Clarification of audit date information	[NDcPP]	
	TD0564	NiT Technical Decision for Vulnerability Analysis Search Criteria	[NDcPP]	
	TD0569	NIT Technical Decision for Session ID Usage Conflict in	[NDcPP]	

	FCS_DTLSS_ EXT.1.7		
TD0570	NiT Technical Decision for Clarification about FIA_AFL.1	[NDcPP]	
TD0571	NiT Technical Decision for Guidance on how to handle FIA_AFL.1	[NDcPP]	
TD0572	NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	[NDcPP]	
TD0580	NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	[NDcPP]	
TD0581	NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	[NDcPP]	
TD0591	NIT Technical Decision for Virtual TOEs and hypervisors	[NDcPP]	
TD0592	NIT Technical Decision for Local Storage of Audit Records	[NDcPP]	
TD0631	NIT Technical Decision for Clarification of public key authentication for SSH Server	[NDcPP]	
TD0632	NIT Technical Decision for Consistency with Time Data for vNDs	[NDcPP]	
TD0633	NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	[NDcPP]	

TD0635	NIT Technical Decision for TLS Server and Key Agreement Parameters	[NDcPP]	
TD0636	NIT Technical Decision for Clarification of Public Key User Authentication for SSH	[NDcPP]	Not applicable: FCS_SSHC_EXT.1 is not claimed.
TD0638	NIT Technical Decision for Key Pair Generation for Authentication	[NDcPP]	
TD0639	NIT Technical Decision for Clarification for NTP MAC Keys	[NDcPP]	
TD0656	Missing EAs for VPN GW Optional Headend SFRs	[VPNMOD]	Not applicable: FTA_SSL.3/VPN not claimed.
TD0657	IPSEC_EXT.1.6 GCM support for VPN GW	[VPNMOD]	
TD0670	NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	[NDcPP]	Not applicable: FCS_TLSC_EXT.2 is not claimed.
TD0683	RFC 2460 to be replaced with RFC 8200	[VPNMOD]	
TD0723	Correction to ECDSA Curve Selection	[VPNMOD]	
TD0738	NIT Technical Decision for Link to Allowed-With List	[NDcPP]	
TD0771	Correction to FIA_PSK_EXT.3 EA	[VPNMOD]	Not applicable: FIA_PSK_EXT.3 not claimed.
TD0790	NIT Technical Decision: Clarification Required for testing IPv6	[NDcPP]	Not applicable: FCS_(D)TLSC_EXT.* are not claimed.

	TD0792	NIT Technical Decision: FIA_PMG_EXT.1 - TSS EA not in line with SFR	[NDcPP]	

1.3 Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	Aruba Remote Access Points and Mobility Controllers with ArubaOS 8.10 Security Target, v1.6
[SUPP]	Aruba Common Criteria Configuration Guidance ArubaOS 8.10 Supplemental Guidance (For Aruba Remote Access Points with Mobility Controllers running ArubaOS 8.10-FIPS), Version 2.3, November 2023
[ADMIN]	ArubaOS 8.10.0.0 User Guide, Revision 14, 2023
[CLI]	ArubaOS 8.x Command-Line Interface Reference Guide, 2023
[SYSLOG]	ArubaOS 8.10.0.0 Syslog Reference Guide
[INSTALL]	Aruba 303H Series Hospitality Access Points Installation Guide, March 2017 Aruba 503H Series Hospitality Access Points Installation Guide, July 2020 Aruba AP-505H Access Points Installation Guide, May 2023 Aruba 7200 Series Controller Installation Guide 0511169-06 | July 2015 Aruba 9004 Gateway Installation Guide, Revision 03 | June 2021
[NDcPP]	collaborative Protection Profile for Network Devices, v2.2e, 23-March-2020
[VPNMOD]	PP-Module for VPN Gateways, Version: 1.2, 2022-03-31
[NDSD]	Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP, December-2019, Version 2.2
[VPNSD]	Supporting Document Mandatory Technical Document PP-Module for VPN Gateways, Version: 1.2, 2022-03-31
[CFG_NDcPP-VPNGW_V1.2]	PP-Configuration for Network Device and Virtual Private Network (VPN) Gateways, Version 1.2, 31 March 2022

2 TOE Details

2.1 TOE Models

The physical boundary of the TOE includes the Aruba Mobility Controller and Remote Access Point hardware models shown in the table below.

Table 4: TOE Models

Type	Model	CPU	Software
Mobility Controller	7210	Broadcom XLP416 (MIPS64)	ArubaOS 8.10
Mobility Controller	7220	Broadcom XLP432 (MIPS64)	
Mobility Controller	9004	Intel Atom C3508 (Denverton)	
Remote Access Point	303H	Qualcomm IPQ4019 (ARM Cortex-A7)	
Remote Access Point	503H	Broadcom BCM47622L (ARM-A7)	
Remote Access Point	505H	Broadcom BCM47622L (ARM-A7)	

2.1.1 Test Platform Equivalency

3 The team used the [NDcPP] as the basis for the equivalency rationale. The equivalency rationale has been provided in the proprietary Detailed Test Report (DTR).

2.1.2 TOE Test Configuration

4 The following diagram provides a high level overview of the test environment.

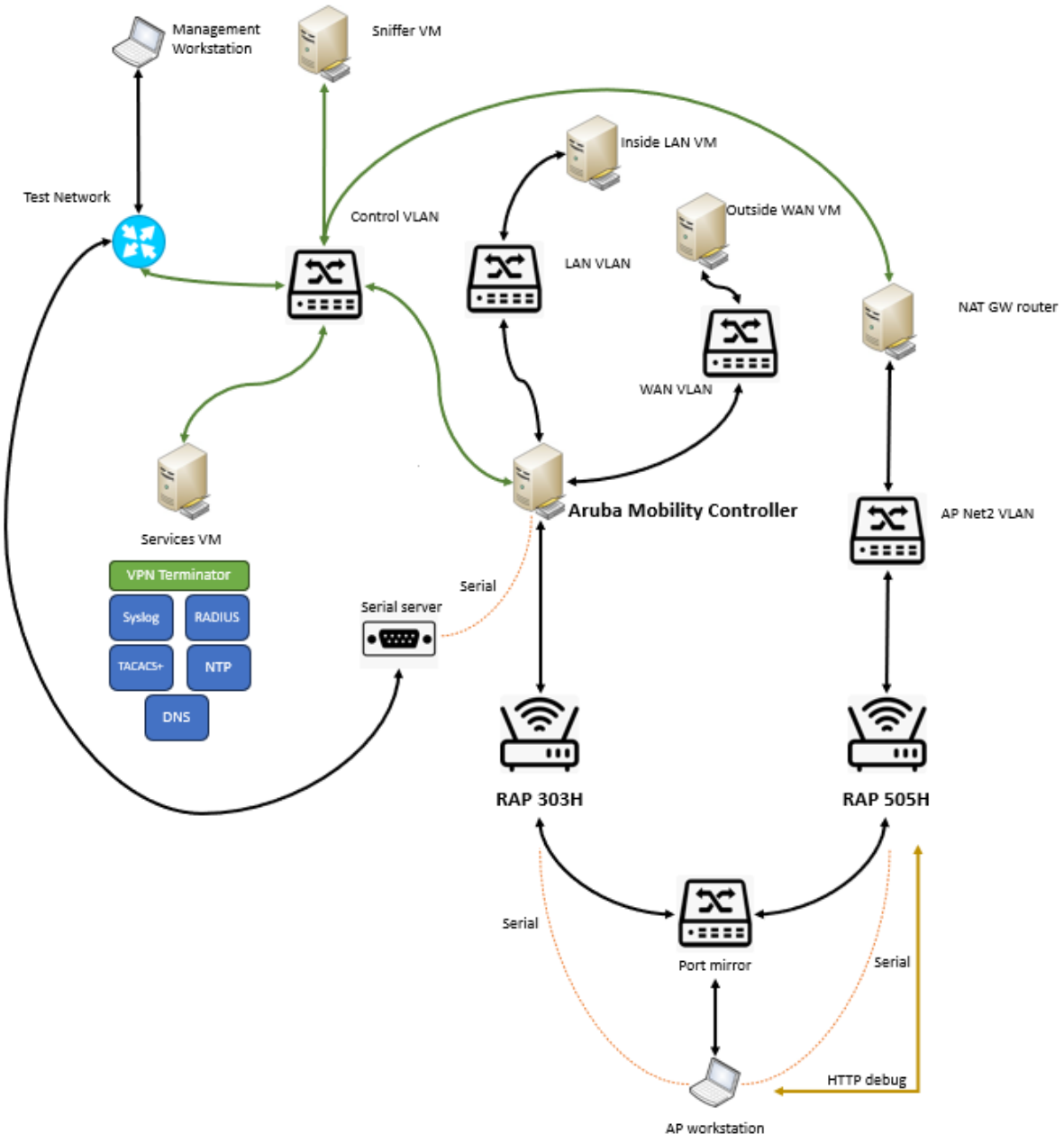

Figure 1 - Test Setup

3 Evaluation Activities for the collaborative Protection Profile for Network Devices

3.1 Security Audit (FAU)

3.1.1 FAU_GEN.1 Audit data generation

3.1.1.1 TSS

5 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.

Findings:	As described in section 6.1.1 of the [ST], 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 Table 15 and Table 16 (in the [ST]). For audit records involving the generating/import of, changing, or deleting of cryptographic keys, the record identifies the key via reference to the certificate or key identifier associated with the key.

6 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.

Findings:	All findings can be referenced in section 6.1 of the [ST]. The TOE generates audit records for security relevant and other events as they occur. The events that can cause an audit record to be logged include: start-up and shutdown of the TOE; all attempts to initiate a secure communication channel; and all administrator actions comprising: • Administrative login and logout (including the name of the user account). • Enabling and disabling communications between a pair of components. • Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed). • Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference is logged). • Resetting passwords (name of related user account is logged). • Attempts to initiate a TOE update. • Modification of the behavior of the transmission of audit data to an external IT entity. The Mobility Controller will generate audit events for all security functions. Remote Access Points generate audit records for the following security relevant audit events which occur on that device. 1. Shutdown of AP/auditing 2. IPsec connection failures

3. Successful/unsuccessful re-imaging

The Remote Access Points do not store any audit records, but rather forward all audit events securely over an IPsec connection to the Controller.

The Controller locally stores the audit records and forwards the audit record in real time to an external audit server.

Table 15 of the [ST] corresponds to the audit events specified in table 2 of the NDcPP and includes the audit events specified in the NDcPP for optional and selected SFRs as selected in the ST.

The TOE identifies the responsible user for each event based on the specific administrator or network entity (identified by IP address) that caused the event.

3.1.1.2 Guidance Documentation

7 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).

Findings: Section 2.1.1 of the [SUPP] includes relevant examples of the claimed functions. The evaluator cross-referenced the total set of SFR claims in the [ST] against the table of auditable messages in 2.1.1 of the [SUPP] and found it to be complete.

8 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.

Findings: This work was conducted as part of FMT_MTD.1/CoreData in section 3.5.4.2 below. The evaluator reviewed the scope of the TOE and the set of guidance documents. Functionality that was found to be contrary to the evaluated configuration were raised with the developer for review to ensure that it was removed from the scope and/or documented to be explicitly disabled.

3.1.1.3 Tests

9 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.

High-Level Test Description
These activities are performed within each of the test cases that required audit messages be generated.
Findings: PASS - The evaluator found audit messages needed to satisfy each of the claimed functions. The audit messages were formatted similarly to those in the guidance document.

10 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.

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

High-Level Test Description
These activities are performed within each of the test cases that required audit messages be generated.
Findings: PASS - The evaluator found audit messages needed to satisfy each of the claimed functions within the distributed TOE.

3.1.2 FAU_GEN.2 User identity association

3.1.2.1 TSS & Guidance Documentation

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

3.1.2.2 Tests

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

14 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.

High-Level Test Description
These activities are performed within each of the test cases that required audit messages be generated.
Findings: PASS - The evaluator found that audit messages generated by the various components were identified by the component which either generated it or responsible for witnessing it.

3.1.3 FAU_GEN_EXT.1 Security Audit Data Generation for Distributed TOE Components

15 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.

3.1.4 FAU_STG_EXT.1 Protected audit event storage

3.1.4.1 TSS

16 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.

Findings:	Section 6.1.2 of the [ST] states that the TOE uses IPsec to protect the communication channel between itself and the remote syslog server and also between the Controller and the RAPs. If an external syslog server has been enabled, all audit logs are simultaneously (in real-time) written to both the local audit log on the Mobility Controller and the syslog server. The local audit logs and logs sent to a remote server are identical.

17 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.

Findings:	Section 6.1.2 of the [ST] states that for audit records stored locally on MC, the maximum log file size for all processes is 1.04 MiB. There are three log files comprising the audit log and each has a maximum file size of 341 KiB. The local MC protected log storage operates using the first in, first out (FIFO) method, therefore audit logs are overwritten when the available space is exhausted. Finally, section 6.1.2 of the [ST] claims the TOE protects audit records in local storage from unauthorized modification or deletion. There are no CLI or GUI commands to delete or modify the local logs.

18 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.

Findings:	Section 6.1.1 of the [ST] states that the Remote Access Points (RAP) do not store any audit records, but rather forward all audit events securely over an IPsec connection to the Controller. The Controller locally stores the audit records and forwards the audit record in real time to an external audit server. The Controller and the RAP devices comprise a distributed TOE rather than a standalone TOE.

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

Findings:	Section 6.1.2 of the [ST] states that the local Mobility Controller log storage operates using the first in, first out (FIFO) method, therefore audit logs are overwritten when the available space is exhausted.

20 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 as well as acceptable frequency for the transfer of audit data.

Findings:	Section 6.1.2 of the [ST] clarifies that all transfers occur in real time.

21 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).

Findings:	Section 6.1.1 of the [ST] states that the Remote Access Points (RAP) do not store any audit records, but rather forward all audit events securely over an IPsec connection to the Controller. The Controller locally stores the audit records and forwards the audit record in real time to an external audit server. Thus, the Controller acts as a central storage mechanism for all audit events from all claimed distributed RAP devices and forwards everything to the external audit server.

22 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.

Findings:	Section 6.1.2 of the [ST] states that only the Mobility Controller stores audit records locally. For the local storage on this Mobility Controller, section 6.1.2 of the [ST] also describes the local storage exhaustion strategies as a FIFO-based overwriting mechanism. The Remote Access Point (RAP) devices forward data to the Mobility Controller in real-time.

3.1.4.2 Guidance Documentation

23 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.

Findings:	Section 2.1.4 of the [SUPP] provides instructions that the audit records are offloaded using IPSec to an external IT entity. The system permits the use of syslog as the logging protocol. According to [ADMIN] in section “Management Access > Configuring Logging”, the format of the syslog messages can be configured by the administrator within the TOE as being “CEF” or “BSD-standard”. The TOE is required to be configured to send syslog messages over an established IPSec tunnel. The instructions for setting up such a tunnel are exemplified in section 2.1.4 of the [SUPP].

24 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.

Findings:	Section 2.1.4 of the [SUPP] indicates that audit records for the Mobility Controller are stored locally and transmitted to the remote syslog server simultaneously. Furthermore, section 2.1.2 of the [SUPP] claims that logs for the Remote AP (RAP) are transferred to the Mobility Controller (MC) via the already-established secure channel between the RAP and MC as part of the provisioning process.

25 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.

Findings:	Section 2.1.4 of the [SUPP] states that the Mobility Controller and RAP are designed to use FIFO (First-In-First-Out) if storage exhaustion occurs. This option is not configurable.

3.1.4.3 Tests

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

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

High-Level Test Description
Review of audit data sent encrypted over the claimed channel is performed as part of FTP_ITC.1, test 3. The audit server version used is reported in section 2.1 of the DTR. Ensuring that the TOE is capable of transferring audit data successfully to the receiver is performed throughout the DTR and is evidenced in the DTR Evidence document.
Findings: PASS - The evaluator successfully configured the audit channel between the TOE and the remote logging server. Throughout test cases, the evaluator found the appropriate audit messages arrive to the remote logging server via the configure protected channel without administrator intervention. This data was not viewable in the clear. The audit server version used is reported in section 2.1 of the DTR.

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).

High-Level Test Description
Show the beginning of the security log file. Then, perform login operations 10 times while the security logging verbosity is set to debug (to maximize log entries). Then review the security log file again and show that the first message has rolled off.
Findings: PASS - After performing the test case, the evaluator found that the TOE successfully rolled off the oldest audit records as described in the [ST].

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

High-Level Test Description
FAU_STG_EXT.2/LocSpace is not claimed by the TOE.
Findings: Not applicable

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.

High-Level Test Description
Test 1 and Test 2 has been applied to all the TOE components.
Findings: PASS - Only the Aruba Mobility Controller locally stores and forwards audit data to an external audit server. The Remote Access Points (RAP) devices forward their audit data in real-time over the protected ITT channel to the Controller. The TOE does not claim FAU_STG_EXT.2/LocSpace for any component.

3.1.5 FAU_STG_EXT.4 Protected Local audit event storage for distributed TOEs & FAU_STG_EXT.5 Protected Remote audit event storage for Distributed TOEs

3.1.5.1 TSS

27 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.

Findings:	[ST] section 6.1.2: The TOE stores audit records locally on the Mobility Controller and can also be configured to send audit records to a trusted third-party syslog server in the operational environment. Audit records generated on the RAPs are sent to the Mobility Controller where they are stored and then forwarded to the remote syslog server. All transfers occur in real-time. Section 6.1.3 of the [ST] indicates that the transfer of audit records (from the RAPs) to the Mobility Controller occurs in real-time over an IPsec protected channel according to FTP_ITT.1.

28 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.

Findings:	Section 6.1.1 of the [ST] claims that the Remote Access Points (RAP) do not store any audit records, but rather forward all audit events securely over an IPsec connection to the Mobility Controller. Section 6.1.3 of the [ST]: The Remote Access Points buffer audit records in Flash for transmission to the Mobility Controller for storage. If the buffer becomes full, previous audit records are overwritten according to a first in, first out rule.

3.1.5.2 Guidance Documentation

29 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.

30 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.

Findings:	Section 2.1.2 of [SUPP] states “The Remote AP maintains locally stored audit records which are sent to the Controller during regular operation. This traffic is sent through the secure channel established between the Controller and AP and is established during the provisioning process without Security Administrator interaction.”

3.1.5.3 Tests

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

32 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.

High-Level Test Description
Test 1 has been combined with test 2 and test 3 (as appropriate).
Findings: PASS – The evaluator was able to confirm the generation of appropriately formed audit records during the execution of test 2 (for Controllers) and test 3 (for RAP devices).

33 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.

34 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.

High-Level Test Description
All components other than the controller distribute their records to the controller itself. The controller trusted channel for sending logging to a remote server has been tested in FAU_STG_EXT.1 and FTP_ITC.1.
Findings: PASS – The evaluator confirmed the required behaviour during FAU_STG_EXT.1 and FTP_ITC.1 which, combined, offers the same tests as requested here.

35 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.

High-Level Test Description
Examine the traffic between the RAP and the controller. Perform a task which results in an auditable message from the RAP back to the controller. Show that this audit message is not sent in the clear and is well-formed according to FAU_GEN.1.2.
Findings: PASS – The valuator confirmed that auditable tasks performed on the RAP device are not transmitted to the Controller in the clear.

36 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.

Findings:	The evaluator confirmed that the behaviour witnessed during these tests is consistent with the TSS and Guidance documentation.

3.2 Communication (FCO)

3.2.1 FCO_CPC_EXT.1 Component Registration Channel Definition

37 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/Join), and shall report the answers.

a) What stops[1] 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

[1] 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.

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/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?

3.2.1.1 TSS

38 (Note: [NDSD, 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.)

39 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/Join.

40 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/Join option in the main selection in FCO_CPC_EXT.1.2.

Findings:	[ST] section 6.2.1: Administrators must use the management interfaces (Web UI or CLI) of the Mobility Controller to manually enable/disable each RAP before it can be registered to the controller and provisioned.

	The FCO_CPC_EXT.1.2 selection is of the “first type”. The TSS in section 6.2.1 of the [ST] identifies FPT_ITT.1/Join as the relevant SFR iteration that specifies the channel used. The registration channel meets an iteration of FPT_ITT.1.

3.2.1.2 Guidance Documentation

41 (Note: [NDSD, 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.)

42 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).

Findings:	Section 2.3.1 of the [SUPP] provides the instructions to enable a Remote Access Point (RAP) device. The same section indicates that RAP devices can be disconnected from the distributed TOE solution by disabling the RAP user, VPN pool the RAP is a part of, the AP group the RAP is a part of, or to remove the RAP device from the associated whitelist. The [SUPP] in section 2.3.1 describes the method of disconnection as preventing the Controller from responding to the disabled component.

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

Findings:	In the [SUPP] section 2.3.1, there are no active steps that a security administrator would need to take. “If during the provisioning process the connection between the Controller and Remote AP is interrupted, the process will halt and would resume once connectivity is re-established.”

44 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/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 steadystate inter-component channel (as in FTP_ITC.1 or FPT_ITT.1)

Findings:	Section 2.3.1 claims that IPsec is used for both the initial join as well as the ongoing communications between the components. The IPsec connection is described in detail in section 2.2.6 of the [SUPP] and satisfies the requirements for meeting the requirements of 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)

Findings:	There are no obvious points of concern with respect to the use of IPsec for the registration vs. during subsequent inter-component communications.

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).

Findings:	The IPsec channel relies on the administrator to configure each component to meet the needs of their operational environment. Section 2.4.6.1 of the [SUPP] provides the instructions to use a provisioning-time Web UI on the RAP devices to load a custom certificate. Section 2.3.1 of [SUPP] states that RAP devices use a pre-configured unique RSA certificate by default which can be modified to be a custom certificate (using the instructions in section 2.4.6.1 of the [SUPP]).

45 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).

46 In the case of a distributed TOE for which the ST author uses the FTP_TRP.1/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.

Findings:	The [ST] does not claim the use of FTP_TRP.1/Join.

3.2.1.3 Tests

47 (Note: [NDSD, 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.)

48 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[2] that it is required to communicate with (non-equivalent TOE components are as defined in the minimum configuration for the distributed TOE)

[2] 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 A.9). 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.

High-Level Test Description
Show the results of FCO_CPC_EXT.1 test 2 and test 3 as upholding the testing for this AA.
Findings: PASS - Test 1.1 is conducted as part of test 2 and test 3 below. Test 2 shows that a RAP cannot communicate when it is not a member; test 3 shows that a RAP can commiunicate once it becomes a member. There is only one type of enablement process.

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.

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

High-Level Test Description
Given two RAP devices within the testing environment, neither of which is already enabled within the Mobility Controller, show that when one device is successfully enabled into the distributed TOE, other components that have not been explicitly enabled are still unable to communicate.
Findings: PASS – The evaluator showed that after enablement of a RAP device, the device could communicate with the Mobility Controller. Another unjoined RAP device remained unable to communicate with the Mobility Controller.

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.

High-Level Test Description
Remove a previously provisioned AP from the controller allowlist. Use the controller to reboot the AP and show that the controller no longer permits the AP to act as an associated AP.
Findings: PASS – The evaluator confirmed that once an AP is removed from the allow list of APs, the Controller will no longer respond 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/Join.

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

High-Level Test Description
The TOE claims the use of FPT_ITT.1. For the purposes of FCO_CPC_EXT.1, we perform the same test cases in FPT_ITT.1 in the context of the joining process. For the purposes of FPT_ITT.1, there are three tests cases which need to be conducted: (1) show that the connection can be set up successfully; (2) show that there is no plaintext information over the protected link; and (3) show that the link is resilient to both short- and long-term network outages. Starting with an unjoined RAP device, initiate a joining process with the controller and power on the AP. During the boot process, disconnect the AP from the network until the AP reboots due to running out of attempts. Confirm the controller has not yet accepted the RAP. Reconnect the AP to the controller as the device reboots. Once the AP starts to rejoin the controller and IPsec starts, disconnect the AP from the controller for 5 seconds to interrup the MAC layer and then reconnect. Show that the Remote AP continues to negotiate IPsec and does not transmit any plaintext information. Confirm the controller has accepted the RAP.
Findings: PASS – The evaluator confirmed that the registration channel meets the requirements of FPT_ITT.1 while attempting to join a RAP device to the Controller.

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 intercomponent communication, and in all cases where the second selection in FCO_CPC_EXT.1.2 is made (i.e. using FTP_TRP.1/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

High-Level Test Description
The registration channel is subsequently used for intercomponent communication and therefore this test case is N/A.
Findings: N/A

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).

High-Level Test Description
There is no operational guidance required to meet the requirements for a steady state intercomponent channel.
Findings: N/A

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.

High-Level Test Description
As part of the provisioning process, the RAP devices can be provisioned with a custom X.509 certificate set. This involves uploading both a trust anchor as well as constructing a CSR/certificate. For this test, upload the certificate and build the CSR.
Findings: PASS – The evaluator was able to provision a custom certificate to the RAP device and use it to communicate successfully with the Controller. The use of a custom certificate can enhance the security of the distributed TOE communications channels.

3.3 Cryptographic Support (FCS)

3.3.1 FCS_CKM.1 Cryptographic Key Generation

3.3.1.1 TSS

50 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.

Findings:	Section 6.3.1 of the [ST] states that for IPsec, SSH and TLS, the TOE supports cryptographic key generation for RSA schemes using key sizes of 2048 bits, ECC schemes using NIST curves P-256, P-384, and FFC schemes using key sizes of 2048 bits.

3.3.1.2 Guidance Documentation

51 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.

Findings:	Section 2.2.1 of the [SUPP] provides the instructions necessary to configure the TOE to construct keys that meet the requirements. The TOE must be configured to operate in FIPS mode – and instructions are provided to enable this mode of operation. Long-term key construction is required for TLS (as a server) and IPSec (as a peer). SSH host keys are constructed at initialization time and no configuration is needed to provision host keys of the correct algorithm or size. The size and parameters for TLS and IPsec keys are offered in section 2.2.1 of the [SUPP] and these meet the claims in the [ST].

3.3.1.3 Tests

52 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

53 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*.

54 Key Pair generation specifies five 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

55 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

56 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

57 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)

58 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.

59 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

60 and two ways to generate the cryptographic group generator g:

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

61 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.

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

63 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.

64 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

65 for each FFC parameter set and key pair.

NIAP TD0580

FFC Schemes using "safe-prime"

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

Findings:	The vendor uses CAVP certificates A2689 and A2690 for RSA and ECDSA key generation. These are described in [ST] Table 18.

3.3.2 FCS_CKM.2 Cryptographic Key Establishment

3.3.2.1 TSS

67 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.

Findings:	Section 6.3.3 of the [ST] claims that the TOE performs finite field-based key establishment using elliptic curve Diffie-Hellman for SSH, TLS, and IPsec. For IPsec, MODP group 14 is also used. The evaluator considers the description to

	unambiguously identify the SFR and service given that SSH and TLS are used for a single purpose, and IPsec to define the remainder of the services.

NIAP TD0580

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

Findings:	This activity was removed by TD0580

3.3.2.2 Guidance Documentation

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

Findings:	According to section 2.2.1 of the [SUPP], once FIPS mode has been enabled in the TOE, there is no further configuration required to permit the required algorithms.

3.3.2.3 Tests

Key Establishment Schemes

70 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

71 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

72 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.

73 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.

74 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.

75 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.

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

Validity Test

77 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.

78 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).

79 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.

Findings:	The vendor uses CAVP certificates A2689 and A2690 for EC key agreement schemes and CAVP A2689 for FFC schemes and safe-prime groups in RFC 3526. These are described in [ST] Table 18.

RSA-based key establishment schemes

80 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.

High-Level Test Description
The [ST] does not claim the use of RSA based key establishment schemes.
Findings: Not applicable

NIAP TD0580 Removed:

~~Diffie-Hellman Group 14~~

81 ~~The evaluator shall verify the correctness of the TSF's implementation of Diffie-Hellman group 14 by using a known good implementation for each protocol selected~~

~~in FTP_TRP.1/Admin, FTP_TRP.1/Join, FTP_ITC.1 and FPT_ITT.1 that uses Diffie-Hellman group 14.~~

High-Level Test Description
Removed per TD0580
Findings: Removed per TD0580

82 ***FFC Schemes using "safe-prime" groups***

83 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.

High-Level Test Description
Tests are conducted as part of the cryptographic protocol testing implemented for FTP_ITC.1, FTP_ITC.1/VPN and FPT_ITT.1.
Findings: PASS - The evaluator verified the TSF's implementation during testing FCS_IPSEC_EXT.1/VPN (for FTP_ITC.1/VPN and FTP_ITC.1) and FCS_IPSEC_EXT.1/ITT (for FPT_ITT.1). Only MODP-2048 (RFC3526 group 14) is claimed as an FFC safe prime group. This is supported by CAVP testing in certificate A2689 for FFC schemes and safe-prime groups in RFC 3526.

3.3.3 FCS_CKM.4 Cryptographic Key Destruction

3.3.3.1 TSS

84 The evaluator examines the TSS to ensure it lists all relevant keys (describing the origin and storage location of each), all relevant key destruction situations (e.g. factory reset or device wipe function, disconnection of trusted channels, key change as part of a secure channel protocol), and the destruction method used in each case. For the purpose of this Evaluation Activity the relevant keys are those keys that are relied upon to support any of the SFRs in the Security Target. The evaluator confirms that the description of keys and storage locations is consistent with the functions carried out by the TOE (e.g. that all keys for the TOE-specific secure channels and protocols, or that support FPT_APW.EXT.1 and FPT_SKP_EXT.1, are accounted for[3]). 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.

Findings:	Section 6.3.4 of the [ST] provides a table listing all relevant keys, their storage location, use cases and key destruction methods. Due to the size of the table, please see table in the ST for reference.

85 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).

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

Findings:	The table in section 6.3.4 of the [ST] states that almost all keys are stored in plaintext in non-volatile memory (listed in the table listed as "Key Usage" in the TSS under "Stored in flash memory") and are zeroized by issuing a "write erase all" command. For those that are not listed as being stored in plaintext, the "Storage" column of the table indicates they are encrypted using 3DES. From an evaluator perspective, 3DES is not an approved cryptographic algorithm for the claimed set of PPs. The efficacy of the algorithm implementation was not tested and therefore the key is not considered a CSP. The 3DES KEK is listed in the table as being used to "protect/obfuscate" the listed private keys. The TOE end-user should consider the private keys to be no better protected than "plain text" due to the limitations in assessing the 3DES implementation.

86 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.

Findings:	The [ST] in section 6.3.4 claims destruction of keys in non-volatile storage using an invocation of an interface command called "write erase all". Only one type of non-volatile media is described.

87 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.

Findings:	The [ST] in section 6.3.4 claims all keys are stored in plaintext form. Some private keys are considered to be "obfuscated" which carries the same implication as "plaintext".

88 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.

Findings:	No such configurations or circumstances are presented in the TSS.

89 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.

Findings:	The [ST] does not claim this selection.

3.3.3.2 Guidance Documentation

90 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.

91 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[4] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings:	Section 2.2.2 of the [SUPP] indicate there are no additional configuration actions needed to enforce FCS_CKM.4. Specifically for runtime CSPs, "…all CSPs will be zeroized automatically when no longer needed." Furthermore, to destroy long-term keys stored on flash media, commands are provided in the [SUPP] section 2.2.2 along with additional instructions needed to ensure their correct handling post-sanitization.

3.3.3.3 Tests

92 None

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

3.3.4.1 TSS

93 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.

Findings:	In the [ST] section 6.3.5, the TOE supports AES CBC, CTR and AES GCM (128 and 256 bits) for data encryption/decryption.

3.3.4.2 Guidance Documentation

94 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.

Findings:	According to section 2.2.3 of the [SUPP], once FIPS mode has been enabled in the TOE as well as the Advanced Cryptography license is installed, there is no further configuration required to permit the required algorithms.

3.3.4.3 Tests

AES-CBC Known Answer Tests

95 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

[4] 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).

evaluator shall compare the resulting values to those obtained by submitting the same inputs to a known good implementation.

96 **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.

97 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.

98 **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.

99 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.

100 **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].

101 To test the decrypt functionality of AES-CBC, the evaluator shall supply the two sets of key 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.

102 **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].

103 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

104 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.

105 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

106 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]
```

107 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.

108 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

109 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.

110 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.

111 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.

112 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

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

114 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.

115 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.

116 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.

117 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].

118 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

119 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

120 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]

121 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.

122 There is no need to test the decryption engine.

Findings:	The vendor uses CAVP certificates A2689 and A2690 for AES encryption and decryption. These are described in [ST] Table 18.

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

3.3.5.1 TSS

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

Findings:	Section 6.3.6 of the [ST] claims the TOE supports RSA (modulus 2048) and ECDSA with elliptical curve size 256 or 384 bits for signature generation and verification.

124

3.3.5.2 Guidance Documentation

125 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.

Findings:	According to section 2.2.3 of the [SUPP], once FIPS mode has been enabled in the TOE as well as the Advanced Cryptography license is installed, there is no further configuration required to permit the required algorithms.

3.3.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

126 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

127 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

128 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.

129 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

130 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.

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

Findings:	The vendor uses CAVP certificates A2688 (RSA verification only), A2689 (both ECDSA and RSA signature generation and verification), and A2690 (both ECDSA and RSA signature generation and verification). These are described in [ST] Table 18.

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

3.3.6.1 TSS

132 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.

Findings:	As per section 6.3.7 of the [ST], the SHA hash algorithm is used as part of HMAC, but is also used as part of RSA digital signature creation and verification.

3.3.6.2 Guidance Documentation

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

Findings:	According to section 2.2.3 of the [SUPP], once FIPS mode has been enabled in the TOE as well as the Advanced Cryptography license is installed, there is no further configuration required to permit the required algorithms.

3.3.6.3 Tests

134 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.

135 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

136 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

137 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

138 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 $1 \leq i \leq m$. 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

139 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

140 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.

Findings:	The vendor uses CAVP certificates A2688 (SHA-256 only), A2689, and A2690 for hash algorithm operations. These are described in [ST] Table 18.

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

3.3.7.1 TSS

141 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.

Findings: Section 6.3.8 of the [ST] claims the TOE's HMAC key length, hash function used, block size, and output MAC lengths (aka, the "digest size" in the table below. These are summarized in the following table:

Algorithm	Block Size	Key Size	Digest Size
HMAC-SHA-1	512 bits	160 bits	160 bits
HMAC-SHA-256	512 bits	256 bits	256 bits
HMAC-SHA-384	1024 bits	384 bits	384 bits

3.3.7.2 Guidance Documentation

142 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.

Findings:	According to section 2.2.3 of the [SUPP], once FIPS mode has been enabled in the TOE as well as the Advanced Cryptography license is installed, there is no further configuration required to permit the required algorithms.

3.3.7.3 Tests

143 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.

Findings:	The vendor uses CAVP certificates A2689 and A2690 for keyed-hash algorithm operations. These are described in [ST] Table 18.

3.3.8 FCS_HTTPS_EXT.1 HTTPS Protocol

3.3.8.1 TSS

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

Findings:	Within [ST] section 6.3.10 the TSS states that the TOE web GUI is accessed via an HTTPS connection. The TOE does not use HTTPS in a client capacity. The TOE's HTTPS protocol complies with RFC 2818. RFC 2818 specifies HTTP over TLS. The majority of RFC 2818 is spent on discussing practices for validating endpoint identities and how connections must be setup and torn down. The TOE web GUI operates on an explicit port designed to natively speak TLS: it does not attempt STARTTLS or similar multi-protocol negotiation which is described in section 2.3 of RFC 2818. The web server attempts to send closure Alerts prior to closing a connection in accordance with section 2.2.2 of RFC 2818.

3.3.8.2 Guidance Documentation

145 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.

Findings:	Section 2.2.4 of the [SUPP] indicates that no configuration is required. The TOE will function over HTTPS, compliant to RFC 2818, when operating in FIPS mode.

3.3.8.3 Tests

146 This test is now performed as part of FIA_X509_EXT.1/Rev testing.

147 Tests are performed in conjunction with the TLS evaluation activities.

148 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.

3.3.9 FCS_IPSEC_EXT.1/VPN IPsec Protocol

3.3.9.1 TSS

FCS_IPSEC_EXT.1.1/VPN

149 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.

150 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE implements an SPD and processes packets to satisfy the behavior of DISCARD, BYPASS, and PROTECT packet processing as described in RFC 4301 to determine what traffic gets protected with IPsec, what gets bypassed, and what gets dropped. Each packet is either PROTECTed using IPsec security services, DISCARDed, or allowed to BYPASS IPsec protection, based on the applicable SPD policies. The SPD is achieved via the routing table and firewall policies. The TOE administrator implicitly configures the IPsec SPD via the routing table and firewall policies. The TOE compares packets against the configured rules to determine if any of the packets match the rules. The packets can be matched based upon source IP address, destination IP address, protocol type (e.g., TCP, UDP, ICMP). Traffic not matching any rule is passed to the next stage of processing. The TOE includes a final rule that causes the network packet to be discarded if no other rules are matched.

FCS_IPSEC_EXT.1.3/VPN

151 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).

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE includes an implementation of IPsec in accordance with RFC 4301. The TOE supports IPsec for tunnel mode.

FCS_IPSEC_EXT.1.4/VPN

152 The evaluator shall examine the TSS to verify that the selected algorithms are implemented. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1/KeyedHash Cryptographic Operations (for keyed-hash message authentication) and if the SHA-based HMAC function truncated output is utilized it must also be described.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the IPsec ESP protocol is implemented in conjunction with AES-CBC-128 and AES-CBC-256 (as specified by RFC 3602) together with the following truncated versions of SHA-based HMAC algorithms: HMAC-SHA-1 and with AES-GCM-128 and AES-GCM-256 (as specified by RFC 4106). This is consistent with claims made in FCS_COP.1/KeyedHash for HMAC-SHA1. The TOE is claiming in section 6.3.11.1 of the [ST] use of a truncated version of SHA-based HMAC from 160-bits to 96-bits.

FCS_IPSEC_EXT.1.5/VPN

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

154 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE implements IKEv2.

FCS_IPSEC_EXT.1.6/VPN

155 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE uses the AES-CBC-128 and AES-CBC-256 algorithms as specified in RFC 3602 to encrypt the IKE payload.

FCS_IPSEC_EXT.1.7/VPN

156 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the lifetimes for IKEv2 SAs are established during configuration of the IKE policies via the CLI function by an authorized administrator and can be configured with 1-24 hours for the IKEv2 IKE_SA and within 1-8hrs for the IKEv2 IKE_CHILD SA. The TOE also supports volume-based rekeying for the IKEv2 IKE_CHILD SA.

FCS_IPSEC_EXT.1.8/VPN

157 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the lifetimes for IKEv2 CHILD_SA are established during configuration of the IKE policies via the CLI function by an authorized administrator and can be configured within 1-8hrs for the IKEv2 IKE_CHILD SA. The TOE also supports volume-based rekeying for the IKEv2 IKE_CHILD SA.

FCS_IPSEC_EXT.1.9/VPN

158 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in gx mod p) using the FIPS validated RBG specified in FCS_RBG_EXT.1 and having possible lengths of 112, 192 or 384 bits (for DH Groups 14, 19, and 20, respectively). The TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash.

FCS_IPSEC_EXT.1.10/VPN

159 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash.

160 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in gx mod p) using the FIPS validated RBG specified in FCS_RBG_EXT.1 and having possible lengths of 112, 192 or 384 bits (for DH Groups 14, 19, and 20, respectively). The TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash.

FCS_IPSEC_EXT.1.11/VPN

161 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE implements IKEv2, with support for Diffie-Hellman (DH) Groups 14, 19, and 20. In the IKEv2 IKE_SA and IKE_CHILD exchanges, the TOE and peer will agree on the best DH group both can support. When the TOE initiates IKE negotiation, the DH group is sent in order according to the peer's configuration. When the TOE receives an IKE proposal, it will select the first match and the negotiation will fail if there is no match.

FCS_IPSEC_EXT.1.12/VPN

162 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states the TOE checks to ensure the negotiated symmetric algorithm in the IKEv2 CHILD_SA is less than or equal to the strength of the IKEv2 IKE_SA.

FCS_IPSEC_EXT.1.13/VPN

163 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/SigGen Cryptographic Operations (for cryptographic signature).

Findings:	Within [ST] section 6.3.11.1 the TSS states that both RSA and ECDSA certificates are supported for IPsec authentication. This is consistent with selections made in FCS_COP.1/SigGen.

164 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.

Findings:	Pre-shared keys are not chosen as a selection.

FCS_IPSEC_EXT.1.14/VPN

165 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE will only establish a trusted IPsec channel if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following type: Distinguished Name (DN). Fields within the DN are not individually selectable; the DN must be an exact match for the entire DN string.

3.3.9.2 Guidance Documentation

FCS_IPSEC_EXT.1.1/VPN

166 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 – 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.

Findings:	Section 2.2.6.1 of [SUPP] indicates that the functionality required by the SPD can be achieved using a combination of TOE enforced firewall rules and packet routing Access Control Lists (ACLs). [ADMIN] discusses "Policy-Based Routing" (PBR) within section "Virtual Private Networks > Working with Site to Site VPNs > Adding ANY-ANY Crypto Map" and also in section ""Virtual Private Networks > Session ACL on IPsec Map" which uses ACLs to achieve the goal of meeting BYPASS and PROTECT SPD models. [CLI] using the CLI command "ip access-list route" offers the administrator a comprehensive

understanding of the mechanism with examples provided in [SUPP] sections 2.2.6.1 and 2.6.1.

For DISCARD, the TOE uses firewall rules. Firewall rules are described in summary within [SUPP] under sections 2.2.6.1 and 2.4 and also in [CLI] using the CLI command "ip access-list session" which indicates that an action of "deny" will reject packets.

The evaluator considered section 6.3.11 of the [ST] and found the TSS to be consistent with the guidance documentation.

The [SUPP] in section 2.2.6.1 provides a discussion of how rule ordering impacts the processing of an IP packet.

FCS_IPSEC_EXT.1.3/VPN

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

Findings:	Section 2.2.6.2 of [SUPP] indicates the only permitted operational mode for the TOE (meaning the RAP devices and the Mobility Controller) is tunnel mode.

FCS_IPSEC_EXT.1.4/VPN

168 The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Findings:	Section 2.2.6.3 of [SUPP] provides the commands needed to configure the appropriate ciphers using the "ipsec transform-set" CLI command.

FCS_IPSEC_EXT.1.5/VPN

169 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 (if selected).

Findings:	Section 2.2.6.4 of [SUPP] states "Only IKEv2 is supported by the TOE. NAT-T (NAT Traversal) is supported by Mobility Controllers and RAP to transport packets over UDP port 4500 rather than using IPsec native encapsulation." To configure NAT-T, this can be done using the CLI as described in section 2.2.6.4 of the [SUPP].

170 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.

Findings:	The TOE does not claim use of IKEv1.

FCS_IPSEC_EXT.1.6/VPN

171 The evaluator ensures that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Findings:	Section 2.2.6.5 of [SUPP] provides the commands needed to enable IKEv2 with the cryptographic algorithms selected by the requirement.

FCS_IPSEC_EXT.1.7/VPN

NIAP TD0633

172 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.

Findings:	Section 2.2.6.6 of [SUPP] provides the configuration instructions to enable time-based rekey lifetimes for IKE SA between the Mobility Controller and broader VPN peers in the IT environment. The [SUPP] also claims that a 24-hour IKE SA key lifetime is the default value, and it can be explicitly reconfigured (between 300-86400 seconds) if it has been changed from the default. **NOTE**: This finding applies only to IKE SA lifetimes between the Mobility Controller and VPN peers in the broader IT environment. For information on how IKE SA lifetimes are treated between the MC and the RAP devices, please refer to section 3.3.10.2 within this document.

FCS_IPSEC_EXT.1.8/VPN

NIAP TD0633

173 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 8hours. 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.

Findings:	Section 2.2.6.6 of [SUPP] provides the configuration instructions to enable both time-based and volume-based rekey lifetimes for CHILD SA between the Mobility Controller and broader VPN peers in the IT environment. The [SUPP] also provides an example of setting an 8-hour (28,800 seconds) CHILD SA key lifetime. **NOTE**: This finding applies only to CHILD SA lifetimes between the Mobility Controller and VPN peers in the broader IT environment. For information on how CHILD SA lifetimes are treated between the MC and the RAP devices, please refer to section 3.3.10.2 within this document.

FCS_IPSEC_EXT.1.11/VPN

174 The evaluator ensures that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Findings:	Section 2.2.6.8 in [SUPP] supports DH groups 14, 19, and 20. The configuration of these groups are provided in the [SUPP].

FCS_IPSEC_EXT.1.13/VPN

175 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.

Findings:	[SUPP] section 2.2.6.9 provides a reasonable summary of configuring the TOE to use X.509 certificates with public key and signature algorithms RSA and ECDSA.

176 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.

Findings:	The [ST] does not claim the use of pre-shared keys.

177 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”.

Findings:	[SUPP] section 2.2.6.9 provides the necessary guidance documentation to ensure that a trusted CA can be loaded into the TOE’s trust store. Additional information on the use of an X.509 trust store is found in section 2.4.6 of [SUPP].

FCS_IPSEC_EXT.1.14/VPN

178 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.

Findings:	Section 2.2.6.10 provides the necessary CLI instructions to set the peer DN expected to be contained within the peer’s certificate. Only distinguished names are supported. An important note regarding how to identify the DN is provided in section 2.4.6 of [SUPP]. *“Note: It may be difficult to determine the exact DN to configure, simply by looking at a peer's certificate. Attempting to establish an IPsec tunnel while examining the log file (possibly after enabling "logging level debugging security") will generally show the exact DN string that must be configured, once it is received from the peer.*” Section 2.2.6.10 in the [SUPP] also explicitly states that SAN types are not supported.

3.3.9.3 Tests

FCS_IPSEC_EXT.1.1/VPN

179 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.

High-Level Test Description
Encryption of packets using the VPN is exemplified in FTP_ITC.1/VPN. Plaintext flowing of packets for a channel which can also optionally traverse the VPN is shown for NTP in FCS_NTP_EXT.1 test cases. This test case will specifically show that traffic can be dropped by configuring an IP ACL to drop specific traffic.
Findings: PASS - The evaluator confirmed that TOE correctly forwards packets unencrypted, tunnels packets through the VPN, or drops packets based on the configured rules.

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.

High-Level Test Description
Configure a firewall ACL to drop TCP port 22 packets coming from anywhere. Attempt to SSH to the TOE directly and via the Ipsec network and show access is denied in both cases. Adjust the ACL to permit TCP port 22 packets coming directly from the Ipsec peer IP, but denied from everywhere else. Show SSH access works. Adjust the ACL to then deny TCP port 22 packets coming from the Ipsec network destination, but in such a way that there is an overlapping network segment with the previous permit rule encompassing the peer's IP. Show that SSH access is no longer permitted from the Ipsec network. Reorder the rules and show that SSH access from the Ipsec network is now permitted.
Findings: PASS – The evaluator confirmed that for each scenario the expected behaviour is exhibited and is consistent with both the TSS and guidance documentation.

FCS_IPSEC_EXT.1.2/VPN

180 The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

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

182 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 created” final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was dropped.

High-Level Test Description
Configure an IP ACL to allow SSH traffic from the AP-GW VM, encrypt SSH traffic from the Services VM, and drop all other SSH traffic. Log in using SSH from the AP GW VM, and attempt to log in using SSH from the Sniffer VM.
Findings: PASS – The evaluator confirmed the packet matching the rule to allow the packet was sent in plaintext, and the packet that did not match any of the configured rules was dropped.

FCS_IPSEC_EXT.1.3/VPN

183 The evaluator shall perform the following test(s) based on the selections chosen:

a) 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.

High-Level Test Description
Using the IPsec connection needed for the protection of the remote audit messages, show that the connection is successful when using a permitted cipher proposal in tunnel mode.
Findings: PASS – The evaluator confirmed that a successful connection was established using the tunnel mode.

b) 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.

High-Level Test Description
The TOE does not claim transport mode.
Findings: N/A

FCS_IPSEC_EXT.1.4/VPN

184 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.

High-Level Test Description
For each of the given encryption algorithms, configure the TOE to successfully establish an IPSec tunnel with a test system. For each of the given integrity algorithms, configure the TOE to successfully establish an IPSec tunnel with a test system. In all cases, ensure that the IKE_SA proposals are set to AES-CBC-256 to ensure phase strength compatibility.
Findings: PASS – The evaluator confirmed that a successful connection was established using each of the supported algorithms.

FCS_IPSEC_EXT.1.5/VPN

185 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.

High-Level Test Description
The TOE does not claim IKEv1.
Findings: N/A

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.

High-Level Test Description
Configure a non-TOE interface to hide behind a NAT. Configure the VPN tunnel between the non-TOE entity and the TOE to communicate over the NAT'd IP. Show that the tunnel can be established by traversing the NAT.
Findings: PASS – The evaluator confirmed that the IPsec connection was successfully established by traversing the NAT.

FCS_IPSEC_EXT.1.6/VPN

186 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.

High-Level Test Description
For each of the given encryption algorithms, configure the TOE to successfully establish an IPSec tunnel with a test system. In all cases, ensure that the CHILD_SA proposals are set to the same key strength to ensure phase strength compatibility.

High-Level Test Description
Findings: PASS – The evaluator confirmed that a successful connection with each of claimed IKEv2 algorithms and HMAC functions.

FCS_IPSEC_EXT.1.7/VPN

187 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."

188 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 an 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.

High-Level Test Description
The TOE does not claim "number of bytes" for IKE_SA lifetimes.
Findings: N/A

NIAP TD0633

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.

High-Level Test Description
Configure the TOE to rekey after 24 hours. Wait 24 hours and show that the tunnel rekeys IKE_SA before 24 hours has elapsed and CHILD_SA before each 8 hours has elapsed in the same period.
Findings: PASS – The evaluator confirmed that the TOE rekeyed the IKE_SA before 24 hours had elapsed.

FCS_IPSEC_EXT.1.8/VPN

189 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."

190 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 an 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.

High-Level Test Description
Configure the TOE to rekey after 1 MB of data. Transfer about 1MB of data and show that the system rekeys before 1 MB has been delivered.
Findings: PASS – The evaluator confirmed that after no more than the number of bytes configured the TOE rekeyed the CHILD_SA.

NIAP TD0633

b) 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.

High-Level Test Description
This test was conducted simultaneously with FCS_IPSEC_EXT.1.7/VPN test 2.
Findings: PASS – The evaluator confirmed the TOE rekeyed the CHILD_SA (Phase 2 SA) before 8 hours had elapsed.

FCS_IPSEC_EXT.1.10/VPN

191 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.

High-Level Test Description
This is a TSS item.
Findings: PASS - The DRBG described in FCS_RBG_EXT.1 is used to randomly generate each nonce used in IKEv2 exchanges according to the security strength associated with the negotiated DH group—128 bits for Groups 14 and 19, and 192 bits for Group 20. The nonces generated are at least half the output size of the negotiated pseudorandom function (PRF) hash.

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.

High-Level Test Description
This is a TSS item: Second selection not chosen.
Findings: N/A

FCS_IPSEC_EXT.1.11/VPN

192 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.

High-Level Test Description
For each of the given DH groups under IKEv2, configure the TOE to successfully establish an IPSec tunnel with a test system.
Findings: PASS – The evaluator confirmed that all supported IKE protocols were successfully completed using each of the claimed DH groups.

FCS_IPSEC_EXT.1.12/VPN

193 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.

High-Level Test Description
All ciphers and claimed IKE hash functions were tested as part of FCS_IPSEC_EXT.1.6/VPN test 1. The TOE only claims IKEv2.
Findings: PASS – The evaluator confirmed in FCS_IPSEC_EXT.1.6/VPN test 1 that the TOE successfully established an IKE connection using each of the claimed IKE algorithms.

b) Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an 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.

High-Level Test Description
Configure each peer to use IKE encryption of 128-bits and ESP encryption of 256-bits and attempt to establish a session. The session should fail to be established.
Findings: PASS – The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to select an encryption algorithm with more strength than that being used for the IKE SA.

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.

High-Level Test Description
Configure the non-TOE peer to use an IKE encryption algorithm and hash function which are not claimed and show that the session cannot be established.
Findings: PASS - The evaluator confirmed the TOE will not establish an IKE connection when the peer attempts to use unsupported algorithms.

d) Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an 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.

High-Level Test Description
Configure the non-TOE peer to use an ESP encryption algorithm which is not claimed and show that the session cannot be established. A known-good IKE algorithm ciphersuite will be used to ensure that it fails at CHILD_SA, instead of IKE_SA.
Findings: PASS - The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to use unsupported algorithms.

FCS_IPSEC_EXT.1.13/VPN

194 For efficiency sake, the testing that is performed may be combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

FCS_IPSEC_EXT.1.14/VPN

195 For each 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.

196 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.

High-Level Test Description
The TOE does not claim the use of CN/identifier types. The TOE only claims the use of Distinguished Name.

High-Level Test Description
Findings: N/A

- 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.

High-Level Test Description
The TOE does not claim the use of SAN/identifier types. The TOE only claims the use of Distinguished Name.
Findings: N/A

- Test 3: (conditional) For each CN/identifier type combination selected, the evaluator shall:

e) Create a valid certificate with the CN so it contains the valid identifier followed by '\0'. 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.

High-Level Test Description
The TOE does not claim the use of CN/identifier types. The TOE only claims the use of Distinguished Name.
Findings: N/A

f) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the CN without the '\0' and verify that IKE authentication fails.

High-Level Test Description
The TOE does not claim the use of CN/identifier types. The TOE only claims the use of Distinguished Name.
Findings: N/A

- 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.

High-Level Test Description
The TOE does not claim the use of SAN/identifier types. The TOE only claims the use of Distinguished Name.
Findings: N/A

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.

High-Level Test Description
The TOE does not claim the use of SAN/identifier types. The TOE only claims the use of Distinguished Name.
Findings: N/A

- 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.

High-Level Test Description
This test is conducted in FIA_X509_EXT.1.1/Rev test 1a and test 1b showing that the DN is being used as part of the configuration.
Findings: PASS – The evaluator confirmed in FIA_X509_EXT.1.1/Rev test 1a and test 1b that the TOE successfully authenticates the peer when the peer uses a certificate with a matching DN.

- 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.

High-Level Test Description
Construct a certificate which has two identical CN RDNs. Attempt to establish a VPN tunnel and show that it fails.
Findings: PASS – The evaluator confirmed that the IKE authentication failed when presented with an invalid certificate (duplicate CN).

b) Append '\0' to a non-CN field of an otherwise authorized DN.

High-Level Test Description
Construct a certificate which has a NULL character appended to a non-CN RDN string. Attempt to establish a VPN tunnel and show that it fails.
Findings: PASS – The evaluator confirmed that the IKE authentication failed when presented with an invalid certificate (null character in the OU RDN field).

3.3.10 FCS_IPSEC_EXT.1/ITT IPsec Protocol

3.3.10.1 TSS

FCS_IPSEC_EXT.1.1/ITT

197 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.

198 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.

Findings:	For FCS_IPSEC_EXT.1/ITT, section 6.3.11.2 claims that the SPD is implemented similarly to the description in section 6.3.11.1, with the exception that the SPD for inter-TOE connections applies to all traffic and enforces a route-based SPD for inter-TOE connections. It enforces either “default PROTECT” or “BYPASS only to facilitate IKE traffic” operations. Based on the description in section 6.3.11.2 of the [ST], the SPD is relatively trivial in this case, since it has a “default PROTECT” stance with one exception provided for BYPASS traffic to facilitate IKE traffic.

FCS_IPSEC_EXT.1.3/ITT

199 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).

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE includes an implementation of IPsec in accordance with RFC 4301. The TOE supports IPsec for tunnel mode. No differences are presented in section 6.3.11.2 of the [ST].

FCS_IPSEC_EXT.1.4/ITT

200 The evaluator shall examine the TSS to verify that the selected algorithms are implemented. In addition, the evaluator ensures that the SHA-based HMAC algorithm conforms to the algorithms specified in FCS_COP.1/KeyedHash Cryptographic

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

Findings:	Within [ST] section 6.3.11.2 the TSS states that the IPsec ESP protocol algorithms are implemented contingent on the type of certificate which is implemented by the connected RAP device. For RSA certificates, phase 2 will negotiate AES-CBC keys of size 128- or 256-bits and HMAC-SHA-1. For ECDSA certificates, phase 2 will negotiate AES-GCM keys of size 128- or 256 bits depending on the size of the elliptic curve algorithm claimed in the certificate's public key. The HMAC is consistent with claims made in FCS_COP.1/KeyedHash for HMAC-SHA1. The TOE is claiming in section 6.3.11.1 of the [ST] use of a truncated version of SHA-based HMAC from 160-bits to 96-bits.

FCS_IPSEC_EXT.1.5/ITT

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

202 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE implements IKEv2. No differences are presented in section 6.3.11.2 of the [ST].

FCS_IPSEC_EXT.1.6/ITT

203 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.

Findings:	Within [ST] section 6.3.11.2 the TSS states that the IPsec IKEv2 protocol uses the AES-CBC-128 and AES-CBC-256 algorithms as specified in RFC 3602 to encrypt the IKE payload.

FCS_IPSEC_EXT.1.7/ITT

204 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the lifetimes for IKEv2 SAs are established during configuration of the IKE policies via the CLI function by an authorized administrator and can be configured with 1-24 hours for the IKEv2 IKE_SA.

FCS_IPSEC_EXT.1.8/ITT

205 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the lifetimes for IKEv2 CHILD_SA are established during configuration of the IKE policies via the CLI function by an authorized administrator and can be configured within 1-8hrs for the IKEv2

	IKE_CHILD SA. Section 6.3.11.2 also states that the RAP devices only support time-based rekeying limits.

FCS_IPSEC_EXT.1.9/ITT

206 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in gx mod p) using the FIPS validated RBG specified in FCS_RBG_EXT.1 and having possible lengths of 112, 192 or 384 bits (for DH Groups 14, 19, and 20, respectively). The TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash. No differences are presented in section 6.3.11.2 of the [ST].

FCS_IPSEC_EXT.1.10/ITT

207 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash. No differences are presented in section 6.3.11.2 of the [ST].

208 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE generates the secret value x used in the IKEv2 Diffie-Hellman key exchange ('x' in gx mod p) using the FIPS validated RBG specified in FCS_RBG_EXT.1 and having possible lengths of 112, 192 or 384 bits (for DH Groups 14, 19, and 20, respectively). The TOE generates nonces used in the IKEv2 exchanges of length 112 bits, 128 bits and 192 bits and at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash. No differences are presented in section 6.3.11.2 of the [ST].

FCS_IPSEC_EXT.1.11/ITT

209 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.

Findings:	Within [ST] section 6.3.11.2 the TSS states that DH groups for IKEv2 are implemented contingent on the type of certificate which is implemented by the connected RAP device. For RSA certificates, the RAP device will only negotiate DH group 14. For ECDSA certificates with NIST P-256, the RAP device will negotiate DH group 19; for NIST P-384, the RAP device will negotiate DH group 20.

FCS_IPSEC_EXT.1.12/ITT

210 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.

Findings:	Within [ST] section 6.3.11.1 the TSS states that the TOE checks to ensure the negotiated symmetric algorithm in the IKEv2 CHILD_SA is less than or equal to the strength of the IKEv2 IKE_SA. No differences are presented in section 6.3.11.2 of the [ST].

FCS_IPSEC_EXT.1.13/ITT

211 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/SigGen Cryptographic Operations (for cryptographic signature).

Findings:	Within [ST] section 6.3.11.1 the TSS states that both RSA and ECDSA certificates are supported for IPsec authentication. This is consistent with selections made in FCS_COP.1/SigGen.

212 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.

Findings:	Pre-shared keys are not chosen as a selection.

FCS_IPSEC_EXT.1.14/ITT

213 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.

Findings:	Within [ST] section 6.3.11.2 the TSS states that the RAP devices will only establish a trusted IPsec channel to the controller if the presented CN identifier in the received certificate matches the configured reference identifier. The CN must be an IP address only.

3.3.10.2 Guidance Documentation

FCS_IPSEC_EXT.1.1/ITT

214 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 – 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.

Findings:	Section 2.2.6.1 of the [SUPP] states "RAP devices do not support the ability to configure an SPD and all traffic transits the IPsec tunnel." This is expected since the RAP devices are not general-purpose IPsec peers and are only designed to operate with the Mobility Controller.

FCS_IPSEC_EXT.1.3/ITT

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

Findings:	Section 2.2.6.2 of [SUPP] indicates the only permitted operational mode for the TOE (meaning the RAP devices and the Mobility Controller) is tunnel mode.

FCS_IPSEC_EXT.1.4/ITT

216 The evaluator checks the guidance documentation to ensure it provides instructions on how to configure the TOE to use the algorithms selected.

Findings:	Section 2.2.6.3 of [SUPP] states that ESP algorithms are contingent on the certificate presented and used for the connection to the RAP device. "*For RSA, AES-CBC-128/256 is used. For ECDSA, AES-GCM-128/256 is used with NIST P-256 and AES-GCM-256 is used with NIST P-384*."

FCS_IPSEC_EXT.1.5/ITT

217 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 (if selected).

Findings:	Section 2.2.6.4 of [SUPP] states "Only IKEv2 is supported by the TOE. NAT-T (NAT Traversal) is supported by Mobility Controllers and RAP to transport packets over UDP port 4500 rather than using IPsec native encapsulation." For RAP connections, the RAP device must be an initiator and therefore the Mobility Controller is a responder. Section 2.2.6.4 of [SUPP] clarifies that "For inbound connections where the controller is the IKE responder, NAT-T is supported by default."

218 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.

Findings:	The TOE does not claim use of IKEv1.

FCS_IPSEC_EXT.1.6/ITT

219 The evaluator ensures that the guidance documentation describes the configuration of all selected algorithms in the requirement.

Findings:	Section 2.2.6.3 of [SUPP] provides the commands needed to enable IKEv2 with the cryptographic algorithms selected by the requirement for the dynamic map used by RAP devices.

FCS_IPSEC_EXT.1.7/ITT

NIAP TD0633

220 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.

Findings:	Section 2.2.6.6 of [SUPP] provides the configuration instructions to enable time-based rekey lifetimes for IKE SA for the RAP channel. The [SUPP] also claims that a 28800-second IKE SA key lifetime is the default value for Mobility Controller to RAP devices, and it can be explicitly reconfigured. RAP devices will always initiate a rekey at 7200 seconds for CHILD SA and 28800 seconds for IKE_SA. This behavior occurs even when the Mobility Controller sets a higher rekey limit for RAP connections. If the Mobility Controller sets a limit lower than these thresholds, then the controller will initiate a rekey. **NOTE**: This finding only applies to IKE SA lifetimes between the Mobility Controller and RAP devices. For information on how IKE SA lifetimes are treated between the MC and broader VPN peers in the IT environment, please refer to section 3.3.9.2 within this document.

FCS_IPSEC_EXT.1.8/ITT

NIAP TD0633

221 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 8hours. 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.

Findings:	Section 2.2.6.6 of [SUPP] provides the configuration instructions to enable time-based rekey lifetimes for CHILD SA between the Mobility Controller and RAP devices. The [SUPP] also provides an example of setting an 8-hour (28,800 seconds) CHILD SA key lifetime. RAP devices will always initiate a rekey at 7200 seconds for CHILD SA and 28800 seconds for IKE_SA. This behavior occurs even when the Mobility Controller sets a higher rekey limit for RAP connections. If the Mobility Controller sets a limit lower than these thresholds, then the controller will initiate a rekey. **NOTE**: This finding only applies to CHILD SA lifetimes between the Mobility Controller and RAP devices. For information on how IKE SA lifetimes are treated between the

	MC and broader VPN peers in the IT environment, please refer to section 3.3.9.2 within this document.

FCS_IPSEC_EXT.1.11/ITT

222 The evaluator ensures that the guidance documentation describes the configuration of all algorithms selected in the requirement.

Findings:	Section 2.2.6.8 in [SUPP] supports DH groups 14, 19, and 20. DH groups used are dependent on the certificate in use for the RAP device. For RSA certificates, Group 14 is used. For ECDSA NIST P-256 certificates Group 19 is used, and for ECDSA NIST P-384 certificates Group 20 is used.

FCS_IPSEC_EXT.1.13/ITT

223 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.

Findings:	[SUPP] section 2.2.6.9 provides a reasonable summary of configuring the TOE to use X.509 certificates with public key and signature algorithms RSA and ECDSA.

224 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.

Findings:	The [ST] does not claim the use of pre-shared keys.

225 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”.

Findings:	[SUPP] section 2.2.6.9 provides the necessary guidance documentation to ensure that a trusted CA can be loaded into the TOE’s trust store. Additional information on the use of an X.509 trust store is found in section 2.4.6 of [SUPP].

FCS_IPSEC_EXT.1.14/ITT

226 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.

Findings:	Section 2.2.6.10 provides the necessary CLI instructions to set the peer DN expected to be contained within the peer’s certificate. For RAP devices, only IP addresses in the CN are supported. Note that while the [SUPP] section 2.2.6.10 claims that the RAP CN IP address is set using a CLI command called “*cert-DN <ip_address>*”, this has been confirmed with the developer to be a misnamed command. Only the CN is checked, rather than the entire DN. Section 2.2.6.10 in the [SUPP] also explicitly states that SAN types are not supported.

3.3.10.3 Tests

FCS_IPSEC_EXT.1.1/ITT

227 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.

High-Level Test Description
Send a ping packet from the AP with various IP addresses to the controller and show that it is tunneled through the ITT Ipsec tunnel regardless of the traffic properties.
Findings: PASS – The evaluator showed that traffic will be delivered over the protected channel by default.

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.

High-Level Test Description
There are no additional scenarios; packets are PROTECTed if they need to exit the RAP device.
Findings: PASS – The evaluator showed that traffic will be delivered over the protected channel by default.

FCS_IPSEC_EXT.1.2/ITT

228 The assurance activity for this element is performed in conjunction with the activities for FCS_IPSEC_EXT.1.1.

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

230 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 created” final entry that discards packets that do not match any previous entries). The evaluator sends the packet, and observes that the packet was dropped.

High-Level Test Description
Using a debug interface, manipulate the RAP route-based SPD to show that traffic can meet the requirements for PROTECT, BYPASS, and DISCARD.
Findings: PASS – The evaluator, using a debug interface, was able to show that the RAP device is capable of implementing an SPD that meets the requirements for PROTECT, BYPASS, and DISCARD. A default DISCARD rule is capable of being implemented within the debug interface.

FCS_IPSEC_EXT.1.3/ITT

231 The evaluator shall perform the following test(s) based on the selections chosen:

a) 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.

High-Level Test Description
Connect the RAP to the controller. The RAP and controller will only negotiate tunnel mode.
Findings: PASS – The evaluator confirmed that a successful connection was established using the tunnel mode.

b) 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.

High-Level Test Description
The TOE does not claim transport mode.
Findings: N/A

FCS_IPSEC_EXT.1.4/ITT

232 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.

High-Level Test Description
For each of the given ESP encryption algoroithms, configure the RAP dynamic map to make use of the given algorithm and show that the connection is successful and ESP traffic passes.
Findings: PASS – The evaluator confirmed that a successful connection was established using each of the supported algorithms.

FCS_IPSEC_EXT.1.5/ITT

233 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.

High-Level Test Description
The TOE does not claim IKEv1.
Findings: N/A

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.

High-Level Test Description
Using the RAP device configured behind a NAT'ing gateway, show that the RAP is capable of connecting to the controller through this NAT device.
Findings: PASS – The evaluator confirmed that the IPsec connection was successfully established by traversing the NAT.

FCS_IPSEC_EXT.1.6/ITT

234 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.

High-Level Test Description
For each of the given encryption algorithms, configure the Controller to successfully establish an IPSec tunnel with RAP and show that IKE was established with the correct encryption and integrity algorithms.
Findings: PASS – The evaluator confirmed that a successful connection with each of claimed IKEv2 algorithms and HMAC functions.

FCS_IPSEC_EXT.1.7/ITT

235 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."

236 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 an 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.

High-Level Test Description
The TOE does not claim "number of bytes" for IKE_SA lifetimes.
Findings: N/A

NIAP TD0633

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.

High-Level Test Description
Configure the controller to rekey after 24 hours. Wait 24 hours and show that the RAP <-> controller tunnel rekeys IKE_SA before 24 hours has elapsed and CHILD_SA before each 8 hours has elapsed in the same period.
Findings: PASS – The evaluator confirmed that the TOE rekeyed the IKE_SA before 24 hours had elapsed.

FCS_IPSEC_EXT.1.8/ITT

237 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."

238 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 an 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.

High-Level Test Description
The TOE does not claim “number of bytes” for CHILD_SA lifetimes.
Findings: N/A

NIAP TD0633

b) 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.

High-Level Test Description
This test was conducted simultaneously with FCS_IPSEC_EXT.1.7/ITT test 2.
Findings: PASS – The evaluator confirmed the TOE rekeyed the CHILD_SA (Phase 2 SA) before 8 hours had elapsed.

FCS_IPSEC_EXT.1.10/ITT

239 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.

High-Level Test Description
This is a TSS item.
Findings: PASS - The DRBG described in FCS_RBG_EXT.1 is used to randomly generate each nonce used in IKEv2 exchanges according to the security strength associated with the negotiated DH group—128 bits for Groups 14 and 19, and 192 bits for Group 20. The nonces generated are at least half the output size of the negotiated pseudorandom function (PRF) hash.

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.

High-Level Test Description
This is a TSS item: Second selection not chosen.
Findings: N/A

FCS_IPSEC_EXT.1.11/ITT

240 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.

High-Level Test Description
For each of the given DH groups under IKEv2, configure the TOE to successfully establish an IPSec tunnel with a test system.
Findings: PASS – The evaluator confirmed that all supported IKE protocols were successfully completed using each of the claimed DH groups.

FCS_IPSEC_EXT.1.12/ITT

241 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.

High-Level Test Description
All ciphers and claimed IKE hash functions were tested as part of FCS_IPSEC_EXT.1.6/ITT test 1. The TOE only claims IKEv2.
Findings: PASS – The evaluator confirmed in FCS_IPSEC_EXT.1.6/ITT test 1 that the TOE successfully established an IKE connection using each of the claimed IKE algorithms.

b) Test 2: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an 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.

High-Level Test Description
Configure the controller to select an IKEv2 encryption algorithm of AES-CBC-128 with an IPsec encryption algorithm of AES-CBC-256 while connecting to a RAP device. Show that the configuration fails to establish a working tunnel.
Findings: PASS – The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to select an encryption algorithm with more strength than that being used for the IKE SA.

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.

High-Level Test Description
Configure the controller to select an IKEv2 encryption algorithm of AES-CBC-192. Show that the configuration fails to establish a working tunnel. Ensure that the CHILD_SA is configured for AES-CBC-128 to ensure that failures are not due to a mismatch in IKE/IPsec strength. Then configure the controller to select IKEv2 AES-CBC-256, but a hash function of SHA1-96. Show that the configuration fails to establish a working tunnel.
Findings: PASS - The evaluator confirmed the TOE will not establish an IKE connection when the peer attempts to use unsupported algorithms.

d) Test 4: This test shall be performed for each version of IKE supported. The evaluator shall attempt to establish an 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.

High-Level Test Description
Configure the controller to attempt to use AES-128-GCM for the RAP RSA device. RAP devices configured with RSA certificates cannot make use of the AES-GCM algorithm set. Show that the IKEv2 SA can be set up, but the CHILD_SA cannot.
Findings: PASS - The evaluator confirmed the TOE will not establish an ESP connection when the peer attempts to use unsupported algorithms.

FCS_IPSEC_EXT.1.13/ITT

242 For efficiency sake, the testing that is performed may be combined with the testing for FIA_X509_EXT.1, FIA_X509_EXT.2 (for IPsec connections), and FCS_IPSEC_EXT.1.1.

FCS_IPSEC_EXT.1.14/ITT

243 For each 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.

244 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.

High-Level Test Description
Configure the Mobility Controller with a valid certificate. Show that the CN has been configured on the device. Show that the connection succeeds.
Findings: PASS – The evaluator confirmed that when the CN is set to an expected value, the RAP devices will successfully connect to the Controller.

- 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.

High-Level Test Description
The TOE does not claim the use of SAN/identifier types.
Findings: N/A

- Test 3: (conditional) For each CN/identifier type combination selected, the evaluator shall:

e) Create a valid certificate with the CN so it contains the valid identifier followed by '\0'. 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.

f) Configure the peer's reference identifier on the TOE (per the administrative guidance) to match the CN without the '\0' and verify that IKE authentication fails.

High-Level Test Description
Construct a certificate with a NULL character as the last character in the CN. Attempt to communicate to a RAP device from the controller. Show that the certificate is rejected.
Findings: PASS – The evaluator confirmed that the IKE authentication failed when presented with an invalid certificate (null character in the CN RDN field).

- 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.

High-Level Test Description
The TOE does not claim the use of SAN/identifier types.
Findings: N/A

- 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.

High-Level Test Description
The TOE does not claim the use of DN identifier types.
Findings: N/A

- 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:

c) Duplicate the CN field, so the otherwise authorized DN contains two identical CNs.

High-Level Test Description
The TOE does not claim the use of DN identifier types.
Findings: N/A

d) Append '\0' to a non-CN field of an otherwise authorized DN.

High-Level Test Description
The TOE does not claim the use of DN identifier types.
Findings: N/A

3.3.11 FCS_NTP_EXT.1 NTP Protocol

3.3.11.1 TSS

FCS_NTP_EXT.1.1

245 The evaluator shall examine the TSS to ensure 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.

Findings:	[ST] section 6.3.12: The TOE supports NTP v4. The TOE updates its system time using either IPsec to provide trusted communication between itself and an NTP time source; or using SHA1 pre-shared keys as the message digest algorithm to ensure integrity. The TOE does not update NTP timestamps from Broadcast and multicast addresses. The use of IPsec and SHA-1 message digest algorithm ensures 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.

3.3.11.2 Guidance Documentation

FCS_NTP_EXT.1.1

246 The evaluator shall examine the guidance documentation to ensure it provides the 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.

Findings:	According to [SUPP] section 2.2.5, the TOE supports the use of NTPv4 without additional configuration. Multiple NTP servers are configured using the "ntp server" CLI instruction. NTP authentication is configured using the "ntp authentication-key" CLI command and ensuring the use of "sha1" as the key hashing algorithm. When configuring NTP to route through IPsec tunnels, section 2.2.5 of [SUPP] informs the administrator to specify the IP address of an NTP server that is allowed within the scope of a configured IPsec policy.

FCS_NTP_EXT.1.2

247 For each of the secondary selections made in the ST, the evaluator shall examine the guidance document to ensure it instructs the 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).

Findings:	According to [SUPP] section 2.2.5, the TOE supports the use of SHA1 key authentication and/or use of IPsec as a trusted channel. NTP authentication is configured using the "ntp authentication-key" CLI command and ensuring the use of "sha1" as the key hashing algorithm. When configuring NTP to route through IPsec tunnels, section 2.2.5 of [SUPP] informs the administrator that they need to specify "*…the IP address of an NTP server that is allowed within the scope of a configured IPsec policy. Ensure an IPsec policy has been applied to the VLAN with proper routing*."

FCS_NTP_EXT.1.3

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

Findings:	Section 2.2.5 of [SUPP] indicates that the "*…TOE by default does not accept broadcast or multicast NTP packets*."

3.3.11.3 Tests

FCS_NTP_EXT.1.1

249 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.

High-Level Test Description
With the TOE configured to read the time from an NTP server, capture packets and review the TOE-advertised protocol version to ensure it matches the claimed version(s).
Findings: PASS – The evaluator confirmed that the TOE advertises support for NTPv4 which is consistent with the selection in FCS_NTP_EXT.1.1.

FCS_NTP_EXT.1.2

250 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 new value does not match the configuration on the TOE and confirms that the TOE does not synchronize to this time source.

High-Level Test Description
Using a custom tool, force the NTP server to respond with a MAC which uses the wrong hashing algorithm to respond.
Findings: PASS – The evaluator confirmed that the TOE failed to synchronize to NTP response messages that used an invalid digest algorithm.

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.

High-Level Test Description
Configure the NTP server to have a different time than the TOE. Ensure the TOE uses NTP to synchronize to the NTP server. Verify the correct version and hashing algorithm is being used. Show that the TOE is accepting the time from the NTP server.
Findings: PASS – The evaluator confirmed that the TOE accepts the NTP server's timestamp update and the appropriate protocol was used to ensure integrity of the timestamp.

FCS_NTP_EXT.1.3

251 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.

High-Level Test Description
Configure the NTP server in the environment to only transmit broadcast and multicast. Toggle the NTP client functionality on the TOE and show that the TOE does not synchronize to the NTP server in the environment.
Findings: PASS – The evaluator confirmed that the TOE does not accept broadcast and multicast NTP packets.

FCS_NTP_EXT.1.4

NIAP TD0528

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.

High-Level Test Description
Configure three NTP time servers. When configuring them, ensure that only one IP is active at any given time and show that the TOE is capable of synchronizing to it. Remove all three NTP time servers when complete to show conformance with auditing requirements.
Findings: PASS – The evaluator confirmed that the TOE supports configuration of at least three NTP time sources and the TOE is capable of accepting the NTP packets from each source.

NIAP TD0528

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.

High-Level Test Description
Using a custom tool, transmit legitimate NTP server responses such that the NTP client could theoretically respond. Show that the TOE ignores these response because they are spoofed.
Findings: PASS – The evaluator confirmed that the TOE does not synchronize to other, not explicitly configured time sources.

3.3.12 FCS_RBG_EXT.1 Extended: Cryptographic Operation (Random Bit Generation)

3.3.12.1 TSS

252 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.

Findings:	Section 6.3.9 of the [ST] claims that the TOE uses a software based random bit generator that complies with AES-256 CTR_DRBG when operating in FIPS mode. AES-256 is used in conjunction with a minimum of 256 bits of entropy accumulated from one software based noise source that includes the following: i) Timing variances over computation operations. ii) Timing variances over memory accesses. The entropy value provided by these sources, combined with a NIST vetted SHA3-256 conditioning operation, suffices the minimum requirements for a FIPS approved Jitter entropy implementation that is used to seed the CTR_DRBG which is leveraged by the MC and RAP components.

3.3.12.2 Guidance Documentation

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

Findings:	According to section 2.2.7 of the [SUPP], there is no further configuration required to configure the RNG.

3.3.12.3 Tests

254 The evaluator shall perform 15 trials for the RNG implementation. If the RNG is configurable, the evaluator shall perform 15 trials for each configuration.

255 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).

256 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.

257 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.

Findings:	The vendor uses CAVP certificate A2690 for DRBG operations. This is described in [ST] Table 18.

3.3.13 FCS_SSHS_EXT.1 SSH Server

3.3.13.1 TSS

FCS_SSHS_EXT.1.2

NIAP TD0631

258 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).

259 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.

260 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.

Findings:	Within [ST] section 6.3.13 the TSS states that the TOE supports password-based or client public key (ssh-rsa) authentication which is consistent with the signature verification algorithms selected in FCS_COP.1/SigGen. Section 6.3.13 of the [ST] TSS states that "[d]uring authentication, the TOE establishes a user identity by either verifying that the SSH client's current public key matches the one stored within the TOE's SSH authorized keys file, or by confirming the validity of the presented username and matching password within its database."

FCS_SSHS_EXT.1.3

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

Findings:	Within [ST] section 6.3.13 the TSS states that the TOE examines the size of each received SSH packet. The TOE limits packets to 256k bytes. As SSH packets are being received, the TOE uses a buffer to build all packet information. Once complete, the packet is checked to ensure it can be appropriately decrypted. However, if it is not complete when the buffer becomes full (256k bytes), the packet will be dropped.

FCS_SSHS_EXT.1.4

262 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.

Findings:	Within [ST] section 6.3.13 the TSS states that the TOE utilises AES-CBC-128, AES-CBC-256, AES-128-CTR and AES-256-CTR for SSH encryption. Optional characteristic RFC4344 is specified. The encryption algorithms specified are identical to those listed for FCS_SSHS_EXT.1.4.

FCS_SSHS_EXT.1.5

NIAP TD0631

263 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.

Findings:	Section 6.3.13 of the [ST] states that SSHv2 supports server authentication using RSA public-keys with algorithms ssh-rsa, rsa-sha2-256, and rsa-sha2-512. These algorithms are consistent with those claimed in section 5.3.3 of the [ST] listed for the component.

FCS_SSHS_EXT.1.6

264 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.

Findings:	Within [ST] section 6.3.13 the TSS states that the TOE provides data integrity for SSH connections via HMAC-SHA1, HMAC-SHA1-96 and HMAC-SHA2-256. The list corresponds to the selections made for the component.

FCS_SSHS_EXT.1.7

265 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.

Findings:	Within [ST] section 6.3.13 the TSS states that the TOE supports ecdh-sha2-nistp256 and ecdh-sha2-nistp384 for SSH key exchanges. The list corresponds to the selections in this component.

FCS_SSHS_EXT.1.8

266 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.

Findings:	Within [ST] section 6.2.12 the TSS states that the TOE will re-key SSH connections after 1 hour of after 512 MB of data has been exchanged (whichever occurs first).

3.3.13.2 Guidance Documentation

FCS_SSHS_EXT.1.4

267 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).

Findings:	Section 2.2.8 of [SUPP] indicates no configuration is needed to specify the permitted algorithms after 'fips enable' has been set. The controller will attempt negotiations using AES128-CBC, AES256-CBC, AES128-CTR, and AES256-CTR. These algorithms are consistent with the TSS in section 6.3.13 of the [ST].

FCS_SSHS_EXT.1.5

268 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).

Findings:	Section 2.2.8 of [SUPP] indicates no configuration is needed to specify the permitted algorithms after 'fips enable' has been set. The controller will negotiate SSH-RSA, RSA-SHA2-256, and RSA-SHA2-512 public-key algorithms.

FCS_SSHS_EXT.1.6

269 The evaluator shall also check the guidance documentation to ensure that it contains instructions to the 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).

Findings:	Section 2.2.8 of [SUPP] indicates no configuration is needed to specify the permitted algorithms after 'fips enable' has been set. The controller will attempt negotiations using *"… HMAC-SHA-1, HMAC-SHA1-96, and HMAC-SHA2-256…"*

FCS_SSHS_EXT.1.7

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

Findings:	Section 2.2.8 of [SUPP] indicates no configuration is needed to specify the permitted algorithms after 'fips enable' has been set. The controller will attempt negotiations using *"…the following key exchange methods: ecdh-sha2-nistp256 and ecdh-sha2-nistp384."*

FCS_SSHS_EXT.1.8

271 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.

Findings:	[SUPP] section 2.2.8 states that SSH rekey intervals are non-configurable and are set to a maximum time interval of one (1) hour or 512M, whichever occurs first.

3.3.13.3 Tests

FCS_SSHS_EXT.1.2

NIAP TD0631

272 Test objective: The purpose of these tests is to verify server supports each claimed client authentication method.

273 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.

High-Level Test Description
Configure an SSH public key for a user in the system using all claimed algorithms. Using the private key half, log into the TOE using an SSH client and show that the attempt was successful.
Findings: PASS – The evaluator confirmed all supported public-key algorithms were able to authenticate to the TOE.

274 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.

High-Level Test Description
Using a TOE with an administrator already configured for SSH public key authentication, generate a new RSA private key half and use it to attempt to log into the TOE. Show that the attempt fails.
Findings: PASS - The evaluator confirmed that attempting to authenticate to the TOE using an SSH key ssh-rsa that was not configured as trusted resulted in an authentication failure

275 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.

High-Level Test Description
Verify the TOE allows users to authenticate using a password.
Findings: PASS - The evaluator confirmed a password could be used to authenticate to the TOE while performing FIA_UIA_EXT.1 Test 1.

276 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.

High-Level Test Description
Verify using an incorrect password results in an authentication failure.
Findings: PASS - The evaluator confirmed that using an incorrect password resulted in an authentication failure while performing FIA_UIA_EXT.1 Test 1.

FCS_SSHS_EXT.1.3

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

High-Level Test Description
Using a custom tool, send a packet to the TOE SSH server which is larger than the maximum SSH packet length. Show that the packet is rejected. Show that the connection is terminated and that an audit message is received.
Findings: PASS - The evaluator confirmed the TOE rejects SSH packets larger than 256KB.

FCS_SSHS_EXT.1.4

278 The evaluator must ensure that only claimed ciphers and cryptographic primitives are used to establish a SSH connection. To verify this, the evaluator shall start session establishment for a 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.

High-Level Test Description
For each of the claimed ciphers, connect to the TOE and only permit a single cipher to be negotiated. Show that the TOE will successfully negotiate the cipher.
Findings: PASS - The evaluator confirmed that the TOE successfully negotiates each claimed encryption algorithms and only proposes the claimed encryption algorithms.

FCS_SSHS_EXT.1.5

NIAP TD0631

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

280 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.

High-Level Test Description
Using an SSH client, attempt to force to use the supported host public key algorithms and show that they are transmitted from the TOE back to the client.
Findings: PASS - The evaluator confirmed the TOE successfully identifies itself with each hostkey algorithm specified in the ST.

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

282 Has effectively been moved to FCS_SSHS_EXT.1.2.

283 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.

284 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.

High-Level Test Description
Using an SSH client, attempt to force to use an unsupported host public key algorithms and show that the TOE fails to connect.
Findings: PASS - The evaluator confirmed that the SSH connection was rejected when the client proposed a hostkey algorithm not claimed by the TOE.

FCS_SSHS_EXT.1.6

285 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.

286 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.

High-Level Test Description
Using an SSH client, use each of the supported integrity algorithms and show that the TOE successfully connects.
Findings: PASS - The evaluator confirmed the TOE successfully establishes an SSH connection with each claimed integrity algorithm.

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

288 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.

High-Level Test Description
Using an SSH client, attempt to force to use an unsupported integrity algorithm and show that the TOE fails to connect.
Findings: PASS – The evaluator confirmed an SSH connection with the TOE fails when an unsupported HMAC algorithm is proposed by the client.

FCS_SSHS_EXT.1.7

289 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.

High-Level Test Description
Using an SSH client, attempt to force to use an unsupported key exchange algorithm and show that the TOE fails to connect.
Findings: PASS - The evaluator confirmed an SSH connection with the TOE fails when diffie-hellman-group1-sha1 is the only key exchange algorithm proposed by the client.

290 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.

High-Level Test Description
Using an SSH client, use each supported key exchange algorithm and show that the TOE successfully connects.
Findings: PASS - The evaluator confirmed the TOE successfully establishes a connection using ecdh-sha2-nistp256 and ecdh-sha2-nistp384.

FCS_SSHS_EXT.1.8

291 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.

292 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).

293 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.

High-Level Test Description
Using a custom SSH client, trickle data to the SSH server and detect a rekey initiated by the TOE SSH server within an hour.
Findings: PASS - The evaluator confirmed the TOE initiates a rekey before 1 hour is exceeded.

294 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).

295 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).

296 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.

High-Level Test Description
Using a custom SSH client, sent volumes of data to the SSH server and detect a rekey initiated by the TOE SSH server before 1 GB of data has been delivered.
Findings: PASS - The evaluator confirmed the TOE initiates a rekey before 512MB of data has been encrypted or decrypted using a key.

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

Findings:	These limits are not configurable for this TOE.

298 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.

Findings:	The TOE does not have hardware limitations.

3.3.14 FCS_TLSS_EXT.1 Extended: TLS Server Protocol

3.3.14.1 TSS

FCS_TLSS_EXT.1.1

299 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.

Findings:	Within [ST] section 6.3.14 the TSS states that the following ciphersuites are implemented by the TOE by default: • TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492 • TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492 • TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 • TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492 • TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492 • TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 • TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289 These correspond with selection made in the SFR.

FCS_TLSS_EXT.1.2

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

Findings:	Within [ST] section 6.3.14 the TSS states that the server only allows TLS protocol version 1.2 exclusively and rejects any other protocol version.

FCS_TLSS_EXT.1.3

NIAP TD0635

301 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.

Findings:	Within [ST] section 6.3.14 the TSS states that the TOE performs key establishment using ECDHE curves secp256r1.

FCS_TLSS_EXT.1.4

302 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).

Findings:	Within [ST] section 6.3.14 the TSS states that the TOE supports session resumption based on session IDs according to RFC 5246 and session tickets according to RFC 5077.

303 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.

Findings:	[ST] Section 6.3.14 states that session tickets are protected by implementing symmetric encryption algorithms as described in FCS_COP.1/DataEncryption

304 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.

Findings:	Within section 6.3.14 of the [ST] the TSS states that the TOE session tickets adhere to the structural format provided in section 4 of RFC 5077.

NIAP TD0569

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

Findings:	Within [ST] section 6.3.14 the TSS states session resumption is based on a single context and operates according to the applicable RFCs. Sessions can be reused providing all session properties are still valid and parameters are otherwise not accepted by the TOE. If the latter occurs, a full handshake would be performed.

3.3.14.2 Guidance Documentation

FCS_TLSS_EXT.1.1

305 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).

Findings:	Section 2.2.9 of the [SUPP] indicates that no configuration is required to set the permitted cipher suites once ‘fips enable’ has been entered on the controller. The [SUPP] section then goes on to list the ciphersuites supported by the TOE in its default configured state and the list is consistent with the TSS.

FCS_TLSS_EXT.1.2

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

Findings:	Section 2.2.9 of the [SUPP] provides instructions to the administrator to restrict the WebUI TLS protocol to version 1.2 using the “web-server profile ssl-protocol tlsv1.2” CLI command.

FCS_TLSS_EXT.1.3

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

Findings:	Section 2.2.9 of the [SUPP] provides that the specific WebUI server certificate can be adjusted using the “web-server profile switch-cert <cert>” option. Certificates can be sourced from the certificate trust store to support both RSA and ECDSA key types. The guidance documentation claims that the TOE performs key establishment with DH parameters over NIST curve secp256r1.

NIAP TD0569

FCS_TLSS_EXT.1.4

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

Findings:	The guidance documentation does not describe any configuration necessary to support session resumption. This is consistent with testing.

3.3.14.3 Tests

FCS_TLSS_EXT.1.1

309 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).

High-Level Test Description
Using a custom TLS tool, connect to the TOE using each claimed ciphersuite and show that it works. When switching between RSA and ECDSA, ensure that the web server certificate is switched accordingly.
Findings: PASS - The evaluator confirmed the TOE allows TLS connections with each claimed ciphersuite.

310 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.

High-Level Test Description
Using a custom TLS tool, connect to the TOE using a specific unsupported ciphersuite and show that the TOE rejects the connection and generates an audit message. Also attempt to connect to the TOE using the TLS_NULL_WITH_NULL_NULL ciphersuite and show that the TOE rejects the connection and generates an audit message.
Findings: PASS - Findings: PASS – The evaluator confirmed the TOE rejects connection using the unsupported ciphersuite and TLS_NULL_WITH_NULL_NULL ciphersuite.

311 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.

High-Level Test Description
Using a custom TLS tool, connect to the TOE and transmit a mangled Encrypted Handshake (Finished) message and verify that the TOE fails to complete the handshake.
Findings: PASS - – The evaluator confirmed the TOE rejects a connection when the client sends a modified/corrupted Client Finished message.

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'.

High-Level Test Description
Using a custom TLS tool, connect to the TOE and perform a good handshake and show that application data flowed. Analyse the properties of the Encrypted Handshake (Finished) message and show that it meets the requirements as described above.
Findings: PASS – The evaluator confirmed the TOE encrypts the Server Finished message.

FCS_TLSS_EXT.1.2

312 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.

High-Level Test Description
Using a custom TLS tool, iterate over each of SSLv2, SSLv3, TLSv1, TLSv1.1 and TLSv1.2 to determine which are supported. Only TLS 1.2 should result in a successful handshake.
Findings: PASS – The evaluator confirmed the TOE does not negotiate unsupported versions of TLS/SSL.

FCS_TLSS_EXT.1.3

313 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.

High-Level Test Description
Using a custom TLS tool, connect to the TOE using a supported ECDHE ciphersuite and a supported elliptic curve. Verify that the TOE selects the curve offered by the client.
Findings: PASS – The evaluator confirmed the TOE successfully establishes the connection with the supported elliptic curve.

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.

High-Level Test Description
Using a custom TLS tool, connect to the TOE using a supported ECDHE ciphersuite and an unsupported elliptic curve. Verify that the Server Hello is not sent and the connection is unsuccessful.
Findings: PASS – The evaluator confirmed the TOE does not send a Server Hello message and the connection is not established when the client sends an unsupported elliptic curve.

314 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).

High-Level Test Description
There are no DHE ciphersuites supported.
Findings: N/A

315 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.

High-Level Test Description
There are no RSA key establishment ciphersuites supported.
Findings: N/A

FCS_TLSS_EXT.1.4

316 *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).*

317 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.

NIAP TD0569

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

High-Level Test Description
The TOE claims session resumption based on both session IDs according to RFC5246 (TLS1.2) and session tickets according to RFC5077.
Findings: N/A

318 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.

NIAP TD0569

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

High-Level Test Description
Using a custom tool, perform the test case as described in the Supporting Document and show that the session is resumed for the case of test 2a and not resumed for test 2b.
Findings: PASS – The evaluator confirmed in step 1 that the TOE successfully resumed the session as described in test 2a, and the TOE did not resume a session when an attempt to reuse the session ID from the disrupted handshake was presented as described in test 2b.

319 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):

NIAP TD0556

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

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

NIAP TD0569

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

High-Level Test Description
Using a custom tool, perform the test case as described in the Supporting Document and show that the session is resumed for the case of test 3a and not resumed for test 3b.
Findings: PASS – The evaluator confirmed in step 1 that the TOE successfully resumed the session as described in test 3a, and the TOE did not resume a session when an altered/invalid session ticket was presented as described in test 3b.

320

3.4 Identification and Authentication (FIA)

3.4.1 FIA_AFL.1 Authentication Failure Management

3.4.1.1 TSS

321 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.

Findings:	In the [ST] section 6.4.1, after an administrator specified number of consecutive failed authentication attempts between 1 and 10 that occur in a three minute period, the TOE will lockout the offending remote administrator and log the event. The duration in time that a user is locked out upon crossing the lock out threshold is 0-60 minutes. The offending administrator will remain locked out until the administrator configured lock-out period has expired.

322 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).

Findings:	Section 6.4.1 of the [ST] claims that an administrator with a public-key will never be locked out since public-key-based authentication is not subject to the password lock-out function.

3.4.1.2 Guidance Documentation

323 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.

Findings:	Section 2.4.1 of the [SUPP] provide the CLI commands to configure the successive failed authentication lock out mechanism, including the time period for re-enabling locked out users. There is no indication that the mechanism is different for different interfaces. There is no process to permit a remote administrator to once again successfully log on, other than to wait for the lock out time period to expire.

324 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.

Findings:	Section 2.4.1 of the [SUPP] indicates that remote administrators configured with a public key cannot be locked out.

3.4.1.3 Tests

325 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.

High-Level Test Description
Using the CLI, set the login threshold to 3 attempts. Change the duration to 3 minutes. Using the Web interface, log into the TOE twice using an incorrect password. On the third attempt, log in correctly and verify that the threshold has not been reached. Using the Web interface, log into the TOE three times using an incorrect password. On the fourth attempt, log in correctly and verify that the threshold has been reached and that the user cannot log in. Do the same on the SSH CLI.
Findings: PASS - After configuring the number of successive unsuccessful authentication attempts, the evaluator confirmed that once the authentication limit is reached, any further attempts on any remote interface, even 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).

High-Level Test Description
Function not applicable
Findings: N/A - The TOE does not claim a specific manual action to unlock a remote administrator's access.

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.

High-Level Test Description
Configure the lock out period to be 3 minutes. Lock out a user on the CLI interface and wait 2m30s. Show that the account is not unlocked. Wait 1m45s longer and show that the account is unlocked. The TOE can take up to 60 seconds past the configured timeout to unlock the account. Configure the lock out period to be 5 minutes. Lock out a user on the Web UI and wait 4m30s. Show that the account is not unlocked. Wait 1m45s longer and show that the account is unlocked. The TOE can take up to 60 seconds past the configured timeout to unlock the account.
Findings: PASS - After configuring the unlock time, the evaluator found that trying to authenticate before the time had expired resulted in failed authentication, while authenticating after the time had expired resulted in a successful authentication.

3.4.2 FIA_PMG_EXT.1 Password Management

3.4.2.1 TSS

NIAP TD0792

326 The evaluator shall check that the TSS lists the supported special character(s) for the composition of administrator passwords.

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

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

Findings:	In the [ST] in section 6.4.2, the TSS provides the list of the supported special characters as ! @ # $ % ^ & * ( ) _ +. The passwords are configurable between 8 and 32 characters.

3.4.2.2 Guidance Documentation

329 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.

Findings:	Section 2.4.2 of the [SUPP] provides the CLI commands needed to configure the password complexity options, including the set of characters used to compose passwords. In addition, the CLI commands include a setting to permit the TOE to enforce the minimum length of a password. The evaluated configuration demands a minimum password length of 8 characters or more be used. In [ADMIN], under "Management > Implementing a Password Policy", the available range of the password length is between 6 and 128 characters. The maximum range is greater than the minimum maximum of 15 characters required by the Protection Profile. [SUPP] section 2.4.2 also includes some guidance to end-users on picking a strong password: "Once configured, the TOE only permits the use of strong passwords which should be greater than 8 characters in length and contain a sufficiently unique set of characters representative of all character types described in this section."

3.4.2.3 Tests

330 The evaluator shall perform the following tests.

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

High-Level Test Description
Change the management password length to be 15 characters. Change the password for the built-in 'admin' user using the identified TSFI. Show that the password can be used to login to the Web GUI and local console. Change the password for the built-in 'admin' back to a known good password. Change the password length to be 8 characters. Change the password for the admin user to be only 7 characters and show it is rejected. Change the password for the admin user to be 8 characters and show it is accepted.
Findings: PASS - The evaluator confirmed that 8 character passwords and passwords consisting of all claimed characters could be successfully set and used to login

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

High-Level Test Description
Testing of the invalid password was performed as part of FIA_PMG_EXT.1, Test 1.
Findings: PASS - The evaluator confirmed that the TOE did not allow the user to set passwords that did not meet the configured minimum length.

3.4.3 FIA_UAU_EXT.2 Password-based Authentication Mechanism

331 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.

3.4.4 FIA_UAU.7 Protected Authentication Feedback

3.4.4.1 TSS

332 None

3.4.4.2 Guidance Documentation

333 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.

Findings:	Section 2.4.3 of the [SUPP] indicates no configuration is required to mask the authentication data while performing a local login.

3.4.4.3 Tests

334 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.

High-Level Test Description
Log into the serial console and show that the password is obscured as per the claims in the ST.
Findings: PASS - The evaluator confirmed that no feedback is provided while entering authentication information.

3.4.5 FIA_UIA_EXT.1 User Identification and Authentication

3.4.5.1 TSS

335 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”.

Findings:	Section 6.4.3 of the [ST] provides this information. Prior to requiring the non-TOE entity to initiate the identification and authentication process, the TOE displays an Authorized Administrator-specified advisory notice and consent warning message regarding unauthorized use of the TOE (FTA_TAB.1).). The logon process is initiated by the administrator via the desired interface where an authentication challenge is presented to the administrator. If the credentials entered by the administrator are valid, the authentication sequence will complete successfully and the administrator is presented with the administration interface.

336 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.

Findings:	Section 6.4.3 of the [ST] states the TOE requires an administrator to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that user.

337 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.

Findings:	Section 6.4.3 of the [ST] claims that once the TOE is operational administrators can access the TOE interfaces through the Mobility Controller via the WebUI (HTTPS/TLS) and CLI (SSH) only via authentication. Administrator cannot log in to the RAP since the interfaces are disabled after initial configuration.

338 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.

Findings:	Section 6.4.3 of the [ST] states the TOE requires an administrator to be successfully identified and authenticated before being presented with the administration console and allowing any additional TSF-mediated actions to be executed on behalf of that user. This only applies to the Mobility Controller because, as per section 6.4.3 of the [ST], administrators cannot log in to the RAP since those interfaces are disabled after initial configuration.

3.4.5.2 Guidance Documentation

339 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.

Findings:	Section 2.4.5 of the [SUPP] indicates that users can authenticate via SSH, the WebUI interface, or a direct serial connection. The types of credentials that can be established are briefly discussed: local username/password, local SSH public key, and remote RADIUS/TACACS+ credentials. The means by which the authentication methods can be configured are described. Section 2.4.4 of the [SUPP] informs an administrator how to provision a local user account. Section 2.4.5 of the [SUPP] points the administrator to the relevant manual sections of the [ADMIN] to establish a connection to a remote authentication server. Section 2.2.8 of the [SUPP] describes the CLI commands to establish SSH public key material.

3.4.5.3 Tests

340 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.

High-Level Test Description
Log into the identified management interface using a known-good credential and logout. Attempt to log into the identified management interface using a known-bad credential and verify that the operator cannot login. Ensure the appropriate audit messages appear. Repeat for all claimed credential and interface combinations covering local users, RADIUS, and TACACS. Furthermore, show that the password rescue account is not available on any interface.
Findings: PASS - The evaluator confirmed that the TOE permits logins when valid credentials are used and denies logins when invalid credentials are used. Note that use of good and bad public key authentication was tested as part of FCS_SSHS_EXT.1.2 tests 1 and 2, respectively.

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.

High-Level Test Description
The device does not have any services configured prior to I&A. All claimed services available to remote entities are identified as part of AVA_VAN.1 test scanning.
Findings: PASS - The evaluator confirmed that viewing the warning banner is the only service available to remote entities prior to authentication.

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.

High-Level Test Description
At the local console, verify the user is unable to run any commands or services other than the warning banner.
Findings: PASS - The evaluator confirmed that viewing the warning banner is the only service available at the local console prior to authentication.

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.

High-Level Test Description
Ensure that all components in the distributed TOE are tested for authentication as per the TSS.
Findings: PASS - The Aruba Mobility Controller is the only component in the distributed TOE which permits security administrators to authenticate. This was tested in FIA_UIA_EXT.1 test 1.

3.4.6 FIA_X509_EXT.1/ITT X.509 Certificate Validation

3.4.6.1 TSS

341 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.

Findings:	[ST] section 6.4.7: Certificates used in authentication of distributed TOE communication (between AP and MC) are validated as described in Section 6.4.6 of the ST above using a minimum certificate chain path of two. These channels are IPsec (VPN connection). Per the NDcPP, revocation checking is optional due to the additional requirements surrounding the enabling and disabling of the ITT channel as defined in FCO_CPC_EXT.1. No unsupported rules for the extendedKeyUsage fields are claimed.

3.4.6.2 Guidance Documentation

342 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.

Findings:	[SUPP] Section 2.4.6 in the guidance describes where the check of validity takes place: *"The validity of peer certificates will be checked upon establishment of connections. Any server certificates uploaded to the TOE will be checked at that time."* Other extendedKeyUsage field values "…*are trivially satisfied*" in the [SUPP] section 2.4.6.

3.4.6.3 Tests

343 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.

High-Level Test Description
Configure the Mobility Controller with a certificate which is chained to the trust anchor configured at provisioning time on the RAP device. Show that the IPsec connection succeeds. Configure the Mobility Controller with a certificate which is chained to a trust anchor NOT configured with the RAP device. Show that the RAP fails to validate the certificate.
Findings: PASS – The evaluator confirmed that the TOE will successfully validate the leaf certificate when provided with a valid chain terminating in a trusted CA certificate. The evaluator confirmed that when removing an intermediate CA certificate the TOE fails to validate the chain.

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

High-Level Test Description
Configure the Mobility Controller with an expired certificate chained to the trust anchor configured at provisioning time on the RAP device. Show that the IPsec connection fails.
Findings: PASS – The evaluator confirmed that the IPsec fails to establish a tunnel when the peer presents an expired certificate.

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.

High-Level Test Description
Revocation is not claimed for the Intra-TOE transfer channel.

High-Level Test Description
Findings: N/A

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.

High-Level Test Description
Revocation is not claimed for the Intra-TOE transfer channel.
Findings: N/A

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.)

High-Level Test Description
Use the TOE to generate a new CSR. Sign the CSR to create a new certificate. However, mangle the certificate before loading to the TOE. The TOE will fail to load the mangled certificate.
Findings: PASS – The evaluator confirmed that the certificate fails to validate and the TOE does not accept the certificate for use.

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.)

High-Level Test Description
Configure the Mobility Controller with a certificate chained to the trust anchor configured at provisioning time on the RAP device. Mangle the signature on the certificate such that the recipient should not be able to determine the difference between a mangled certificate and one signed by a completely different CA. Show that the IPsec connection fails.
Findings: PASS – The evaluator confirmed that the TOE fails to validate a certificate with a modified signature field, and the TOE does not establish a connection.

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.)

High-Level Test Description
Use the TOE to generate a new CSR. Sign the CSR to create a new certificate. However, mangle the certificate's public key before loading to the TOE. The TOE will fail to load the mangled certificate.
Findings: PASS – The evaluator confirmed that the certificate fails to validate and the TOE does not accept the certificate for use.

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.

High-Level Test Description
The TOE only supports a chain of two (2) certificates and therefore no Intermediate chain is being presented.
Findings: N/A

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.

High-Level Test Description
The TOE only supports a chain of two (2) certificates and therefore no Intermediate chain is being presented.
Findings: N/A

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.

High-Level Test Description
The TOE only supports a chain of two (2) certificates and therefore no subordinate chain is being presented.
Findings: N/A

344 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

345 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).

346 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).

High-Level Test Description
Construct a certificate such that the intermediate certificate is missing the basicConstraints extension. Show that the TOE fails to load the certificate into the trust store because it is missing the basicConstraints extension.
Findings: PASS - The administrative trust store for the TOE resides on the controller. The controller has been tested as per FIA_X509_EXT.1.2/Rev test 1.

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).

High-Level Test Description
Construct a certificate such that the intermediate certificate is missing the basicConstraints extension. Show that the TOE fails to load the certificate into the trust store because it is missing the basicConstraints extension.
Findings: PASS - The administrative trust store for the TOE resides on the controller. The controller has been tested as per FIA_X509_EXT.1.2/Rev test 2.

3.4.7 FIA_X509_EXT.1/Rev X.509 Certificate Validation

3.4.7.1 TSS

347 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).

Findings: Within [ST] section 6.4.6 the TSS states that the TOE performs X.509 certificate validation at the following points:

a) On load of certificate responses
b) When processing OCSP responses
c) During IPsec peer authentication

In all scenarios, certificates are checked for several validation characteristics:

a) If the certificate 'notAfter' date is in the past, then this is an expired certificate which is considered invalid;
b) The certificate chain must terminate with a trusted CA certificate;
c) A trusted CA certificate is defined as any certificate loaded into the TOE trust store that has, at a minimum, a basicConstraints extension with the CA flag set to TRUE;
d) The TOE validates the extendedKeyUsage field as follows:
    i) TLS server certificates must have the Server authentication purpose in the extendedKeyUsage field
    ii) OCSP certificates must have the OCSP signing purpose in the extendedKeyUsagefield

Certificate revocation checking is performed using OCSP.

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.

Findings: Within [ST] section 6.4.6 the TSS states that Certificate revocation checking is performed when certificates are presented to the TOE and when loaded into the TOE. Revocation status is checked using OCSP as specified in RFC 6960. All certificates in the chain except for the root are verified in order, starting with the peer cert and ending at the penultimate CA certificate.

3.4.7.2 Guidance Documentation

348 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.

Findings:	[SUPP] Section 2.4.6 in the guidance describes where the check of validity takes place: *"The validity of peer certificates will be checked upon establishment of connections. Any server certificates uploaded to the TOE will be checked at that time."* The TOE validates the extendedKeyUsage field in accordance with OCSP certificates presented for OCSP responses. Other extendedKeyUsage field values "…*are trivially satisfied*" in the [SUPP] section 2.4.6. The TOE performs revocation checking on all certificates in the chain using OCSP as long as it is configured to do so. The [SUPP] section 2.4.6 states that a separate Revocation Check Point (RCP) *"…must be configured for each* [n.b. certificate in the chain]*, along with an appropriate OCSP responder certificate.*"

3.4.7.3 Tests

349 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.

High-Level Test Description
Start by adding the top-level trust anchors for RSA and ECDSA certificate chains and show that the addition is audited. Configure an IPsec tunnel to make use of a peer RSA certificate which is missing the intermediate certificate to validate the chain. Show that the IPsec tunnel fails to establish due to an invalid certificate chain. Show that after adding the intermediate over-the-wire, the IPsec connection succeeds. Configure an IPsec tunnel to make use of a peer ECDSA certificate which is missing the intermediate certificate to validate the chain. Show that the IPsec tunnel fails to establish due to an invalid certificate chain. Show that after adding the intermediate over-the-wire, the IPsec connection succeeds. Remove the top-level trust anchors for RSA and ECDSA certificates and show that the removal is audited.

High-Level Test Description
Findings: PASS – The evaluator confirmed that the TOE will successfully validate the leaf certificate when provided with a valid chain terminating in a trusted CA certificate. The evaluator confirmed that when removing an intermediate CA certificate the TOE fails to validate the chain.

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

High-Level Test Description
Transmit an expired certificate from the peer system to the TOE and show that the TOE fails to establish the tunnel.
Findings: PASS – The evaluator confirmed that the IPsec fails to establish a tunnel when the peer presents an expired certificate.

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.

High-Level Test Description
Load the CA into the TOE trust store. Ensure the OCSP has no revoked certificates. Verify that a certificate results in a successful connection. Then revoke the server certificate and restart the OCSP server. Verify the connection now fails due to the certificate being revoked. Then unrevoked the certificate from the OCSP and restart the OCSP server. Revoke the intermediate CA and restart the root CA OCSP server. Verify the connection now fails due to the certificate being revoked. Then unrevoked the intermediate CA and restart the OCSP server. Verify that a certificate now results in a successful connection.
Findings: PASS – The evaluator confirmed that the TOE successfully establishes a connection when valid certificates are used and will not establish a connection if either the leaf or intermediate CA certificates are revoked.

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.

High-Level Test Description
Load the CA into the TOE trust store. Create an OCSP signing certificate using a known good CA certificate that has the OCSPSigning extendedKeyUsage flag enabled. Create an OCSP signing certificate in which the OCSPSigning extendedKeyUsage has been removed. Verify the connection now fails due to the OCSP response being signed by a delegate without the proper flag.
Findings: PASS – The evaluator confirmed that if the OCSP server presents a certificate that does not have the OCSP signing purpose the TOE rejects the OCSP response and the connection 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.)

High-Level Test Description
Using a custom tool which damages part of the ASN.1 structure in the first 8 bytes of a specified certificate, transmit a certificate from the peer system to the TOE and show that the TOE fails to establish the tunnel.
Findings: PASS – The evaluator confirmed that the certificate fails to validate and the TOE does not establish the connection.

f) Test 6: The evaluator shall modify any byte in the certificate signatureValue field (see RFC5280 Sec. 4.1.1.3), which is normally the last field in the certificate, and demonstrate that the certificate fails to validate. (The signature on the certificate will not validate.)

High-Level Test Description
Using a custom tool which damages part of the certificate digital signature field of a specified certificate, transmit a certificate from the peer system to the TOE and show that the TOE fails to establish the tunnel.
Findings: PASS – The evaluator confirmed that the TOE fails to validate a certificate with a modified signature field, and the TOE does not establish a connection.

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.)

High-Level Test Description
Using a custom tool which damages part of the certificate public key of a specified certificate, transmit a certificate from the peer system to the TOE and show that the TOE fails to establish the tunnel.
Findings: PASS - The evaluator confirmed the connection fails when the TOE receives a certificate with a modified public key.

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.

High-Level Test Description
The evaluator confirmed the connection fails when the TOE receives a certificate with a modified public key.
Findings: PASS - This test case was conducted in FIA_X509_EXT.1.1/Rev test 1a/1b showing ECDSA certificates being used such that the entire chain is ECDSA with the root ECDSA certificate loaded into the TOE and the intermediate and leaf certificates being delivered over the wire. The evaluator confirmed the TOE validates the valid 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.

High-Level Test Description
Deliver an explicitly parameterized intermediate ECDSA certificate to the TOE and show that the connection fails.
Findings: PASS – The evaluator confirmed the when the TOE is presented with an intermediate ECDSA certificate with explicit format parameters the TOE will reject the certificate.

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.

High-Level Test Description
Load a named curve intermediate ECDSA certificate to the TOE's trust store and show that it works. Load an explicitly parameterized intermediate ECDSA certificate to the TOE and show that the load to the trust store fails.
Findings: PASS – The evaluator confirmed that the TOE successfully loads the intermediate CA with elliptic curve parameters specified as named curves and rejects the intermediate CA with elliptic curve parameters that use explicit format.

350 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.

351 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).

352 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).

High-Level Test Description
Construct a certificate such that the intermediate certificate is missing the basicConstraints extension. Show that the TOE fails to load the certificate into the trust store because it is missing the basicConstraints extension.
Findings: PASS – The evaluator confirmed the TOE fails to load the intermediate certificate that does not contain the basicConstraints extension.

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).

High-Level Test Description
Construct a certificate such that the intermediate certificate's basicConstraints extension is false. Show that the TOE fails to load the certificate into the trust store because of the false basicConstraints extension.
Findings: PASS - The evaluator confirmed the TOE will not trust an intermediate CA certificate with the Basic Constraints extension in which the CA flag is set to FALSE.

353 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).

Findings:	The TOE only claims use of certificates for IPsec trusted channels.

3.4.8 FIA_X509_EXT.2 X.509 Certificate Authentication

3.4.8.1 TSS

354 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.

Findings:	Section 6.4.8 of the [ST] claims that the TOE is capable of validating certificates from IPSec peers and for identifying itself to IPSec peers. When validating certificates from remote peers, the TOE makes use of a trust store to locate certificate chains for validation purposes. When determining which certificates to present to remote peers for identification purposes, the TSS indicates that administrators can select which certificate to use to present.

355 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.

Findings:	Section 6.4.8 of the [ST] states that when a connection cannot be established to determine the validity of a certificate, the certificate is not accepted

3.4.8.2 Guidance Documentation

356 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.

Findings:	Certificates are used by the TOE for IPsec connections and for the WebUI management interface. Sections 2.2.6.9 and 2.2.6.10 of the [SUPP] provides the information needed to use certificates for IPsec. Section 2.2.9 of [SUPP] provides the information needed to use certificates for the WebUI server. Configuration and management of the TOE trust store is described in section 2.4.6 of the [SUPP]. Certificate management is a large topic and the reader is referred to more information on managing the trust store in the [ADMIN] section '*Management Access'* sub-section '*Managing Certificates'.* The TOE claims OCSP as a certificate revocation mechanism for IPsec trusted channels. OCSP configuration is discussed in section 2.4.6 of the [SUPP]. That section provides the use of a CLI command for "crypto-local pki rcp" to set the "server-unreachable" option to "revoke-cert". If a certificate is considered revoked, then the connection will not be permitted.

3.4.8.3 Tests

357 The evaluator shall perform the following test for each trusted channel:

358 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.

High-Level Test Description
Run an OCSP responder for the Intermediate CA, but fail to start an OCSP responder server for the Root CA. Then make an IPsec connection and show that the TOE fails to connect to the Root CA OCSP responder which results in an unknown revocation status for the Intermediate CA. The IPsec connection will fail as a result.
Findings: PASS – The evaluator confirmed the TOE rejected the connection when it was unable to verify the validity of the intermediate CA. This is consistent with the selection in FIA_X509_EXT.2.2.

359

3.4.9 FIA_X509_EXT.3 Extended: X509 Certificate Requests

3.4.9.1 TSS

360 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.

Findings:	"Device-specific information" was not selected for this SFR. Within the [ST] section 6.4.9, the TOE generates Certificate Request Messages and includes the following information: public key, common name, organization, organizational unit, country.

3.4.9.2 Guidance Documentation

361 The evaluator shall check to ensure that the guidance documentation contains instructions on requesting certificates from a CA, including generation of a Certificate

Request. If the ST author selects "Common Name", "Organization", "Organizational Unit", or "Country", the evaluator shall ensure that this guidance includes instructions for establishing these fields before creating the Certification Request.

Findings:	Section 2.4.7 of [SUPP] provides the CLI instructions to generate a CSR. Those instructions contain guidance to provide the Common Name, Country, Organization, and Organizational Unit as selected in the [ST] before the CSR is generated.

3.4.9.3 Tests

362 The evaluator shall perform the following tests:

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

High-Level Test Description
Using the TOE CSR generator, create a new CSR and download to an external CA entity for signing. Using OpenSSL, verify that the information in the CSR is as expected.
Findings: PASS – The evaluator confirmed the TOE generated a Certification Request and provides the public key and other required information as specified in the ST.

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

High-Level Test Description
The CSR from the previous test is signed and reimported into the TOE which cannot be validated and therefore fails. Then load the signing CA into the TOE and retry the import. The import succeeds.
Findings: PASS – The evaluator confirmed that the TOE fails to validate a response message to a Certification Request without a valid certification path. Once a valid trusted CA is loaded the TOE successfully validates the Certification Request response message.

3.5 Security management (FMT)

3.5.1 General requirements for distributed TOEs

3.5.1.1 TSS

363 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.

Findings: A list of security management functions in noted in the [ST] section 6.5.5 for FMT_SMF.1. These include:

- Configure the access banner warnings
- Configure the session inactivity time before session termination
- Perform TOE updates, query TOE version, and verify the updates using digital signature capability
- Configure the interaction between TOE components
- Configure the authentication failure settings
- Configure the cryptographic functionality including modifying, deleting, generating and importing cryptographic keys and certificates for VPN operation
- Configure IPsec functionality including configuring the lifetime for IPsec SAs
- Manage the TOE's trust store and designate X509 v3 certificates as trust anchors
- Configure and Import X.509v3 certificates to the TOE's trust store
- Set the time by configuring NTP services used for timestamps
- Set the time manually
- Manage the trusted public keys database
- Configure the reference identifier for the peer
- Start and stop services

All of the above functions can be performed on the Mobility Controller using either the CLI or the WebUI.

The following security management functions are performed on and provided by the Access Point during initial configuration only:

- Configure and import client credentials (i.e., X.509v3 certificate) to be used for IKE connections
- Configures the IP address of the VPN Gateway in order to establish an IPsec VPN connection

A default administrator account is configured during initial configuration where the password is set by the admin.

The interaction of TOE components can be configured via the Mobility Controller per FCO_CPC_EXT.1 and as described in section 6.2.1 of the ST.

3.5.1.2 Guidance Documentation

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

Findings: Throughout [SUPP], there are configuration references to both the Mobility Controller (MC) and the Remote Access Points (RAP). Where configuration must occur on one device, the specific device is called out in the document.

The evaluator considered table 1 in the [NDcPP] and their specific allocation of functionality for the FMT SFRs. Table 1 of the [NDcPP] indicates that FMT_MTD.1/CoreData must be enforced by all components; FMT_MOF.1/Services, FMT_MTD.1/CryptoKeys, FMT_MOF.1/ManualUpdate, and FMT_SMF.1 as implemented by this specific TOE must be enforced by at least one component; and FMT_SMR.2 must be enforced by at least one of the components in the distributed TOE.

	Secondly, within the [MODVPN] document section 1.1, the TOE component which performs the VPN GW functionality must be implemented by a single component within the distributed TOE. Therefore, management functions FMT_MTD.1/CryptoKeys and FMT_SMF.1/VPN must be documented as being applicable to the MC component. According to various sections in section 2.6 of the [SUPP] use of the TOE's "root" role by an account maps to the [NDcPP] "Security Administrator" role. The "Security Administrator" role in the TOE permits the execution of functionality specific to FMT_MOF.1/ManualUpdate (section 2.5.1 of [SUPP]), FMT_MOF.1/Services (section 2.5.2 of [SUPP]), FMT_MTD.1/CoreData (section 2.5.3 of [SUPP]), and FMT_MTD,1/CryptoKeys (section 2.5.4 of [SUPP]). It is important to note that the RAP devices simply have no way to administer them and therefore the functionality is performed at the Mobility Controller. FMT_SMR.2 requires the TOE to maintain a "Security Administrator" role which is described in the [SUPP] in various sections as mapping to the TOE's "root" role. The "root" role applies to both components. No additional management is required to meet FMT_SMR.2 as described in section 2.5.6 of [SUPP]. For FMT_SMF.1 the [SUPP] in section 2.5.5 refers the reader to [ADMIN] for a full list of configuration instructions available through the CLI and Web GUI. Specific management functions for claimed functions are summarized in the [SUPP] where necessary. For more information on how these administrative actions are performed, please refer to sections 3.5.5 and 4.3.1 in this AAR.

3.5.1.3 Tests

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

High-Level Test Description
Tests defined to verify the correct implementation of security management functions are performed for every applicable TOE component.
Findings: PASS – Security management functions are tested for each applicable component within the test case they belong to.

3.5.2 FMT_MOF.1/ManualUpdate

3.5.2.1 TSS

366 For distributed TOEs see chapter 2.4.1.1 *(n.b. in the NDcPP)*. There are no specific requirements for non-distributed TOEs.

Findings:	See section 3.5.1.1 in this document.

3.5.2.2 Guidance Documentation

367 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).

Findings:	Section 2.7.8 of [SUPP] describes the process for updating the TOE. The steps to perform the upgrade appear to be summarized in the [SUPP] section 2.7.8 for both CLI and WebUI methods of updating the TOE. Of specific note, the Mobility Controller (MC) will automatically push the firmware to managed Remote Access Points (RAPs) but only after a Security Administrator has uploaded a correctly signed firmware image to the MC. The [SUPP] section 2.7.8 indicates that when the RAP firmware is pushed, the RAP will be automatically rebooted, thereby implying that its functions will be unavailable during the upgrade process. Significantly, as per section 2.7.8 of the [SUPP], "*The controller will not allow a connection from the RAP unless* [the RAP and the MC] *are running on the same version of ArubaOS.*"

368 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).

Findings:	Please refer to the previous description above.

3.5.2.3 Tests

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

370 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.

High-Level Test Description
Log into the CLI using an account with privileges which should not permit upgrades. Attempt to upgrade the device. The action should fail. Note that the Remote Access Points derive their boot image from the Controller. If the user cannot provision images on the controller, then the user cannot provision images to the APs.
Findings: PASS – The evaluator logged in as an unprivileged user and confirmed the update using a legitimate update image failed without authentication as Security Administrator. While testing FPT_TUD_EXT.1 Test 1, the evaluator confirmed that the Security Administrator is able to install legitimate updates.

371

3.5.3 FMT_MOF.1/Services Management

3.5.3.1 TSS

372 For distributed TOEs see chapter 3.5.1.1.

Findings:	The TOE is distributed. Please refer to section 3.5.1.1.

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

Findings:	N/A: The TOE is a distributed TOE.

3.5.3.2 Guidance Documentation

374 For distributed TOEs see chapter 3.5.1.2.

Findings:	The TOE is distributed. Please refer to section 3.5.1.2.

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

Findings:	N/A: The TOE is a distributed TOE.

3.5.3.3 Tests

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

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

High-Level Test Description
Using an unprivileged ‘readonly’ user, attempt to disable the remote syslog logging mechanism and show the attempt is unsuccessful. Successful use of this management function is tested in FTP_ITC.1 test 3.
Findings: PASS – The evaluator confirmed the unprivileged user is unable to modify the configuration of the remote logging mechanism. Successful use of this management function is tested in FTP_ITC.1 test 3.

3.5.4 FMT_MTD.1/CoreData Management of TSF Data

3.5.4.1 TSS

378 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.

Findings:	Section 6.5.3 of the [ST] states only Security Administrators can manage TSF data. There are no security functions available through any interfaces prior to administrator login.

379 If 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.

Findings:	Section 6.5.3 of the [ST] states only authorized Security Administrators can manage TSF data. There are no security functions available through any interfaces prior to successful administrator login. Access to TOE functionality is restricted until a Security Administrator has been identified and authenticated, at which point they will be presented with the management console and can then perform all administrative tasks including those listed in [ST] section 6.5.5. Section 6.5.5 of the ST notes that one of the security functions of the admin is: • Manage the TOE's trust store and designate X509 v3 certificates as trust anchors; and • Configure and Import X.509v3 certificates to the TOE's trust store

3.5.4.2 Guidance Documentation

380 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.

Findings:	The evaluator reviewed the [ST] to understand the scope of the management functions in which TSF data could be manipulated. SFRs which require administrative-configurable parameters and management functions in FMT_SMF.1 and FMT_SMF.1/VPN were the inputs to this work unit.
	
	The evaluator considered the following functions which manipulate TSF data and offers the reference to support this assessment:
	• FAU_GEN.1.1(c): Generating/import of, changing, or deleting of cryptographic keys
	  ○ Described for FCS_CKM.1 and FCS_CKM.1/IKE below
	• FAU_GEN.1.1(c): Changing user passwords
	  ○ Described in [SUPP] section 2.4.4 and expanded upon within [CLI] using the "mgmt-user" CLI command
	• FAU_STG_EXT.1.1: Configuring of remote audit parameters (IP, port, protocol, etc.)
	  ○ The IP address of the remote auditing service can be configured as per the instructions in [SUPP] section 2.1.4. The protocol must be IPsec as per the evaluated configuration as described in section

2.1.4 of [SUPP]. The IPsec configuration is managed using commands described in FCS_IPSEC_EXT.1.* below.

- FCO_CPC_EXT.1: altering the allowlist of APs permitted to communicate with the Controller
  - Information on adding and removing RAP devices to the allowlist of APs permitted to communicate with the Controller is described in section 2.3.1 of the [SUPP].
- FCS_CKM.1 and FCS_CKM.1/IKE: generating long-lived keys for
  - TLS web server private key,
    - Please refer to FCS_TLSS_EXT.1.3 below.
  - IPsec TOE private key.
    - Please refer to FCS_IPSEC_EXT.1.13 below.
- FCS_CKM.4: zeroizing keys in non-volatile storage
  - Section 2.2.2 of [SUPP] provides the commands needed to destroy keys in non-volatile storage.
- FCS_IPSEC_EXT.1.1: Changes to the SPD
  - The SPD can be managed using the instructions provided in section 2.2.6.1 of the [SUPP].
- FCS_IPSEC_EXT.1.4: changes to ESP algorithms
  - IPsec crypto algorithms for ESP algorithms can be changed using the instructions provided in section 2.2.6.3 of [SUPP].
- FCS_IPSEC_EXT.1.6: changes to IKE algorithms
  - IPsec crypto algorithms for IKE algorithms can be changed using the instructions provided in section 2.2.6.5 of [SUPP].
- FCS_IPSEC_EXT.1.7, 1.8: changes to rekey limits for IKE and CHILD SAs
  - IPsec rekey limits for IKE SA and CHILD SA can be changed using the instructions provided in section 2.2.6.6 of [SUPP].
- FCS_IPSEC_EXT.1.11: changes to IKE DH groups
  - IPsec IKE SA DH algorithms can be changed using the instructions provided in section 2.2.6.8 of [SUPP].
- FCS_IPSEC_EXT.1.13: changing the certificate used for peer authentication
  - Section 2.2.6.9 of [SUPP] permits the administrator to adjust the TOE's IPsec authentication certificate.
- FCS_IPSEC_EXT.1.14: changing the reference IDs used to identify peers
  - Section 2.2.6.10 in [SUPP] describes the commands to set the reference identifiers.
- FCS_NTP_EXT.1.2: changes to time due to NTP stepping adjustments and changing the authentication key
  - Section 2.2.5 of [SUPP] permits the administrator to enable NTP and to configure it for authentication using an explicit key.
- FCS_NTP_EXT.1.4: adding, removing, changing NTP servers
  - Section 2.2.5 of [SUPP] provides instructions to permit the administrator to add, remove, and change multiple NTP servers.
- FCS_SSHS_EXT.1.2: changes to user account's SSH public keys
  - Using the mgmt-user CLI command as described in section 2.2.8 of [SUPP], administrators can associate SSH public keys to administrative users.
- FCS_SSHS_EXT.1.4: Changes to the ciphers for SSH server
  - Section 2.2.8 of [SUPP] indicates the set of ciphers are non-configurable.
- FCS_SSHS_EXT.1.5: changes to the SSH host key
- FCS_SSHS_EXT.1.6: Changes to MACs
  - [CLI] for the "ssh" command informs the administrator that SSH authentication supports hmac-sha1, and hmac-sha2-256 by default.
- FCS_SSHS_EXT.1.7: changes to KEXs
  - The key exchange algorithms are configured during the initial configuration by specifying the "ssh disable-kex dh" CLI command,

as per [SUPP] section 2.2.8 and then remains in the evaluated configuration.

- FCS_SSHS_EXT.1.8: changes to rekey limits
  - The limits are non-configurable as per section 2.2.8 of [SUPP].
- FCS_TLSS_EXT.1.1: changes to ciphersuites
  - Section 2.2.9 of the [SUPP] indicates that no configuration is required to set the permitted cipher suites.
- FCS_TLSS_EXT.1.3: changes to web server key/algorithm
  - Section 2.2.9 of [SUPP] permits the administrator to adjust the WebUI server cert. Server certificates can make use of both RSA and ECDSA key types.
- FIA_AFL.1.1: changes to authentication failure limits
  - Section 2.4.1 of [SUPP] provides the necessary information.
- FIA_AFL.1.2: changes to unlock timeout
  - Section 2.4.1 of [SUPP] provides the necessary information.
- FIA_PMG_EXT.1.1: changes to password complexity rules
  - Section 2.4.2 of [SUPP] provides the necessary information.
- FIA_X509_EXT.1/Rev, /ITT: changes to the trust store
  - The trust store manageability is described in section 2.4.6 of [SUPP].
- FIA_X509_EXT.3.1: generating a CSR, changes to the trust store
  - Section 2.4.7 of [SUPP] describes the process to create a CSR; section 2.4.6 of [SUPP] describes the manageability of the trust store.
- FMT_MOF.1.1/ManualUpdate: Initiating a change to the software/firmware
  - There is no TSF data requiring configuration as per section 2.5.1 of [SUPP].
- FMT_MOF.1/Services: starting and stopping the claimed services:
  - Logging can be started and stopped as per section 2.5.2 of the [SUPP]. The process to do so is described in section 2.1.4 of [SUPP].
- FMT_MTD.1/CoreData: restricting TSF data to appropriate admins
  - Provided a TOE user with the role of “root” via the “mgmt-user” CLI command provides them with the ability to service the TOE. This information is provided in [SUPP] section 2.5.3.
- FMT_MTD.1/CryptoKeys: restricting crypto key management to appropriate admins
  - Provided a TOE user with the role of “root” via the “mgmt-user” CLI command provides them with the ability to service the TOE. This information is provided in [SUPP] section 2.5.4.
- FMT_SMF.1.1: All management functions are represented within this list already
- FMT_SMF.1 /VPN: All management functions are represented within this list already
- FPF_RUL_EXT.1.2: defining packet rules,
  - Please refer to section 2.6.1 of the [SUPP].
- FPF_RUL_EXT.1.4: associating those rules with interfaces,
  - Please refer to section 2.6.1 of the [SUPP].
- FPF_RUL_EXT.1.5: altering the order of rules
  - Please refer to section 2.6.1 of the [SUPP].
- FPT_STM_EXT.1.2: manual changes to the time, enabling/disabling NTP
  - Manual changes to time are described in section 2.7.5 of [SUPP].
  - Enabling and disabling NTP is described in section 2.7.5 of [SUPP].
- FTA_SSL_EXT.1, FTA_SSL.3: configure an idle period
  - The idle timeout for local CLI and SSH CLI are unified in a single configuration option described in sections 2.8.1 and 2.8.3 of [SUPP].
  - WebUI interface timeout is described in section 2.8.1 of [SUPP]

- FTA_TAB.1: configure the banner
  - Section 2.4.5 of [SUPP] provides the instructions to set the TOE banner.
- FTP_ITC.1, FTP_ITC.1/VPN: configure the channel properties for all claimed channels:
  - IPsec channels for logging server, RADIUS, TACACS+ and NTP (if desired) are managed using the instructions for IPsec provided in section 2.2.6 of [SUPP].
  - Note that the connection between RAP and MC is covered under section 2.3.1 of [SUPP].
- FTP_TRP.1/Admin: configure the state of the remote administrative paths:
  - CLI over SSH
    - As per section 2.9.2 of [SUPP], no additional configuration for accessing the CLI over SSH is required beyond those needed for configuring the properties of the SSH channel or for configuring SSH public key access to users via the commands already mentioned in FCS_SSHS_EXT.1.2 above.
  - Web UI over HTTPS/TLS
    - As per section 2.9.2 of [SUPP], no additional configuration for accessing the WebUI over HTTPS/TLS is required beyond those needed for configuring the properties of the TLS channel already mentioned in FCS_TLSS_EXT.1.3 above.

381 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.

Findings:	The TOE supports handling of X.509v3 certificates and provides a trust store. Section 2.4.6 of the [SUPP] describes how to load and manage certificates.

3.5.4.3 Tests

382 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.

3.5.5 FMT_MTD.1/CryptoKeys Management of TSF Data

3.5.5.1 TSS

383 For distributed TOEs see chapter 3.5.1.1.

Findings:	TOE is distributed. Please refer to section 3.5.1.1.

384 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.

Findings:	N/A: The TOE is a distributed TOE

3.5.5.2 Guidance Documentation

385 For distributed TOEs see chapter 3.5.1.2.

Findings:	TOE is distributed. Please refer to section 3.5.1.2.

386 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.

Findings:	N/A: The TOE is a distributed TOE

3.5.5.3 Tests

387 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.

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

High-Level Test Description
Attempt to generate/import a certificate as the read only user and show that it is not successful. Successfully generating/importing a certificate with prior authentication as Security Administrator was previously performed as part of FIA_X509_EXT.3 Test 2 testing activities.
Findings: PASS – The evaluator confirmed that the readonly user is unable to generate/import crypto keys. Successfully generating/importing a certificate with prior authentication as Security Administrator was previously performed as part of FIA_X509_EXT.3 Test 2 testing activities.

3.5.6 FMT_SMF.1 Specification of Management Functions

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

3.5.6.1 TSS (containing also requirements on Guidance Documentation and Tests)

390 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).

Findings:	See section 3.5.1.1 above.

391 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.

Findings:	Section 6.5.5 of the [ST] states the TOE provides the administrator with local and remote interfaces to manage all security functions identified in this Security Target. Section 2.4.5 of [SUPP] iterates that the local serial connection should remain physically local and not be redirected to remote consoles.

392 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.

Findings:	The interaction of TOE components can be configured via the Mobility Controller per FCO_CPC_EXT.1 and as described in section 6.2.1 of the [ST]. The interactions between the Mobility Controller (MC) and the Remote Access Point (RAP) devices are described in the [SUPP] in section 2.3.1.

3.5.6.2 Guidance Documentation

393 See section 3.5.6.1.

3.5.6.3 Tests

394 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.

High-Level Test Description
The evaluator confirmed that all in-scope management functions can be tied back to testing performed in other AAs.
Findings: PASS – The evaluator confirmed all in-scope management functions were tied back to testing performed in other AAs. No additional testing was found to be required to cover other claimed management functions.

3.5.7 FMT_SMR.2 Restrictions on security roles

3.5.7.1 TSS

395 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.

Findings:	Section 6.5.6 of the [ST] indicates that the TOE provides the administrator role that corresponds to the Security Administrator role specified in the NDcPP. The administrator can manage all aspects of the TOE locally or remotely using the CLI or through the GUI.

3.5.7.2 Guidance Documentation

396 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.

Findings:	Section 2.4.5 of the [SUPP] indicates that users can authenticate via serial, SSH or the WebUI interface. [SUPP] provides information needed to establish local console access for the controllers and on configuring the SSH and WebUI services appropriate for the evaluated configuration. For more information on the remote interface configuration, please refer to FCS_SSHS_EXT.1 and FCS_TLSS_EXT.1, respectively. As the negotiated algorithms are described in full in the set of administrative guidance documents (primarily in [SUPP]), an administrator can easily configure their remote clients to interact. There are no specific or obscure requirements on the remote clients.

3.5.7.3 Tests

397 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.

High-Level Test Description
Verify that all supported administrative interfaces are exercised during the evaluation.
Findings: PASS - All interfaces are tested in the course of performing other tests.

398

3.6 Protection of the TSF (FPT)

3.6.1 FPT_APW_EXT.1 Protection of Administrator Passwords

3.6.1.1 TSS

399 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.

Findings:	Section 6.7.4 of the [ST] indicates that passwords are not stored in plaintext but rather stored in flash using a SHA1 hash. The TOE does not provide any interfaces to view plaintext passwords.

3.6.2 FPT_ITT.1 and FPT_ITT.1/Join

3.6.2.1 FPT_ITT.1 Basic internal TSF data transfer protection

400 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.

3.6.2.1.1 TSS

401 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.

Findings:	[ST] section 6.7.2: All data transmitted between TOE components is protected from disclosure and modification by using IPsec. IPsec VPN tunnels are established between Aruba Remote Access Points and Aruba Mobility Controllers. The IPSec protocol is specified in the SFR for this requirement in section 5.3.7 of the [ST].

3.6.2.1.2 Guidance Documentation

402 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.

Findings:	Section 2.3.1 of the [SUPP] provides a high-level description of the steps needed to ensure an ongoing operational connection between a RAP device and the MC. Of those steps, bullet point 3 states to "Establish the connection between the Remote AP and Controller". This step is further broken out in section 2.3.1 to provide the high-level description to create the appropriate IPSec connection channel. An administrator could use these steps to adjust the properties of the IPSec connection channel as desired. If the connection is unintentionally broken, [SUPP] section 2.3.1 claims there are no active steps that a security administrator would need to take. "*If the operational channel (FPT_ITT.1) is unintentionally broken, the connection will be retried automatically.*"

3.6.2.1.3 Tests

403 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.

High-Level Test Description
With a RAP no longer connected to the Controller, show that the RAP, once added using the whitelist, can then join with the Controller and use the operational channel.
Findings: PASS – The TOE is configured as per the guidance documentation in [SUPP] section 2.2.6 specifically for RAP devices using IPsec dynamic-maps, appropriate IKEv2 and IPSec algorithms, and using the RAP device "whitelist" to permit the connection to occur. The evaluator showed that when the guidance was followed, the RAP device was able to successfully connect to the Controller and that the protocol seen during operational use was IPsec.

b) Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

High-Level Test Description
When interacting with the AP, show that information traversing between the controller to the RAP is protected by IPsec.
Findings: PASS – The evaluator confirmed that the information sent between the RAP and the Controller is not sent in plaintext.

c) Test 3: 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 nonequivalent 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.

High-Level Test Description
With the AP connected to the controller and traffic transiting the network, pull the cable quickly and then replace it to perform a short disconnect test. Show that the connection is interrupted, but when the AP reconnects, traffic continues to flow as protected. With the AP connected to the controller and traffic transiting the network, pull the cable and leave it out for about 1 minute. Show that the connection is interrupted, but when the AP reconnects, traffic continues to flow as protected.
Findings: PASS – The evaluator confirmed that when network interruptions occur, the connection is automatically re-initiated without leaking any plaintext.

404 Further assurance activities are associated with the specific protocols.

3.6.2.2 FPT_ITT.1/Join Basic internal TSF data transfer protection during Registration

NOTE:	This section has been added as per [PP] Application Note 48. While the TOE implements a single IPsec stack that handles both operational and component registration, the times at which they are invoked differ.

405 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.

3.6.2.2.1 TSS

406 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.

Findings:	[ST] section 6.7.2: All data transmitted between TOE components is protected from disclosure and modification by using IPsec. IPsec VPN tunnels are established between Aruba Remote Access Points and Aruba Mobility Controllers. The registration channel (denoted at FPT_ITT.1/Join) is protected using IPsec. Aruba Remote Access Points initiate communication with Aruba Mobility Controllers using IPsec VPN tunnels. The IPSec protocol is specified in the SFR for this requirement in section 5.3.7 of the [ST].

3.6.2.2.2 Guidance Documentation

407 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.

Findings:	Section 2.3.1 of the [SUPP] provides a high-level description of the steps needed to establish a connection between a RAP device and the MC. Of those steps, bullet point 3 states to "Establish the connection between the Remote AP and Controller". This step is further broken out in section 2.3.1 to provide the high-level description to create the appropriate IPSec connection channel. If the connection is unintentionally broken, [SUPP] section 2.3.1 claims there are no active steps that a security administrator would need to take. "*If during the provisioning process the connection between the Controller and Remote AP is interrupted, the process will halt and would resume once connectivity is re-established*."

3.6.2.2.3 Tests

408 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.

High-Level Test Description
With a RAP no longer connected to the Controller, show that the RAP, once added using the whitelist, can then join with the Controller and use the operational channel..
Findings: PASS – The TOE is configured as per the guidance documentation in [SUPP] section 2.2.6 specifically for RAP devices using IPsec dynamic-maps, appropriate IKEv2 and IPSec algorithms, and using the RAP device "whitelist" to permit the connection to occur. The evaluator showed that when the guidance was followed, the RAP device was able to successfully connect to the Controller and that the protocol seen during registration was IPsec.

b) Test 2: The evaluator shall ensure, for each communication channel with an authorized IT entity, the channel data is not sent in plaintext.

High-Level Test Description
Relying on the output from the previous test case, show that the traffic over the wire does not contain plaintext information.
Findings: PASS – The evaluator confirmed that the information sent between the RAP and the Controller during the joining process is not sent in plaintext.

c) Test 3: 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 nonequivalent 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.

High-Level Test Description
Starting with an unjoined RAP device, initiate a joining process with the controller and power on the AP. During the boot process, disconnect the AP from the network until the AP reboots due to running out of attempts. Confirm the controller has not yet accepted the RAP. Reconnect the AP to the controller as the device reboots. Once the AP starts to rejoin the controller and IPsec starts, disconnect the AP from the controller for 5 seconds to interrup the MAC layer and then reconnect. Show that the Remote AP continues to negotiate IPsec and does not transmit any plaintext information. Confirm the controller has accepted the RAP. Show that the traffic over the wire does not contain plaintext information.

High-Level Test Description
Findings: PASS – The evaluator confirmed that when network interruptions occur, the connection is automatically re-initiated without leaking any plaintext.

409 Further assurance activities are associated with the specific protocols.

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

3.6.3.1 TSS

410 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.

Findings:	Section 6.7.3 of the [ST] claims the TOE provides no interfaces that allow pre-shared, symmetric or private keys to be read. Section 6.3.4 describes how the pre-shared keys, symmetric keys and private keys are stored.

3.6.4 FPT_STM_EXT.1 Reliable Time Stamps

3.6.4.1 TSS

411 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.

Findings:	Section 6.7.8 of the [ST] claims the TOE has an internal battery-backed hardware clock that provides reliable time stamps used for auditing. The internal clock can be set by the administrator and can be synchronized with a time signal obtained from an external NTP server. The clock is used to provide a timestamp for audit records, and to support timing elements of cryptographic functions, certificate validity checks, session timeouts, and unlocking of administrator accounts locked as a result of authentication failure. When an external NTP server is used, the TOE can update its system time using either pre-shared keys or IPsec to provide trusted communication between itself and the NTP time source, depending on how it is configured.

NIAP TD0632

412 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.

Findings:	This option is not claimed in the [ST] and therefore is N/A.

3.6.4.2 Guidance Documentation

413 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.

Findings:	According to section 2.7.5 of the [SUPP], the Security Administrator can set the time on the Mobility Controller using either NTP or manual time changes using the CLI. When NTP is used, IPSec can be used to protect the communication path, though the NTP configuration also supports symmetric key-based NTP protections as per sections 2.2.5 and 2.7.5 of [SUPP]. When IPsec is used to protect the communication path, [SUPP] provides a summary of those instructions in section 2.2.6.

NIAP TD0632

414 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.

Findings:	This option is not claimed in the [ST] and therefore is N/A.

3.6.4.3 Tests

415 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.

High-Level Test Description
Using the CLI, change the date/time of the Mobility Controller. Using the CLI and the Web UI, show that the date/time reflects the change. Using the CLI, change the date/time of the Mobility Controller. Show that the date/time is adjusted on the connected RAP device.
Findings: PASS – The evaluator confirmed that after manually setting the date and time, the TOE correctly maintained the new date/time. The evaluator also confirmed that the RAP device synchronized with the time from the Controller.

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.

High-Level Test Description
Using the TOE interface, enable NTP to an NTP server in the test environment. Show that the TOE updates the date/time to synchronize with the NTP server's time. Also show that the date/time is adjusted on the connected RAP devices.
Findings: PASS – The evaluator confirmed that once NTP is enabled on the TOE and properly synced with an NTP server, the TOE correctly updates the date/time to synchronize with the NTP server's time. The evaluator also confirmed that the RAP devices synchronized with the time from the Controller.

NIAP TD0632

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.

High-Level Test Description
The TOE is not a virtualized TOE and does not obtain the time from the underlying VS.
Findings: N/A

416 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.

High-Level Test Description
Set the date/time on the Mobility Controller. Then reboot the Remote Access Point and show that the audit messages are time-stamped with the synchronized time. It remains unambiguous how to relate time stamps in audit messages for the Remote Access Point to the Mobility Controller time stamps.
Findings: PASS – The evaluator found that time-stamps in audit records from distributed components maintaining independent time information were able to be interpreted unambiguously.

3.6.5 FPT_TST_EXT.1 TSF testing

3.6.5.1 TSS

417 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.

Findings: Section 6.7.5 of the [ST] details the self-tests. The Mobility Controller and Remote Access Points run a suite of self-tests during power-up, which includes demonstration of the correct operation of the hardware and the use of cryptographic functions to verify the integrity of TSF executable code and static data. The Mobility Controller and Remote Access Points run the suite of FIPS 140-2 validated cryptographic module self-tests during start-up or reboot. Conditional self-tests are also run during the course of normal operation.

A series of conditional self-tests are executed. Each of these are described in detail in the ST. The description of the tests are detailed enough such that the evaluator could easily understand what function was being exercised.

The described tests are sufficient to demonstrate that the TSF is operating correctly by verifying the integrity of the TSF and the correct operation of cryptographic components.

418 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.

Findings: Section 6.7.5 of the [ST] provided the information about which tests are executed on which components All of the tests in section 6.7.5 of the [ST] run on both the MC and the RAP components. They run on startup/reboot.

- ArubaOS OpenSSL Module:
  - AES Known Answer Tests (KAT)
  - Triple-DES KAT
  - RNG KAT
  - RSA KAT
  - ECDSA (sign/verify)
  - SHA (SHA1, SHA256 and SHA384) KAT
  - HMAC (HMAC-SHA1, HMAC-SHA256 and HMAC-SHA384) KAT
- ArubaOS Cryptographic Module
  - AES KAT
  - Triple-DES KAT
  - SHA (SHA1, SHA256, SHA384) KAT
  - HMAC (HMAC-SHA1, HMAC-SHA256, HMAC-SHA384) KAT
  - RSA (sign/verify)
  - ECDSA (sign/verify)
- ArubaOS Uboot BootLoader Module
  - Firmware Integrity Test: RSA 2048-bit Signature Validation
- Aruba Hardware Crypto Accelerator Known Answer Tests:
  - AES KAT
  - AES-GCM KAT
  - Triple DES KAT
  - HMAC (HMAC-SHA1, HMAC-SHA256, HMAC-SHA384) KAT

The following Conditional Self-tests are performed by the TOE:

- **Continuous Random Number Generator Test** –This test is run upon generation of random data by the switch's random number generators to detect failure to a constant value. The module stores the first random number for subsequent comparison, and the module compares the value of the new random number with the random number generated in the previous round and enters an error state if the comparison is successful.
- **Noise Source Health** – This test continuously measures the local frequency of occurrence of a sample value in a sequence of noise source

samples to determine if the sample occurs too frequently. Thus, the test is able to detect when some value begins to occur much more frequently than expected, given the source's assessed entropy per sample.

- **Bypass test**. Ensures that the system has not been placed into a mode of operation where cryptographic operations have been bypassed, without the explicit configuration of the cryptographic officer. To conduct the test, a SHA1 hash of the configuration file is calculated and compared to the last known good hash of the configuration file. If the hashes match, the test is passed. Otherwise, the test fails (indicating possible tampering with the configuration file) and the system is halted.
- **RSA Pairwise Consistency test**. When the TOE generates a public and private key pair, it carries out pair-wise consistency tests for both encryption and digital signing. The test involves encrypting a randomly-generated message with the public key. If the output is equal to the input message, the test fails. The encrypted message is then decrypted using the private key and if the output is not equal to the original message, the test fails. The same random message is then signed using the private key and then verified with the public key. If the verification fails, the test fails.
- **ECDSA Pairwise Consistency test**. See above RSA pairwise consistency test description.
- **Firmware Load Test**. This test is identical to the Uboot BootLoader Module Firmware Integrity Test, except that it is performed at the time a new software image is loaded onto the system. Instead of being performed by the BootLoader, the test is performed by the ArubaOS operating system. If the test fails, the newly loaded software image will not be copied into the image partition, and instead will be deleted.

3.6.5.2 Guidance Documentation

419 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.

Findings: Section 2.7.6 of [SUPP] states that the TOE will immediately halt operation and enter an error state if any self-test fails. Furthermore the section continues:

"*The error output of a failed self-test will appear as follows: "FIPS Aruba Cryptographic asymmetric key KAT failure, main: FIPS_powerupSelfTest failed." If a firmware image fails its integrity check, the TOE will load the previous image (if one is present). An error will be output during boot in this instance stating that the firmware validation failed.*

If the issue continues, the administrator should contact support at http://support.arubanetworks.com."

The description was found to be consistent with the TSS.

420 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.

Findings: In [SUPP] section 2.7.6, it states: "*The local console for Mobility Controller and RAP devices can be reviewed to determine which component has failed a self-test. Mobility Controllers maintain health status information for each connected RAP device. For*

	devices not marked as 'connected', an investigation and review of the local console can determine if the RAP device has experienced an issue."

3.6.5.3 Tests

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

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

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

422 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.

423 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.

424 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.

High-Level Test Description
Reboot the Mobility Controller and the RAP devices and watch the output on the serial consoles as the devices boot. Witness that cryptographic and firmware-integrity self-tests are executed successfully.
Findings: PASS – The TOE components executed the appropriate self-tests on reboot.

425

3.6.6 FPT_TUD_EXT.1 Trusted Update

3.6.6.1 TSS

426 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.

Findings:	Section 6.7.7 of the [ST] claims the TOE allows administrators to query the currently executing (using "show version" CLI command) and most recently installed versions of its firmware/software (using the "show image version" CLI command). For RAP devices, the currently executing image version can be viewed using the command "show ap image version". The Mobility Controller is capable of delayed activation and the RAP is not. Section 6.7.7 of the [ST] states that when the MC installs the new image, the administrator has the option of rebooting immediately for the update to take effect or may elect to continue operation with the currently executing version. For RAP devices, once the

	updated version is installed, the RAP will reboot immediately and the controller verifies that their versions now match. This ensures that TOE updates cannot lead to the situation where different TOE components are running different software versions.

427 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.

Findings:	According to section 6.7.7 of the [ST], Administrators can update the TOE executable code using image files manually downloaded from the Aruba support portal. The administrator may initiate an update from either the WebUI or CLI. Upgrade instructions are documented in the release notes for each software release, which will be posted in the same directory as the image file on the support portal. The TOE computes a SHA256 sum over the entire software image. Using the RSA public key stored internally in the product, the computed hash is compared against the hash obtained from the digital signature included in the software image to ensure it is authentic. Software images are verified at the time of receipt (before writing to the flash) and is also verified by the bootloader each time the TOE boots. RSA2048 and SHA256 are used to sign (and subsequently verify) the image. For the RAP components, section 6.7.7 of the [ST] further states that RAP obtain TOE updates directly from their managing Controller. Once the Controller is updated, the RAP will detect – on next connection attempt – that an update is available. The RAP then downloads the update. A software image that is downloaded from the Aruba Mobility Controller is both verified at the time of receipt (before writing to the flash) and is also verified by the bootloader each time the TOE boots. For both the Controller and the RAP devices, any image with an invalid signature will not be copied by the TOE into the image partition. For the controller, upon successful verification, MC will install the new image and the administrator has the option of rebooting immediately for the update to take effect or may elect to continue operation with the currently executing version (delayed activation) For RAP devices, when the download completes, the TOE sends a message to the Aruba Mobility Controller, informing it that the TOE has either successfully downloaded the new software version, or that the preload has failed for some reason (one of these reasons includes a signature verification failure). Once the updated version is installed, the RAP will reboot immediately and the controller verifies that their versions now match.

428 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.

Findings:	Selection not applicable.

429 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.

Findings:	Section 6.7.7 of the [ST] describes the process. The Mobility Controller is updated manually by the administrator. In contrast, the Remote Access Points obtain TOE updates from their managing Controller. When a RAP connects to the controller, the controller checks the version of the RAP to ensure it is the same. IF the version do not match, the controller will force the RAP to update to the same version as the controller. The RAP will then reboot and the controller verifies that their versions now match. This ensures that TOE updates cannot lead to the situation where different TOE components are running different software versions. This ensures the continuous proper functioning of the distributed TOE components as the Controller is being updated. Section 6.7.7 of the [ST] describes that digital signatures are used to validate the TOE images during the installation process as well as upon each boot cycle.

430 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.

Findings:	N/A: The TOE uses a digital signature to protect the trusted update mechanism.

3.6.6.2 Guidance Documentation

431 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.

Findings:	Section 2.7.8 of [SUPP] provides CLI commands to query the currently active version of the software/firmware of the TOE (for the MC, it is `show version`; for the RAPs, it is `show ap image version`). The TOE claims delayed activation for the MC: according to the instructions in [SUPP] section 2.7.8, the administrator can choose whether the device should be rebooted when the image file is transferred. Section 2.7.8 of [SUPP] provides the MC CLI command to query the loaded, but currently inactive version (`show image version`).

432 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.

Findings:	Section 2.7.8 of [SUPP] states that the TOE uses digital signatures to validate update images. The TOE will refuse to install the images if the signature fails to validate. If digital signature verification succeeds, the TOE will proceed with the installation of the image. This process is used for both updates to the MC and the RAP components. This description corresponds to the information provided in section 6.7.7 of the [ST].

433 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.

Findings:	Published hash is not used by the trusted update mechanism.

434 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 user; it does not need to give information about the internal communication that takes place when applying updates.

Findings:	According to section 2.7.8 of [SUPP], the MC can have its current executing version queried using the “show version” CLI command. The attached managed devices (RAP) components can have their current executing image version queried using the “show ap image version” CLI command. For the MC only, because it claims delayed activation, the MC can also query the loaded, but not yet active versions of the software using the CLI command “show image version”. The RAP devices do not claim delayed activation. As per [SUPP] section 2.7.8, the MC is the primary means to upgrade the distributed TOE. The MC can receive updates to itself; the RAP components receive images that are uploaded by the Security Administrator to the MC (which are then pushed down to the RAP devices). In all cases, if a digital signature failure occurs, the image will not be applied. If digital signature verification fails, the [SUPP] section 2.7.8 indicates the TOE will enter into an error state. The TOE’s error state will allow direct console access only, where an administrator can change to a new file partition or TFTP a new image and re-boot. Because the MC and the RAP must be running the same firmware versions to operate together, upgrade failures and MC to RAP connectivity issues go hand-in-hand. The [ADMIN] provides a reasonable amount of troubleshooting with respect to upgrade issues, such as in the section “Access Points > AP Discovery Logic > Troubleshooting the AP Discovery Logic”.

435 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.

Findings:	Please refer to the previous work unit above which describes how all TOE components are updated and show the signature verification process works. As the upgrade mechanism is independent between the MC and the RAP components, each component can upgrade itself while the others continue to operate. Specifically, as per [SUPP] section 2.7.8 RAP components cannot operate within the managed access point network until the software/firmware on the APs matches that on the Mobility Controller.

436 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.

Findings:	The TOE does not claim the use of certificate-based mechanisms for software update digital signature verification.

3.6.6.3 Tests

437 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.

High-Level Test Description
Get the current version of the TOE. Attempt to install a legitimate version of the TOE for an upgrade. After the install, get the current version of the TOE and ensure it is consistent with the newly installed version.
Findings: PASS - The evaluator confirmed the TOE displayed its current version, successfully loaded a valid update, displays both the current version and most recently installed version. The evaluator activated the update and performed the version verification again and the TOE displayed the current version and updated version that matched.

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.

High-Level Test Description
Attempt to install a bad image, an unsigned image and a badly signed image for firmware upgrades. After each attempt, get the current version of the TOE using all available means and ensure they are consistent.
Findings: PASS - The evaluator confirmed that the TOE did not install illegitimate updates and the current version did not change.

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 user 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. 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.

High-Level Test Description
The TOE does not support verification of published hashes.
Findings: N/A

438 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.

439 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).

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

High-Level Test Description
Tests 1 and 2 were executed for both the Mobility Controller and Remote Access Points. Test 3 was not applicable to any component.
Findings: PASS - The evaluator confirmed that tests 1 and 2 passed for all components in the distributed TOE.

3.7 TOE Access (FTA)

3.7.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.7.1.1 TSS

441 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.

Findings:	Section 6.8.1 of the [ST]: Local inactive administrator sessions on the TOE are terminated after the configured timeout period. Session timeout can be configured for local CLI administrative sessions. To define a timeout interval for a CLI session, use loginsession timeout <*value*> where value is from 1 to 3600 seconds.

3.7.1.2 Guidance Documentation

442 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.

Findings:	Section 2.8.3 of [SUPP] indicates that idle local administrative sessions are disconnected based on an administrator-configurable inactivity time period. The CLI instructions to set this timeout are provided.

3.7.1.3 Tests

443 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 re-authentication is needed when trying to unlock the session.

High-Level Test Description
Log into the serial console of the Mobility Controller. Configure the idle timeout to be 1 minute. Log out and log back in again. Wait 1 minute. Confirm the TOE has logged out the user. Configure the idle timeout to be 3 minutes. Log out and log back in again. Wait 3 minutes and confirm the TOE has logged out the user.
Findings: PASS – The evaluator confirmed the TOE terminates local console sessions when the inactivity timeout period is reached.

3.7.2 FTA_SSL.3 TSF-initiated Termination

3.7.2.1 TSS

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

Findings:	Section 6.8.2 of the [ST]: Inactive remote administrator sessions on the TOE are terminated after the configured timeout period. Session timeout thresholds are configurable for the WebUI and CLI interfaces. To define a timeout interval for a WebUI session, use the command: web-server profile and session-timeout <session-timeout> where the session-timeout value can be any number of seconds from 30 to 3600. For the CLI, use loginsession timeout <value> where value is from 1 to 3600 seconds.

3.7.2.2 Guidance Documentation

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

Findings:	Section 2.8.1 of [SUPP] indicates that idle remote administrative sessions are disconnected based on an administrator-configurable inactivity time period. Both CLI and Web UI instructions to set this timeout are provided.

3.7.2.3 Tests

446 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.

High-Level Test Description
Log into the SSH CLI of the Mobility Controller. Configure the idle timeout to be 1 minute. Log out and log back in again. Wait 1 minute. Confirm the TOE has logged out the user. Configure the idle timeout to be 5 minutes. Log out and log back in again. Wait 5 minutes and confirm the TOE has logged out the user. Perform the same series of steps, but this time use the Web UI with timeouts of 1 minute and 10 minutes.
Findings: PASS - The evaluator confirmed that the TOE terminates remote sessions (both CLI and Web UI) when the inactivity timeout period is reached.

3.7.3 FTA_SSL.4 User-initiated Termination

3.7.3.1 TSS

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

Findings:	Section 6.8.3 of the [ST]: The TOE allows users to terminate their own sessions by providing the “exit” command at the CLI or by using an appropriate “log out” button in the Web UI.

448

3.7.3.2 Guidance Documentation

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

Findings:	Section 2.8.2 of [SUPP] indicates that Security Administrators can terminate their own session by exiting from the CLI or logging out of the WebUI. Within [ADMIN], in section "Mobility Conductor Configuration Hierarchy > Mobility Conductor User Interface > Navigation Model", there are directions provided for how to "logout" of the WebUI. Within [CLI], the "exit" command is used to exit the CLI. Note the use of the term "Mobility Conductor" in the [ADMIN] guide which is a different product line. The Mobility Conductor and the Mobility Controller user interfaces are laid out identically and therefore the evaluator accepted the above.

3.7.3.3 Tests

450 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.

High-Level Test Description
Log into the serial console Log out using the TSFI previous discussed. Verify that the session has been terminated.
Findings: PASS - The evaluated confirmed that the local console session is terminated when the administrator logs out.

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.

High-Level Test Description
Log into the SSH and Web UI interfaces. Log out of each session.
Findings: PASS - The evaluated confirmed that the remote administrative sessions at the Web UI and remote CLI are terminated when the administrator logs out.

3.7.4 FTA_TAB.1 Default TOE Access Banners

3.7.4.1 TSS

451 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).

Findings:	The [ST] in section 6.4.3 states that administration can occur to the Mobility Controller using web GUI or CLI. In addition, section 6.4.4 of the [ST] also indicates local administrative support using a local serial console. Section 6.8.4 of the [ST]: The TOE displays an advisory warning banner regarding use of the TOE prior to establishing an administrator session. The administrator can configure the warning message displayed in the banner using the CLI or the GUI. The banner will be displayed when accessing the CLI locally via the console or remotely via SSH and when accessing the GUI. The configured message is identical across interfaces.

3.7.4.2 Guidance Documentation

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

Findings:	Section 2.4.5 of [SUPP] provides the CLI commands and GUI affordances necessary to configure the banner message.

3.7.4.3 Tests

453 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.

High-Level Test Description
Log into the Web interface and change the banner to a random string. Log into fresh sessions for all interactive interfaces and show that the banner was modified and is presented prior to I&A. Log into the CLI and change the banner to a random string. Log into fresh sessions for all interactive interfaces and show that the banner was modified and is presented prior to I&A.
Findings: PASS – The evaluator confirmed that the administrator is able to configure the warning message and that the warning message is displayed prior to authentication at each administrative interface.

3.8 Trusted path/channels (FTP)

3.8.1 FTP_ITC.1 Inter-TSF trusted channel

3.8.1.1 TSS

454 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.

Findings:	Section 6.9.1 of the [ST] states that the TOE uses the IPsec/IKE protocol with certificates to establish VPN tunnels and to establish trusted channels between the Controller and the external authentication server, syslog server, and NTP server. VPN tunnels can be established between Aruba Mobility Controllers and Remote Access Points. The use of certificates within IPSec provides for assured identification of the remote non-TSF endpoint. These IPsec channels are peer-to-peer connections whereby the TOE can act as either the server or the client.

3.8.1.2 Guidance Documentation

455 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.

Findings:	In section 2.9.1 of [SUPP], the TOE only uses IPsec as the allowed protocol for communicating securely with IT entities such as (a) a syslog server, (b) a RADIUS or TACACS+ server, (c) an NTP server, and (d) remote VPN gateways/peers. Section 2.9.1 of [SUPP] provides information on the various parts of an IPsec tunnel as implemented within the TOE. It describes how an administrator would define and configure each of the policies and profiles to ensure that an IPsec tunnel is established. Finally, it notes that IP addresses that fall within the protected traffic selectors must be used when defining protected services. Instructions to configure the IP addresses for the in-scope trusted channels are provided: - for syslog appear in [SUPP] section 2.1.4 (`logging <ip address>`); - for RADIUS in [SUPP] section 2.4.5, which points to [ADMIN] section “Enabling RADIUS Server Authentication” (which provides both CLI and GUI instructions to provision the correct IP address); - for TACACS+ in [SUPP] section 2.4.5, which points to [ADMIN] section “Configuring a TACACS+ Server" (which provides both CLI and GUI instructions to provision the correct IP address); - for NTP appear in [SUPP] section 2.2.5 (`server {<ip>|<ipv6>} ...`) - for remote VPN Gateways/Peers – these are TOE environment specific and no specific TOE commands apply other than to ensure the correct traffic selectors are configured during construction of the IKEv2 profile. The TOE is responsible for automatically re-establishing a connection when unintentionally broken.

3.8.1.3 Tests

456 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.

457 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.

High-Level Test Description
Test 1, Test 2 and Test 3 were done in conjunction. 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.
Findings: PASS – The TOE maintains an IPSec trusted channel to the remote audit, authentication, and NTP servers. The trusted channel is set up as per the evaluated configuration and is constantly tested throughout the evaluation. The trusted channel is specifically tested as part of FCS_IPSEC_EXT.1. FPT_ITC.1 Test 3 the evaluator confirmed that the trusted channel is successfully established.

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.

High-Level Test Description
Test 1, Test 2 and Test 3 were done in conjunction. Ensure the trusted channel can be initiated from the TOE.
Findings: PASS – FTP_ITC.1 Test 3 Step 5 shows the TOE can initiate the trusted channel.

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

High-Level Test Description
Test 1, Test 2 and Test 3 were done in conjunction. All trusted channels make use of the same underlying IPsec cryptographic service for transport. As a result, we will pick one channel to test. Enable and disable logging to the remote syslog server and then show that the connection is successful and that (encrypted) information is sent over the connection.
Findings: PASS – FTP_ITC.1 Test 3 Step 4 shows that 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.

High-Level Test Description
With the TOE logging at a high rate of speed over a configured IPsec channel, physically disconnect and then reconnect the ethernet cable connecting the controller to the network. Show that the traffic is uninterrupted and remains protected. With the TOE logging at a high rate of speed over a configured IPsec channel, physically disconnect the ethernet cable connecting the controller to the network and wait for the controller to detect that the peer is dead. Reconnect the controller and show that the traffic remains protected.
Findings: PASS - The evaluator confirmed that the TOE did not send trusted channel data (IPsec) in plaintext when the channel was disrupted for the network layer or application layer timeout durations.

Further assurance activities are associated with the specific protocols.

458 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.

High-Level Test Description
Test steps were provided in the previous test case for all distributed TOE components.
Findings: PASS – All TOE components are exercised as described in the previous test cases. Only the Mobility Controller needs to meet the requirements for FTP_ITC.1. The Controller <-> RAP channel falls under FPT_ITT.1 and under FCS_IPSEC_EXT.1/ITT (an iteration of FCS_IPSEC_EXT.1).

459 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.

Findings:	The developer provided sufficient information regarding application layer timeout settings for the evaluator to perform FTP_ITC.1 Test 4.

3.8.2 FTP_TRP.1/Admin Trusted Path

3.8.2.1 TSS

460 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.

Findings:	Section 6.9.2 of the [ST]: The TOE uses HTTPS/TLS to offer secure remote WebUI-based administration and SSH to offer a secure remote administration CLI. Administrators can initiate a remote session that is secured (from disclosure and modification) using NIST-validated cryptographic operations, and all remote security management functions require the use of this secure channel. Each connection can be tunneled over IPsec for an additional layer of security. The evaluator reviewed the SFR in section 5.3.9 of the [ST] and found the use of HTTPS/TLS and SSH to be consistent with the requirement. The use of IPsec as a means to tunnel the already protected trusted paths is also consistent with the claims in the SFR.

3.8.2.2 Guidance Documentation

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

Findings:	According to section 2.9.2 of [SUPP], the TOE makes use of both SSH and HTTPS/TLS to offer secure remote administration. Of specific note, "*No configuration is required on RAP devices once all components have been placed into evaluated configuration, therefore no admin interfaces are available on RAP devices once this state is achieved.*" Information about tunnelling administrative sessions over IPsec is provided in section 2.9.2 of the [SUPP]. While the details of how to configure the IT environment workstation hosting the SSH client or Web browser is out of scope, the means to establish the workstation as a viable peer to the TOE for using IPSec follows the same process as outlined in [SUPP] 2.9.1.

3.8.2.3 Tests

462 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.

High-Level Test Description
Connect to the SSH CLI using an SSH client and show that the connection is successful and that data is not transferred in plaintext. Connect to the Web UI using a web browser and show that the connection is successful and that data is not transferred in plaintext.
Findings: PASS - The trusted paths are the TLS/HTTPS Web UI and SSH Remote CLI, which both are set up as per the evaluated configuration. They are constantly tested throughout the evaluation. TLS is tested in FCS_TLSS_EXT.1, and SSH is tested in FCS_SSHS_EXT.1.

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

High-Level Test Description
Ensure that the trusted channel data is not sent in plaintext.
Findings: PASS – FCS_TLSS_EXT.1 and FCS_SSHS_EXT.1 testing shows the TOE successfully establishing trusted paths. The remote trusted path client is a known good TLS or SSH client implementation, so the successful transfer of channel data shows the channel data is not sent in plaintext (i.e., the client would terminate the connection due to decryption and/or integrity errors if the data was sent in plaintext).

463 Further assurance activities are associated with the specific protocols.

464 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.

High-Level Test Description
Ensure that components in the distributed TOE that provide a trusted path are tested.
Findings: PASS – Only the Mobility Controller has a trusted path. The RAP devices have no administrative login.

4 Evaluation Activities for the VPN Gateway PP-Module

4.1 Security Audit (FAU)

4.1.1 FAU_GEN.1/VPN Audit data generation (MOD VPNGW)

4.1.1.1 TSS

465 The evaluator shall examine the TSS to verify that it describes the audit mechanisms that the TOE uses to generate audit records for VPN gateway behavior. If any audit mechanisms the TSF uses for this are not used to generate audit records for events defined by FAU_GEN.1 in the Base-PP, the evaluator shall ensure that any VPN gateway-specific audit mechanisms also meet the relevant functional claims from the Base-PP. For example, FAU_STG_EXT.1 requires all audit records to be transmitted to the OE over a trusted channel. This includes the audit records that are required by FAU_GEN.1/VPN. Therefore, if the TOE has an audit mechanism that is only used for VPN gateway functionality, the evaluator shall ensure that the VPN gateway related audit records meet this requirement, even if the mechanism used to generate these audit records does not apply to any of the auditable events defined in the Base-PP.

Findings:	Section 6.1.1 of the [ST] describes the mechanism used by the TOE to generate audit records. The TSS does not stipulate there are separate mechanisms needed to fulfil the audit log generation for the VPN Gateway functionality; rather, the same mechanism is used as per the Base-PP.

4.1.1.2 Guidance

466 The evaluator shall examine the operational guidance to verify that it identifies all security-relevant auditable events claimed in the ST and includes sample records of each event type. If the TOE uses multiple audit mechanisms to generate different sets of records, the evaluator shall verify that the operational guidance identifies the audit records that are associated with each of the mechanisms such that the source of each audit record type is clear.

Findings:	Section 2.1.1 of the [SUPP] includes relevant examples of the claimed functions. The evaluator cross-referenced the total set of SFR claims in the [ST] against the table of auditable messages in 2.1.1 of the [SUPP] and found it to be complete.

4.1.1.3 Tests

467 The evaluator shall test the audit functionality by performing actions that trigger each of the claimed audit events and verifying that the audit records are accurate and that their format is consistent with what is specified in the operational guidance. The evaluator may generate these audit events as a consequence of performing other tests that would cause these events to be generated.

High-Level Test Description
These activities are performed within each of the test cases that required audit messages be generated.

High-Level Test Description
Findings: PASS - The evaluator found that audit messages generated by the various components were identified by the component which either generated it or responsible for witnessing it.

4.2 Cryptographic Support (FCS)

4.2.1 FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication) (MOD VPNGW)

4.2.1.1 TSS

The evaluator shall check to ensure that the TSS describes how the key-pairs are generated.

Findings:	Section 6.3.2 of the [ST] claims that for keys used for IKE peer authentication, the TOE supports cryptographic key generation for RSA schemes using key sizes of 2048 bits, and ECC schemes using NIST curves P-256 and P-384.

In order to show that the TSF implementation complies with FIPS PUB 186-4, the evaluator shall ensure that the TSS contains the following information:

- The TSS shall list all sections of Appendix B to which the TOE complies.
- For each applicable section listed in the TSS, for all statements that are not "shall" (that is, "shall not", "should", and "should not"), if the TOE implements such options it shall be described in the TSS. If the included functionality is indicated as "shall not" or "should not" in the standard, the TSS shall provide a rationale for why this will not adversely affect the security policy implemented by the TOE;
- For each applicable section of Appendix B, any omission of functionality related to "shall" or “should” statements shall be described;

Findings:	Section 6.3.2 of the [ST] claims conformance with FIPS PUB 186-4 Appendix B.3, and Appendix B.4. No deviations from the required functionality via the “shall” statements are claimed, nor are any “shall not”, "should" or “should not” statements claimed.

Any TOE-specific extensions, processing that is not included in the Appendices, or alternative implementations allowed by the Appendices that may impact the security requirements the TOE is to enforce shall be described.

Findings:	Section 6.3.2 of the [ST] does not list any such extensions.

4.2.1.2 Guidance

468 The evaluator shall check that the operational guidance describes how the key generation functionality is invoked, and describes the inputs and outputs associated with the process for each signature scheme supported. The evaluator shall also check that guidance is provided regarding the format and location of the output of the key generation process.

Findings:	In section 2.2.1 of the [SUPP], the administer is informed that they "…can invoke the use of RSA and ECDSA during generation of certificates used for X.509." Information on the process for generating keys for X.509 is found in [SUPP] in section 2.4.7. According to [ADMIN], the parameters associated with the RSA and EC signature schemes are documented in "Management Access > Managing Certificates > Obtaining Server Certificate" which provides both GUI and CLI commands. Specifically, the CLI commands are generically: `crypto pki csr {rsa key_len <key_val> |{ec curve-name <key_val>} common_name <common_val> country <country_val> state_or_province <state> city <city_val> organization <organization_val> unit <unit_val> email <email_val>` Where, according to the same section in [ADMIN], RSA key_len can be 2048 or 4096 bits and EC curve-name is secp256r1 or secp384r1. Note that section 2.2.1 of the [SUPP] further restricts RSA key lengths to only be 2048 bits. According to section 2.4.7 of the [SUPP], CSR requests can be exported to the console screen by the administrator using a "show" command. The format, according to "Management Access > Managing Certificates > Obtaining Server Certificate" in [ADMIN] is PEM format.

4.2.1.3 Tests

For FFC Schemes using "safe-prime" groups:

469 Testing for FFC Schemes using safe-prime groups is done as part of testing in FCS_CKM.2.

For all other selections:

470 The evaluator shall perform the corresponding tests for FCS_CKM.1 specified in the NDcPP SD, based on the selections chosen for this SFR. If IKE key generation is implemented by a different algorithm than the NDcPP key generation function, the evaluator shall ensure this testing is performed using the correct implementation**.**

Findings:	The vendor uses CAVP certificates A2689 and A2690 for RSA and ECDSA key generation. These are described in [ST] Table 18

4.3 Security management (FMT)

4.3.1 FMT_SMF.1/VPN Specification of Management Functions (VPN) (MOD VPNGW)

4.3.1.1 TSS

471 The evaluator shall examine the TSS to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the TSS identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	Section 6.5.5 of the [ST] claims the TOE provides the admin with both local and remote interfaces to manage all security functions identified in the ST. All functions can be performed on the Mobility Controller using either CLI or WebUI.

4.3.1.2 Guidance

472 The evaluator shall examine the operational guidance to confirm that all management functions specified in FMT_SMF.1/VPN are provided by the TOE. As with FMT_SMF.1 in the Base-PP, the evaluator shall ensure that the operational guidance identifies what logical interfaces are used to perform these functions and that this includes a description of the local administrative interface.

Findings:	In the [ST], FMT_SMF.1/VPN claims the following functions: • Definition of packet rules; • Association of packet filtering rules to network interfaces; • Ordering of filtering rules by priority; Section 2.4 of the [SUPP] describes the process to define packet rules, how to associate the rules with interfaces and how to adjust the priority ordering of rules. The [SUPP] and [CLI] refers to CLI-based commands to configure the firewall rules as required. [ADMIN] provides WebUI affordances described in "Rules and Policies > Firewall Policies > Creating a Firewall Policy" to manipulate firewall rules as well. That section allows to define packet rules and informs the administrator as to how to reorder rules for priority. Assignment of rules to interfaces is performed in the CLI only.

4.3.1.3 Tests

473 The evaluator tests management functions as part of testing the SFRs identified in sections 2.2, 3, and 4. No separate testing for FMT_SMF.1/VPN is required unless one of the management functions in FMT_SMF.1.1/VPN has not already been exercised under any other SFR.

4.4 Packet Filtering (PFP)

4.4.1 FPF_RUL_EXT.1 Packet Filtering Rules (MOD VPNGW)

4.4.1.1 FPF_RUL_EXT.1.1 - TSS

474 The evaluator shall verify that the TSS provide a description of the TOE's initialization and startup process, which clearly indicates where processing of network packets begins to take place, and provides a discussion that supports the assertion that packets cannot flow during this process. The evaluator shall verify that the TSS also includes a narrative that identifies the components (e.g., active entity such as a process or task) involved in processing the network packets and describes the safeguards that would prevent packets flowing through the TOE without applying the ruleset in the event of a component failure. This could include the failure of a component, such as a process being terminated, or a failure within a component, such as memory buffers full and cannot process packets.

Findings:	Section 6.6.1 of the [ST] provides information that the TOE is initialized such that the network interfaces are permitted to process packets only after the data plane initialization is completed: "*All packet level processing and enforcement is performed within the data plane, which is the first component that is initialized. Network*

interfaces are not brought 'up' until the data plane initialization is complete. Therefore, packets cannot flow during this process. In case of system error, packets are dropped by default."

4.4.1.2 FPF_RUL_EXT.1.1 - Guidance

475 The operational guidance associated with this requirement is assessed in the subsequent test EAs.

4.4.1.3 FPF_RUL_EXT.1.1 - Tests

476 The evaluator shall perform the following tests:

- Test 1: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would otherwise be denied by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization.

High-Level Test Description
Configure rules to deny ICMP traffic from flowing through the TOE. Initiate continuous ICMP pings while rebooting the TOE. Verify no ICMP pings flow through the TOE while the TOE is being initialized.
Findings: PASS – The evaluator confirmed that no packets were permitted through the TOE during initialization.

- Test 2: The evaluator shall attempt to get network traffic to flow through the TOE while the TOE is being initialized. A steady flow of network packets that would be permitted by the ruleset should be sourced and directed to a host. The evaluator shall use a packet sniffer to verify none of the generated network traffic is permitted through the TOE during initialization and is only permitted once initialization is complete.

High-Level Test Description
Configure rules to permit ICMP traffic from flowing through the TOE. Initiate continuous ICMP pings while rebooting the TOE. Verify no ICMP pings flow through the TOE while the TOE is being initialized.
Findings: PASS – The evaluator confirmed that no packets were permitted through the TOE during initialization regardless of the fact that there is a rule to permit this traffic.

477 Note: The remaining testing associated with application of the ruleset is addressed in the subsequent test EAs.

4.4.1.4 FPF_RUL_EXT.1.2

478 There are no EAs specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

4.4.1.5 FPF_RUL_EXT.1.3

479 There are no EAs specified for this element. Definition of packet filtering policy, association of operations with packet filtering rules, and association of these rules to network interfaces is described collectively under FPF_RUL_EXT.1.4.

4.4.1.6 FPF_RUL_EXT.1.4 – TSS

NIAP TD 0683

480 The evaluator shall verify that the TSS describes a packet filtering policy that can use the following fields for each identified protocol, and that the RFCs identified for each protocol are supported:

- IPv4 (RFC 791)
  - source address
  - destination address
  - protocol
- IPv6 (RFC 8200)
  - source address
  - destination address
  - next header (protocol)
- TCP (RFC 793)
  - source port
  - destination port
- UDP (RFC 768)
  - source port
  - destination port

481 The evaluator shall verify that the TSS describes how conformance with the identified RFCs has been determined by the TOE developer (e.g., third party interoperability testing, protocol compliance testing). The evaluator shall verify that each rule can identify the following actions: permit, discard, and log. The evaluator shall verify that the TSS identifies all interface types subject to the packet filtering policy and explains how rules are associated with distinct network interfaces. Where interfaces can be grouped into a common interface type (e.g., where the same internal logical path is used, perhaps where a common device driver is used), they can be treated collectively as a distinct network interface.

Findings:	In section 6.6.1 of the [ST], the TSS claims: "*The TOE implements the IPsec protocol and supports the following protocols: RFC 791 (Ipv4), RFC 8200 (Ipv6), RFC 793 (TCP), RFC 768 (UDP). The Aruba Quality Assurance (QA) team performs protocol compliance testing using standards based tools and interoperability testing using a range of external vendor equipment.*" The same section in the [ST] claims that the filtering policy can be set to permit, deny or log the traffic for the attributes defined above. The TSS claims that "access lists" – which contain the rules -- can be applied to Ethernet interfaces.

4.4.1.7 FPF_RUL_EXT.1.4 – Guidance

NIAP TD 0683

482 The evaluator shall verify that the operational guidance identifies the following protocols as being supported and the following attributes as being configurable within packet filtering rules for the associated protocols:

- IPv4 (RFC 791)
  - destination address
  - protocol
- IPv6 (RFC 8200)
  - source address
  - destination address
  - next header (protocol)
- TCP (RFC 793)
  - source port
  - destination port
- UDP (RFC 768)
  - source port
  - destination port

483 The evaluator shall verify that the operational guidance indicates that each rule can identify the following actions: permit, discard, and log.

484 The evaluator shall verify that the operational guidance explains how rules are associated with distinct network interfaces.

485 The guidance may describe the other protocols contained within the ST (e.g., IPsec, IKE, potentially HTTPS, SSH, and TLS) that are processed by the TOE. The evaluator shall ensure that it is made clear what protocols were not considered as part of the TOE evaluation.

Findings:	Section 2.6.1 of [SUPP] provides examples for the TOE to handle IPv4, IPv6, TCP and UDP traffic. Each of the attributes for the protocols above are mentioned as being handled by the TOE. The TOE is capable of permitting traffic using the “permit” rule action (as per section 2.6.1 of the [SUPP]), discarding traffic using the “deny” action (as per section 2.6.1 of the [SUPP]), and logging traffic hitting such rules using the “log” extended action as described in section 2.6.1 of the [SUPP]. The rule actions and extended actions are expanded upon within the [CLI] guide in section “ip access-list”. Section 2.6.1 shows how to assign an ACL to an interface using the “access-group” CLI command against an Ethernet interface: “interface gigabitethernet 1/3; ip access-group FFW_RUL_EXT_1_3 session”. The [SUPP] does not provide any indication that certain protocols are not handled by the firewall. Testing efforts showed that all protocols were subjected to firewall rules.

4.4.1.8 FPF_RUL_EXT.1.4 - Tests

486 The evaluator shall perform the following tests:

- Test 1: The evaluator shall use the instructions in the operational guidance to test that packet filter rules can be created that permit, discard, and log packets for each of the following attributes:

- IPv4
  - destination address
  - protocol
- IPv6
  - source address
  - destination address
  - next header (protocol)
- TCP
  - source port
  - destination port
- UDP
  - source port
  - destination port

NOTE:	Conducted as part of FPF_RUL_EXT.1.6 tests 1-10 as directed in the note below.

- Test 2: The evaluator shall repeat Test 1 above for each distinct network interface type supported by the TOE to ensure that packet filtering rules can be defined for all supported types.

NOTE:	The TOE only has a single network interface type which is tested as part of FPF_RUL_EXT.1.6 below.

487 Note that these test activities should be performed in conjunction with those of FPF_RUL_EXT.1.6 where the effectiveness of the rules is tested; here the evaluator is just ensuring the guidance is sufficient and the TOE supports the administrator creating a ruleset based on the above attributes. The test activities for FPF_RUL_EXT.1.6 define the combinations of protocols and attributes required to be tested. If those combinations are configured manually, that will fulfill the objective of these test activities, but if those combinations are configured otherwise (e.g., using automation), these test activities may be necessary in order to ensure the guidance is correct and the full range of configurations can be achieved by a TOE administrator.

NOTE:	Conducted as part of FPF_RUL_EXT.1.6 as directed in the note above.

4.4.1.9 FPF_RUL_EXT.1.5 - TSS

488 The evaluator shall verify that the TSS describes the algorithm applied to incoming packets, including the processing of default rules, determination of whether a packet is part of an established session, and application of administrator defined and ordered ruleset.

Findings:	Section 6.6.1 of the [ST] provides the high-level algorithm applied for incoming packets: • Check for IP fragments and assemble. • Parse and identify protocol in the IP packet. • Consult the state table to determine whether packets are part of an established session.

- Perform length checks and apply default rules (the default rules are not covered in the scope of evaluation).
- Enforce interface access-lists (ACLs) if configured.
- Derive role for the user and apply role based ACLs. If no role ACLs, then apply default ACLs (deny).
- Perform bandwidth contract enforcement.
- Perform NATing if required.

In addition, section 6.6.1 also states: *"Rules are enforced based on the order defined by the administrator in a first match basis."*

4.4.1.10 FPF_RUL_EXT.1.5 - Guidance

489 The evaluator shall verify that the operational guidance describes how the order of packet filtering rules is determined and provides the necessary instructions so that an administrator can configure the order of rule processing.

Findings: Section 2.6.1 of the [SUPP] states that "*Rules should be configured in order from highest priority to lowest; enforcement is based on a first-match principle where the first rule that matches a traffic flow is applied, and further rules are not processed. The <position> field may be used to insert new rules somewhere other than at the end of a policy.*"

4.4.1.11 FPF_RUL_EXT.1.5 - Tests

490 The evaluator shall perform the following tests:

- Test 1: The evaluator shall devise two equal packet filtering rules with alternate operations – permit and discard. The rules should then be deployed in two distinct orders and in each case the evaluator shall ensure that the first rule is enforced in both cases by generating applicable packets and using packet capture and logs for confirmation.

High-Level Test Description
Attempt to configure two rules as defined in the AA, one that permits and one that denies traffic. Show that the TOE does not permit two rules with the same parameters to exist at the same time and, instead, accepts the last rule entered.
Findings: PASS – The evaluator confirmed that when two rules are entered into the TOE such that the parameters are exactly the same, differing only by whether it is permitted or denied, the TOE will not permit those two rules to coexist and will, instead, use the last rule entered. This was found to meet the description found in the [SUPP] section 2.6.1: "*Note that if an access rule is applied, a duplicate cannot be entered. If the administrator applied a permit rule and then enters a deny rule with the same parameters, the deny rule will replace the permit rule and vice versa*."

- Test 2: The evaluator shall repeat the procedure above, except that the two rules should be devised where one is a subset of the other (e.g. a specific address vs. a network segment). Again, the evaluator should test both orders to ensure that the first is enforced regardless of the specificity of the rule.

High-Level Test Description
Configure two rules, one that allows traffic to pass from the WAN to the LAN, and a second (more specific rule) that denies traffic from the WAN VM to the LAN VM. With the allow rule ordered before the deny rule, attempt a connection from the WAN to the LAN. Verify the connection succeeds. With the deny rule ordered before the allow rule, attempt a connection from the WAN to the LAN. Verify the connection fails.
Findings: PASS – The evaluator confirmed that the rules were successfully applied in the administrative-defined order.

4.4.1.12 FPF_RUL_EXT.1.6 - TSS

491 The evaluator shall verify that the TSS describes the process for applying packet filtering rules and also that the behavior (either by default, or as configured by the administrator) is to discard packets when there is no rule match. The evaluator shall verify the TSS describes when the IPv4 and IPv6 protocols supported by the TOE differ from the full list provided in the RFC Values for IPv4 and IPv6 table.

Findings:	The TSS in section 6.6.1 of the [ST] describes the process for applying packet filtering rules. Specifically it says that "*[a]n authorized administrator can define the traffic that needs to be protected by configuring access-lists. The permit, deny and log operations can be associated with rules in the access-lists. Only a single access-list may be applied to a distinct Ethernet interface. Each rule can identify the following actions: permit, deny, and log.*" Rules are enforced based on the order defined by the administrator in a first match basis. The access lists can be applied to all network interfaces. The same section provides a statement that it is up to the administrator to configure a default deny rule: "*Packets that do not match a rule are then by default handled as configured by the administrator to drop/deny.*" The TSS in the [ST] section 6.6.1 indicates that only IPv6 protocols 135 and 140 are blocked by default.

4.4.1.13 FPF_RUL_EXT.1.6 - Guidance

492 The evaluator shall verify that the operational guidance describes the behavior if no rules or special conditions apply to the network traffic. If the behavior is configurable, the evaluator shall verify that the operational guidance provides the appropriate instructions to configure the behavior to discard packets with no matching rules. The evaluator shall verify that the operational guidance describes the range of IPv4 and IPv6 protocols supported by the TOE.

Findings:	Section 2.6.1 of the [SUPP] provides the administrative configuration needed to implement a default deny rule using the "*any any any deny log*" to drop and log packets. This rule must be implemented at the end of an ACL. Section 2.6.1 of the [SUPP] provides the following information about the range of IPv4 and IPv6 protocols supported by the TOE: - The TOE blocks the following protocols by default: IPv6 Protocol 135

Protocol 140 All other protocols are supported and allowed during normal operation of the TOE.

4.4.1.14 FPF_RUL_EXT.1.6 - Tests

493 The evaluator shall perform the following tests:

- Test 1: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit and log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE permits and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

- Test 2: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv4 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
Configure the TOE to deny and log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

- Test 3: The evaluator shall configure the TOE to permit and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall

configure the TOE to discard and log each supported IPv4 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each supported IPv4 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit/log and deny/log each IPv4 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol but not matching any of the source/destination address combinations and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE drops and logs all traffic not matching any of the rules.

- Test 4: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit and log each supported IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE permits and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

- Test 5: The evaluator shall configure the TOE to permit all traffic except to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and within the configured source and destination addresses in order to ensure that the supported protocols are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must also be denied but are not required to be logged.

High-Level Test Description
Configure the TOE to deny and log each supported IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol through the TOE and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

- Test 6: The evaluator shall configure the TOE to permit and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. Additionally, the evaluator shall configure the TOE to discard and log each supported IPv6 Transport Layer Protocol (see RFC Values for IPv4 and IPv6 table for full possible list) in conjunction with different (than those permitted above) combinations of a specific source address and specific destination address, specific source address and wildcard destination address, wildcard source address and specific destination address, and wildcard source address and wildcard destination address. The evaluator shall generate packets matching each defined IPv6 Transport Layer Protocol and outside the scope of all source and destination addresses configured above in order to ensure that the supported protocols are dropped (i.e., by capturing no applicable packets passing through the TOE) and logged. Any protocols not supported by the TOE must be denied.

High-Level Test Description
Configure the TOE to permit/log and deny/log each IPv6 Transport Layer Protocol with the source and destination address combinations specified in the Test. Send packets matching each transport layer protocol but not matching any of the source/destination address combinations and verify the TOE denies and logs each protocol.
Findings: PASS – The evaluator confirmed that the TOE drops and logs all traffic not matching any of the rules.

- Test 7: The evaluator shall configure the TOE to permit and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination TCP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged.

High-Level Test Description
Configure rules permitting TCP traffic using a selected source and destination port combination, a selected source port, and a selected destination port. Attempt connections from the WAN to the LAN using the ports in the rules and verify the connections succeed.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly.

- Test 8: The evaluator shall configure the TOE to discard and log protocol 6 (TCP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets

matching the configured source and destination TCP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged.

High-Level Test Description
Configure rules denying TCP traffic using a selected source and destination port combination, a selected source port, and a selected destination port. Attempt connections from the WAN to the LAN using the ports in the rules and verify the connections fail.
Findings: PASS – The evaluator confirmed that the TOE enforces all rules properly

- Test 9: The evaluator shall configure the TOE to permit and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are permitted (i.e., by capturing the packets after passing through the TOE) and logged. Here the evaluator ensures that the UDP port 500 (IKE) is included in the set of tests.

High-Level Test Description
Configure rules permitting UDP traffic using a selected source and destination port combination, a selected source port, and a selected destination port. We ensure that port 500 is included in the set of ports that are tested. Attempt connections from the WAN to the LAN using the ports in the rules and verify the connections succeed.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

- Test 10: The evaluator shall configure the TOE to discard and log protocol 17 (UDP) using a selected source port, a selected destination port, and a selected source and destination port combination. The evaluator shall generate packets matching the configured source and destination UDP ports in order to ensure that they are denied (i.e., by capturing no applicable packets passing through the TOE) and logged. Again, the evaluator ensures that UDP port 500 is included in the set of tests.

High-Level Test Description
Configure rules denying UDP traffic using a selected source and destination port combination, a selected source port, and a selected destination port. We ensure that port 500 is included in the set of ports that are tested. Attempt connections from the WAN to the LAN using the ports in the rules and verify the connections fail.
Findings: PASS – The evaluator confirmed that the TOE enforces and logs all rules properly.

494 The following table identifies the RFC defined values for the protocol fields for IPv4 and IPv6 to be used in configuring and otherwise testing packet filtering rule definition and enforcement:

NOTE:	Please refer to the table in the [VPNMOD] FPF_RUL_EXT.1.6.

4.5 Protection of the TSF (FPT)

4.5.1 FPT_FLS.1/SelfTest Failure with Preservation of Secure State (Self-Test Failures) (MOD VPNGW)

4.5.1.1 TSS

495 The evaluator shall ensure the TSS describes how the TOE ensures a shutdown upon a self-test failure, a failed integrity check of the TSF executable image, or a failed health test of the noise source. If there are instances when a shut-down does not occur, (e.g., a failure is deemed non- security relevant), the evaluator shall ensure that those cases are identified and a rationale is provided that supports the classification and justifies why the TOE's ability to enforce its security policies is not affected in any such instance.

Findings:	Section 6.7.1 of the [ST] states that in order to prevent entering an insecure state, the TOE will shut down when the following failures occur: failure of power on self-tests, failure of integrity check of the TSF executable image, and failure of noise source health tests. The specifics of the self-testing mechanisms are described in section 6.7.5 of the [ST] and states: "If a self-test fails, the TOE will immediately halt operation and enter an error state thereby preventing potentially insecure operations (i.e., maintaining a secure state)."

4.5.1.2 Guidance

496 The evaluator shall verify that the operational guidance provides information on the self-test failures that can cause the TOE to shut down and how to diagnose the specific failure that has occurred, including possible remediation steps if available.

Findings:	Section 2.7.3 of [SUPP] points to section 2.7.6 of [SUPP] which states that the TOE will immediately halt operation and enter an error state if any self-test fails. Furthermore the section continues: "*The error output of a failed self-test will appear as follows: "FIPS Aruba Cryptographic asymmetric key KAT failure, main: FIPS_powerupSelfTest failed." If a firmware image fails its integrity check, the TOE will load the previous image (if one is present). An error will be output during boot in this instance stating that the firmware validation failed.* *If the issue continues, the administrator should contact support at http://support.arubanetworks.com.*" Section 2.7.6 of [SUPP] also informs the user that "The local console for Mobility Controller and RAP devices can be reviewed to determine which component has failed a self-test."

4.5.1.3 Tests

497 There are no test Evaluation Activities for this SFR.

4.5.2 FPT_TST_EXT.3 Self-Test with Defined Methods (MOD VPNGW)

4.5.2.1 TSS

498 The evaluator shall verify that the TSS describes the method used to perform self-testing on the TSF executable code, and that this method is consistent with what is described in the SFR.

Findings:	Section 6.7.6 of the [ST] claims a Firmware Integrity Test is performed at startup using RSA 2048-bit Signature Validation. The evaluator confirmed this method is consistent with the SFR in section 5.3.7 of the [ST].

4.5.2.2 Guidance

499 There are no operational guidance Evaluation Activities for this SFR.

4.5.2.3 Tests

500 There are no test Evaluation Activities for this SFR.

4.6 Trusted path/channels (FTP)

4.6.1 FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications) (MOD VPNGW)

4.6.1.1 TSS

501 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Findings:	Refer to findings for FTP_ITC.1.

4.6.1.2 Guidance

502 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications.

Findings:	Refer to findings for FTP_ITC.1.

4.6.1.3 Tests

503 The evaluation activities specified for FTP_ITC.1 in the Supporting Document for the Base-PP shall be applied for IPsec VPN communications. Additional evaluation testing for IPsec is covered in FCS_IPSEC_EXT.1.

High-Level Test Description
Test according to FTP_ITC.1 and FCS_IPSEC_EXT.1/VPN.

High-Level Test Description
Findings: PASS – Test cases were conducted successfully as per FTP_ITC.1 and FCS_IPSEC_EXT.1/VPN (an iteration of FCS_IPSEC_EXT.1).

5 Evaluation Activities for Security Assurance Requirements

5.1 ASE: Security Target

5.1.1 General ASE

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

Findings:	See above sections.

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

ASE_TSS.1 element	**Evaluator Action**
ASE_TSS.1.1C	The evaluator shall examine the TSS to determine that it is clear which TOE components contribute to each SFR or how the components combine to meet each SFR. The evaluator shall verify the sufficiency to fulfil the related SFRs. This includes checking that the TOE as a whole fully covers all SFRs and that all functionality that is required to be audited is in fact audited regardless of the component that carries it out.

Findings:	Section 7.4 of the [ST] shows the distribution of SFRs between all components. Within this table, the SFR is listed as "all" or as "one". The table includes a column representing the Mobility Controller and a Remote Access Point. The components needing to be implemented by "all" components have a tick mark in both component columns; the components needing to be satisfied by "one" component have a tick mark in at least one of the component columns.

5.2 ADV: Development

5.2.1 Basic Functional Specification (ADV_FSP.1)

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

507 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.

508 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.

509 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.

5.2.1.1 Evaluation Activity:

510 *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.*

511 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.

512 The set of TSFI that are provided as evaluation evidence are contained in the Administrative Guidance and User Guidance.

Findings:	From section 7.2.1 of the NDcPP : "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." The [ST] and the AGD comprise the functional specification. If the test in [NDSD] cannot be completed because the [ST] or the AGD are incomplete, then the functional specification is not complete and observations are required. During the evaluator's use of the product and its interfaces (the WebUI, SSH CLI, local serial port), there were no areas that were deficient.

5.2.1.2 Evaluation Activity

513 *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.*

Findings:	See comments in the previous work unit.

5.2.1.3 Evaluation Activity:

514 *The evaluator shall examine the interface documentation to develop a mapping of the interfaces to SFRs.*

515 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.

516 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.

517 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'.

Findings:	See comments in the previous work unit.

5.3 AGD: Guidance Documents

518 It is not necessary for a TOE to provide separate documentation to meet the individual requirements of AGD_OPE and AGD_PRE. Although the EAs in this section are described under the traditionally separate AGD families, the mapping between the documentation provided by the developer and AGD_OPE and AGD_PRE requirements may be many-to-many, as long as all requirements are met in documentation that is delivered to Security Administrators and users (as appropriate) as part of the TOE.

519 Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in section A.9.1.1. (in the NDcPP-SD)

5.3.1 Operational User Guidance (AGD_OPE.1)

520 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.

521 In addition, the evaluator performs the EAs specified below.

5.3.1.1 Evaluation Activity:

522 *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.*

Findings:	The documentation is available for public download from the NIAP PCL page for the TOE.

5.3.1.2 Evaluation Activity

523 *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.*

Findings:	The TOE only claims a single operational environment. While the OE may have several optional components, these components have been described in [SUPP] and [ADMIN]. Specifically, [SUPP] defines the evaluated configuration.

5.3.1.3 Evaluation Activity

524 *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.*

Findings:	The [SUPP] states in section 2.2.1: Ensure the controller has FIPS mode enabled so that cryptographic requirements are met. *(config)# fips enable*

5.3.1.4 Evaluation Activity

525 *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.*

Findings:	After review of the [SUPP], it appears clear to the evaluator that the set of evaluated security functions and interfaces is clear. This exercise involved the evaluator reviewing the set of all functions and interfaces available in the product by extensive review of [ADMIN] and [CLI] and then ensuring that functionality which was mutually exclusive to the security posture was disabled by default or actively disabled as part of the evaluated configuration. In some cases, the evaluator raised observations to ensure the guidance documentation was clear about the scope.

5.3.1.5 Evaluation Activity

526 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.

NIAP TD0536

b) The documentation must describe the process for verifying updates to the TOE for each method selected for FPT_TUD_EXT.1.3 in the Security Target. The evaluator shall verify that this process includes the following steps:

5) 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).

6) 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.

Findings:	See work unit [NDcPP] 5.3.1.3 for configuration of the cryptographic engine.

	The process for verifying updates to the TOE for each method selected in FPT_TUD_EXT.1.3 can be found in the "Guidance Documentation" EA in section 3.6.6.2 of this AAR. Firmware updates can be obtained from the Aruba Support Portal (http://support.arubanetworks.com) as specified in [SUPP] section 2.7.8 and can be uploaded directly to the Mobility Controller (MC) to upgrade the MC and the Remote Access Points (RAPs). Instructions to load the firmware and initiate an update to both the MC and the RAP components are described in section 2.7.8 of the [SUPP]. The TOE will automatically validate the digital signature on the firmware updates. See work unit [NDcPP] 5.3.1.4 for details as to what was covered by the EAs.

5.3.2 Preparative Procedures (AGD_PRE.1)

527 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.

528 Preparative procedures are distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.

529 In addition, the evaluator performs the EAs specified below.

5.3.2.1 Evaluation Activity:

530 *The evaluator shall examine the Preparative procedures to ensure they include a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality (including the requirements of the Security Objectives for the Operational Environment specified in the Security Target).*

531 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).

Findings:	[SUPP] Section 1.6 – Preparatory Guidance provides a description of how the Security Administrator verifies that the operational environment can fulfil its role to support the security functionality. The documentation is written with sufficient detail and explanation that can be understood and used by the target audience.

5.3.2.2 Evaluation Activity

532 *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.*

Findings:	The TOE only claims a single operational environment. All models in the Security Target are addressed in the [SUPP] and are covered by the instructions. Specific hardware installation instructions are provided in section 3 of the [SUPP].

5.3.2.3 Evaluation Activity

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

Findings:	See previous work unit.

5.3.2.4 Evaluation Activity

534 *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.*

Findings:	The guidance documentation provides extensive information on managing the security of the TOE as an individual product. Additional best practice guidance provided within those documents help instil a culture of secure manageability within a larger operational environment.

5.3.2.5 Evaluation Activity

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

536 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.

Findings:	[SUPP] section 1.6 “Preparatory Guidance” includes a description of client requirements to connect with protected administrative interfaces. Sections 2.2.4, 2.2.8 and 2.2.9 of [SUPP] provides instructions for configuring the TOE to support HTTPS/TLS and SSH administrative interfaces. The [SUPP] section 2.4.5 identifies the default password recovery account, which contains a default value, must be disabled in the evaluated configuration.

5.4 ALC: Life-cycle Support

5.4.1 Labelling of the TOE (ALC_CMC.1)

537 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.

Findings:	The evaluator verified that the ST, TOE and Guidance are all labelled 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.

5.4.2 TOE CM coverage (ALC_CMS.1)

538 When evaluating the developer’s coverage of the TOE in their CM system, the evaluator performs the work units as presented in the CEM.

Findings:	The evaluator verified that the ST, TOE and Guidance are all labelled 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.

5.5 ATE: Tests

5.5.1 Independent Testing – Conformance (ATE_IND.1)

539 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.

540 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.

541 The evaluator should consult Appendix 709 when determining the appropriate strategy for testing multiple variations or models of the TOE that may be under evaluation.

542 Note that additional Evaluation Activities relating to evaluator testing in the case of a distributed TOE are defined in section A.9.3.1.

Findings:	A high level overview of the independent testing document is provided throughout the AAR. The full details of the Independent Testing effort are documented in the non-public Detailed Test Report. The TOE is a distributed TOE and the additional EAs in section A.9.3.1 are relevant.

5.6 Vulnerability Assessment

5.6.1 Vulnerability Survey (AVA_VAN.1)

543 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

544 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.

545 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.

5.6.1.1 Evaluation Activity (Documentation):

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

547 *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.*

NIAP TD0547

548 The developer shall provide documentation identifying the list of software and hardware components that compose the TOE. Hardware components should identify at a minimum the processors used by the TOE. Software components include applications, the operating system and other major components that are independently identifiable and reusable (outside of the TOE), for example a web server, protocol or cryptographic libraries, (independently identifiable and reusable components are not limited to the list provided in the example). This additional documentation is merely a list of the name and version number of the components and will be used by the evaluators in formulating vulnerability hypotheses during their analysis.

Findings:	The evaluator collected this information from the developer which was used to feed into the Type 1 Flaw Hypotheses search (below).

549 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 3.4.1.2 and 3.5.1.2.

5.6.1.2 Evaluation Activity:

550 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.

Findings:	The following sources of public vulnerabilities were considered in formulating the specific list of flaws to be investigated by the evaluators, as well as to reference in directing the evaluators to perform key-word searches during the evaluation of the TOE. Hypothesis sources for public vulnerabilities were: - NIST National Vulnerabilities Database (can be used to access CVE and US-CERT databases identified below): https://web.nvd.nist.gov/view/vuln/search - Common Vulnerabilities and Exposures: http://cve.mitre.org/cve/ https://www.cvedetails.com/vulnerability-search.php

- US-CERT: http://www.kb.cert.org/vuls/html/search

- Tenable Network Security: https://www.tenable.com/cve

- Tipping Point Zero Day Initiative: http://www.zerodayinitiative.com/advisories

- Offensive Security Exploit Database: https://www.exploit-db.com/

- Rapid7 Vulnerability Database: https://www.rapid7.com/db/vulnerabilities

Type 1 Hypothesis searches were conducted on October 31, 2023 and included the following search terms:

- Aruba Mobility Controller

- Aruba Remote Access Point

- ArubaOS 8.10

- Aruba Crypto Module

- Aruba OpenSSL Module

- Aruba Bootloader Module

- Aruba 303H

- Aruba 503H

- Aruba 505H

- Aruba 7210

- Aruba 7220

- Aruba 9004

- Broadcom XLP416

- Broadcom XLP432

- Intel Atom C3508

- Qualcomm IPQ4019

- Broadcom BCM47622L

- FreeRADIUS

- Ntp.org

- Mocana

- OpenSSH

- OpenSSL

The evaluation team reviewed the potential vulnerabilities and determined that none of the potential vulnerabilities are exploitable in the evaluated configuration. The

evaluation team determined, based on these searches, that no other residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

RSA key transport attacks are the only type-2 hypotheses identified for the NDcPP. The TOE does not support RSA key transport.

The evaluation team developed Type 3 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

The evaluation team developed Type 4 flaw hypotheses in accordance with Sections A.1.3, A.1.4, and A.2, and no residual vulnerabilities exist that are exploitable by attackers with Basic Attack Potential.

5.7 Evaluating additional components for a distributed TOE

551 In the case of a distributed TOE the Security Target will identify an evaluated configuration that consists of a number of separate components chosen by the ST author, which collectively satisfy the requirements of the cPP. This evaluated configuration need not be the minimum set of components that could possibly meet the cPP (e.g. if the TOE is intended for large enterprise deployments then the evaluated configuration might include some redundancy in components in order to support expected connectivity and loads), but because this is the main configuration referred to in the ST and the evaluation, it is treated in this section as the minimum configuration of interest and is referred to here as the 'minimum configuration' as well as the 'evaluated configuration'.

552 In addition to the minimum configuration above, the ST may also identify (at the author's discretion, and subject to verification as described in this section) which TOE components can have instances added to an operational configuration without affecting the validity of the CC certification. The ST description may include constraints on how such components are added, including required and/or prohibited configurations of the components.

553 Extra instances of a TOE component must have the same hardware and software as the original component included in the evaluated configuration.

554 It is noted that undesirable configurations may be possible in the operational deployment of a TOE – such as allowing a TOE component to be managed from separate and potentially conflicting administration domains. However, the definition of 'undesirable' and of the risks involved in such cases will be specific to each operational environment and is therefore not treated as part of the evaluation. Correct and appropriate configuration of this sort remains a matter for expert network planning and design in the operational environment.

5.7.1 Evaluator Activities for Assessing the ST

5.7.1.1 TSS

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

Findings:	The tested configuration included one Aruba Mobility Controller and two Aruba Remote Access Points. The [ST], in section 2.4, states that this does not restrict the number of RAPs that may be managed in a conformant deployment.

5.7.2 Evaluator Activities for Assessing the Guidance Documentation

5.7.2.1 Guidance Documentation

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

Findings:	Section 1.1 of the [SUPP] states the following: "*The evaluated configuration was tested with more than one RAP device. The number of RAP devices in a deployment has no impact on the overall enforcement of the SFR's since each RAP is configured in the same way as described in this document*."

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

Findings:	The introduction of multiple Remote Access Point (RAP) devices still all use the stated IPsec channels for communicating within the distributed TOE solution as per section 1.1 of the [SUPP].

5.7.3 Evaluator Activities for Testing the TOE

5.7.3.1 Tests

558 The evaluator tests the TOE in the minimum configuration as defined in the ST (and the guidance documentation).

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

High-Level Test Description
No differences were identified with regards to the use of extra RAP components.
Findings: N/A.

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

561 Communications: the evaluator follows the guidance documentation to confirm, by testing, that any additional connections introduced with the extra component and not

present in the minimum configuration are consistent with the requirements stated in the ST (e.g. with regard to protocols and ciphersuites used). An example of such an additional connection would be if a single instance of the component is present in the minimum configuration and adding a duplicate component then introduces an extra communication between the two instances. Another example might be if the use of the additional components necessitated the use of a connection to an external authentication server instead of using locally stored credentials.

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

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

High-Level Test Description
Test the use of multiple RAP devices and show that they independently and collectively uphold all of the applicable SFRs.
Findings: PASS – The evaluator tested with two RAP devices and found that each RAP was capable of communicating with the controller using an IPsec tunnel, offloading their own logs to the controller and being independently managed by the Controller.