Aruba Mobility Conductor with ArubaOS 8.10

Assurance Activity Report

Version 0.11

June 2023

Document prepared by

www.lightshipsec.com

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
TOE	Aruba Mobility Conductor with ArubaOS 8.10
Security Target	Aruba Mobility Conductor with ArubaOS 8.10 Security Target, v1.2
Protection Profile	collaborative Protection Profile for Network Devices, Version 2.2e, 23-March-2020 (NDcPP)

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 (NDcPP-SD)		
Interpretations	**TD #**	**Name**	**Exclusion Rationale**
	TD0527	Updates to Certificate Revocation Testing (FIA_X509_EXT.1)	
	TD0528	NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	
	TD0536	NIT Technical Decision for Update Verification Inconsistency	

TD0537	NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	
TD0546	NIT Technical Decision for DTLS - clarification of Application Note 63	FCS_DTLSC_EXT.1 not claimed.
TD0547	NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	
TD0555	NIT Technical Decision for RFC Reference incorrect in TLSS Test	
TD0556	NIT Technical Decision for RFC 5077 question	
TD0563	NiT Technical Decision for Clarification of audit date information	
TD0564	NiT Technical Decision for Vulnerability Analysis Search Criteria	
TD0569	NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7	
TD0570	NiT Technical Decision for Clarification about FIA_AFL.1	
TD0571	NiT Technical Decision for Guidance on how to handle FIA_AFL.1	
TD0572	NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	
TD0580	NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	
TD0581	NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	
TD0591	NIT Technical Decision for Virtual TOEs and hypervisors	
TD0592	NIT Technical Decision for Local Storage of Audit Records	

	TD0631	NIT Technical Decision for Clarification of public key authentication for SSH Server	
	TD0632	NIT Technical Decision for Consistency with Time Data for vNDs	
	TD0633	NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	
	TD0634	NIT Technical Decision for Clarification required for testing IPv6	N/A – FCS_TLSC_EXT.1 and FCS_DTLSC_EXT.1 not claimed.
	TD0635	NIT Technical Decision for TLS Server and Key Agreement Parameters	
	TD0636	NIT Technical Decision for Clarification of Public Key User Authentication for SSH	N/A - FCS_SSHC_EXT.1 not claimed.
	TD0638	NIT Technical Decision for Key Pair Generation for Authentication	
	TD0639	NIT Technical Decision for Clarification for NTP MAC Keys	
	TD0670	NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	N/A - FCS_TLSC_EXT.2 not claimed.
	TD0738	NIT Technical Decision for Link to Allowed-With List	

1.3 Reference Documents

Table 3: List of Reference Documents

Ref	Document
[ST]	Aruba Mobility Conductor with ArubaOS 8.10 Security Target, v1.2, June 2023
[AGD]	Aruba OS 8.10 Supplemental Guidance (Common Criteria Configuration Guidance for Aruba Mobility Conductor with ArubaOS 8.10-FIPS) Version 2.6, June 2023
[GUIDE-1]	ArubaOS 8.10.0.0 User Guide, Revision 02
[GUIDE-2]	ArubaOS 8.10.0.0 Getting Started Guide, Revision 01

Ref	Document
[GUIDE-3]	ArubaOS 8.x Command-Line Interface Reference Guide
[GUIDE-4]	ArubaOS 8.10.0.0 Syslog Reference Guide, Revision 01

2 TOE Details

2.1 TOE Models

The physical boundary of the TOE includes the Aruba Mobility Conductor hardware models shown in the table below.

Table 4: TOE Models

Model	CPU	Software	Notes on Differences
MCR-HW-1K-F1	Intel Xeon E5-2609v4 (Broadwell)	ArubaOS 8.10	Difference in the number of managed nodes/ supported devices, clients, and controllers due to the licenses applied.
MCR-HW-5K-F1	Intel Xeon E5-2620v4 (Broadwell)		
MCR-HW-10K-F1	Intel Xeon E5-2650v4 (Broadwell)		

2.1.1 Test Platform Equivalency

3 The team used the [NDcPP] as the basis for the following 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.

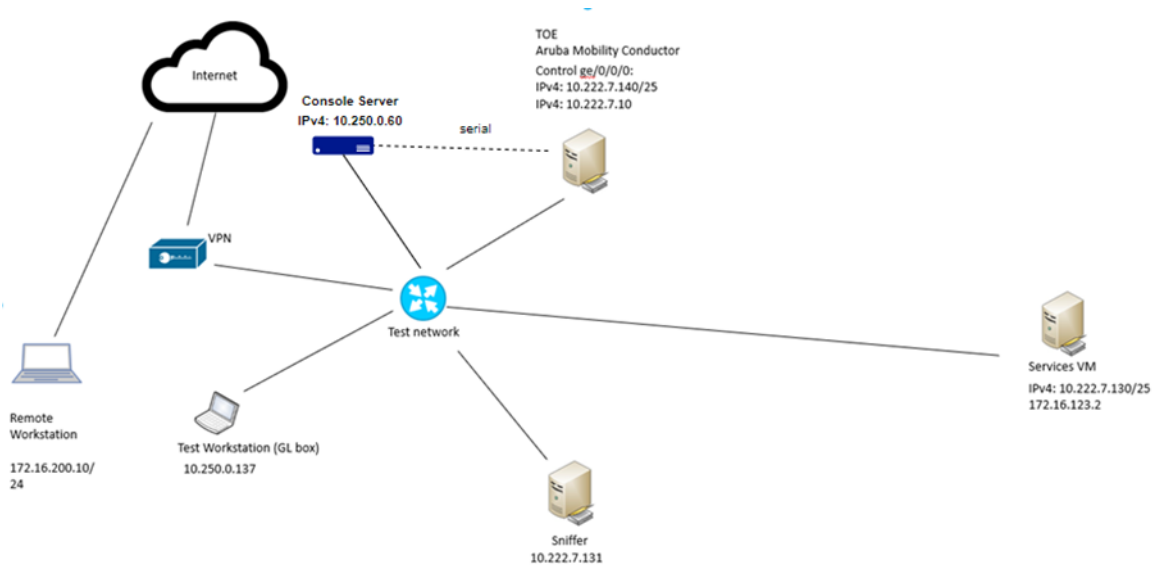

Figure 1: Test Setup

3 Evaluation Activities for SFRs

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:	Within [ST] section 6.1.1 the TSS states that the TOE generates the audit records specified at Table 11 of the [ST]. The following information is logged as a result of the Security Administrator generating/importing or deleting cryptographic keys: a) Generate CSR. Action and key reference. b) Import Certificate. Action and key reference. c) Import CA Certificate. Action and key reference.

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:	Not a distributed TOE.

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:	A list of all auditable events are listed in the chart available in the [AGD] under section 2.1.1.

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:

In [AGD] section 2.1.1 the guidance states that:

All Administrative actions are audited by the TOE. As noted within the Syslog Guide for 8.X (https://support.hpe.com/hpsc/doc/public/display?docId=emr_na-c05321932), the Conductor creates syslog entries for all commands and configuration changes that alter system behavior, the user name of the user making the change, and the location of the user. This information appears in the output of the syslog, with the keyword COMMAND. This same information also appears in the output of the CLI command **show audit-trail**.

The syslog information in the example below shows that a user with the username **admin** logged in to the controller through the serial port, changed logging levels, loaded new software onto partition 1, then updated the system clock.

(host) #show audit-trail

Jul 4 21:53:54 2022 cli[1439]: USER:admin@serial COMMAND: -- command executed successfully

Jul 4 22:20:22 2022 cli[1439]: USER:admin@serial COMMAND: -- command executed successfully

Jul 4 22:31:00 2022 fpcli: USER:admin@10.240.104.135 COMMAND: -- command executed successfully

By default, the Conductor does not generate a log entry for **show** commands issued using the CLI, as these commands display existing settings but do not change system behavior. To create a log entry for all commands issued, (including show commands) access the CLI in config mode and issue the command **audit-trail all**.

A full record of audit records generated by the controller can be found at the following link:

https://support.arubanetworks.com/Documentation/tabid/77/DMXModule/512/Command/Core_Download/Default.aspx?EntryId=32318

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.

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
Ensure that the TOE displays an audit record for each of the auditable events defined for this requirement.
Findings: PASS – The evaluator performed the testing in conjunction with the testing of the security mechanisms directly. The evaluator confirmed that the TOE correctly generates audit records for the events listed in the table of audit events and administrative actions. The TOE is not a 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
The TOE is not a distributed TOE.
Findings: N/A

3.1.3 FAU_STG_EXT.1 Protected audit event storage

3.1.3.1 TSS

15 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:	Within [ST] section 6.1.3 the TSS states that the Security Administrator can configure the TOE to send logs to a Syslog server. Log events are sent in real-time. Logs are sent via IPsec.

16 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:	Within [ST] section 6.1.3 the TSS states that the TOE stores audit records locally and provides CLI and Web UI capabilities for the administrator to view the contents of the audit trail. For local storage, the maximum log file size for all processes is ARUBA_MAX_LOG_FILE_SIZE (i.e. 95,304 bytes). However, the security.log, system.log and user-debug logs have a maximum file size given by A_MAX_SECURITY_USER_DEBUG_LOG_FILE_SIZE (i.e. 1000KB). When the local audit data store is full, the TOE will overwrite audit records starting with the oldest audit record. Only authorized administrators may view audit records and no capability to modify the audit records is provided.

17 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:	The TOE is not distributed. Within [ST] section 6.1.3 the TSS states that the TOE stores audit records locally.

18 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:	Within [ST] section 6.1.3 the TSS states that when the local audit data store is full, the TOE will overwrite audit records starting with the oldest audit record.

19 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:	Within [ST] section 6.1.3 the TSS states that the transmission of audit data is done in real-time.

20 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:	TOE is not distributed.

21 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:	TOE is not distributed.

3.1.3.2 Guidance Documentation

22 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: [AGD] Section 2.1.3 of the guidance states:

Local storage space for audit logs is limited on a Mobility Conductor. The local protected log storage operates using the first in, first out (FIFO) method, therefore audit logs are overwritten when the available space is exhausted. To operate in the evaluated configuration, an external syslog server must be used. All audit logs are simultaneously written to both the local audit log and the syslog server. The local audit logs and logs sent to a remote server are identical.

To configure an external syslog server:

(config)# logging <ip address>

The connection between the Mobility Conductor and the syslog server must be protected using IPsec. Configure a site-to-site VPN tunnel to carry this traffic. The syslog server must use a different IP address for the syslog receiver process than it uses for IPsec termination. Alternatively, a VPN gateway (such as an Aruba Mobility Conductor) may front-end the syslog server to provide the IPsec tunnel. The following is an example of an IPsec tunnel which assumes that the syslog receiver process listens on 192.168.1.1, and the IPsec tunnel terminates on 192.168.2.1 – these IP addresses may be on the same server, or on different systems.

crypto-local ipsec-map <name> 10

version v2

set ikev2-policy <policy>

peer-ip <ip address>

src-net <ip address> <subnet>

dst-net <ip address> <subnet>

set transform-set "<transform-set>"

set security-association lifetime seconds <seconds>

set security-association lifetime kilobytes <kilobytes>

pre-connect enable

trusted enable

uplink-failover disable

force-natt disable

set ca-certificate root-ca

set server-certificate server-cert

Adjust the above ipsec-map as appropriate, following instructions in the ArubaOS User Guide. The peer-ip and dst-net addresses cannot be the same. Note that bi-directional communication is not necessary – syslog is sent using UDP, so the only requirement is that packets are able to flow from the Mobility Conductor to the syslog server.

23 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:	[AGD] Section 2.1.3 states, "All audit logs are simultaneously written to both the local audit log and the syslog server. The local audit logs and logs sent to a remote server are identical."

24 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:	[AGD] Section 2.1.3 states "The local protected log storage operates using the first in, first out (FIFO) method, therefore audit logs are overwritten when the available space is exhausted."

3.1.3.3 Tests

25 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

High-Level Test Description
and is evidenced in the DTR Evidence document by any audit record which contains the text "<142>" (an encoding of the syslog facility and severity level received by our syslog server).
Findings: PASS – The evaluator established a session between the TOE and audit server, examined the traffic and observed that the data could not be viewed in the clear as part of FTP_ITC.1, test 3. The audit server version is syslog-ng 3.19.1 as reported in the Test Plan. The evaluator ensured that the TOE is capable of transferring audit data to an external audit server audit server automatically without administrator intervention and this is evidenced throughout the DTR evidence document.

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 multiple 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 been overwritten.
Findings: PASS – The evaluator confirmed that the TOE overwrites the oldest log file when the configured storage space for audit logs is filled.

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: N/A

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
The TOE is not a distributed TOE.
Findings: N/A

3.2 Cryptographic Support (FCS)

3.2.1 FCS_CKM.1 Cryptographic Key Generation

3.2.1.1 TSS

26 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: Within [ST] section 6.2.1 the TSS states that the TOE supports key generation for the following asymmetric schemes:

a) RSA Schemes. RSA 2048-bit used in SSH, TLS, and IPsec.
b) ECC Schemes. Uses NIST curves P-256 and P-384 for SSH, TLS, and IPsec.
c) FFC Safe Prime Groups. Used in IPsec.

3.2.1.2 Guidance Documentation

27 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: In [AGD] section 2.2.1 the guidance states that:

No configuration required. Ensure the Controller has FIPS mode enabled so that cryptographic requirements are met.

(config)# fips enable

“During regular operation of the TOE, key generation is invoked during session establishment between the TOE and external IT entities for user sessions.”

How to configure the TOE to use selected key generation schemes during session establishment is discussed in the associated protocol sections:
[AGD] section 2.2.7 describes how to configure the TOE to use the key generation schemes and key sizes used in IPSec.

[AGD] section 2.2.9 describes how to configure the TOE to use the key generation schemes and key sizes used in SSH.

[AGD] section 2.2.10 describes how to configure the TOE to use the key generation schemes and key sizes used in HTTPS/TLS.
“No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.”

[AGD] Section 2.2.1 states, “An administrator can invoke the use of RSA and ECDSA during generation of certificates used for X.509.”

[AGD] Section 2.3.6 states, “Certificate Signing Requests (CSRs) may be generated by the Conductor. This process is described in the ArubaOS User Guide. Best practice is to generate the CSR on the Conductor, then load the resulting certificate after issuance by the CA. This protects the private key from disclosure. If the private key is generated externally, the Conductor can also accept a certificate/key combination in the form of a PKCS#12 file.”

[GUIDE-1] Table 207 describes how to configure the TOE to generate RSA and ECDSA keys for use by the TOE.

3.2.1.3 Tests

28 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

29 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*.

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

a) Random Primes:

- Provable primes
- Probable primes

b) Primes with Conditions:

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

31 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

32 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

33 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)

34 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.

35 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

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

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

37 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.

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

39 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.

40 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

41 for each FFC parameter set and key pair.

NIAP TD0580

FFC Schemes using "safe-prime"

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

Findings:	The vendor uses the CAVP certificates A2690 and A2689 for RSA and ECDSA. Schemes using safe primes are tested in FCS_CKM.2.1. This is described in [ST] Table 13.

3.2.2 FCS_CKM.2 Cryptographic Key Establishment

3.2.2.1 TSS

43 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:	Within [ST] section 6.2.2 the TSS states that the TOE supports the following key establishment schemes: a) Elliptic curve-based schemes used for SSH, TLS, and IPsec that meet NIST SP800-56A Rev 3 and implement curves secp256r1 and secp384r1. TOE is sender. b) FFC Safe Prime Groups. Used in IPsec and meets NIST SP 800-56A Rev 3 and uses groups listed in RFC 3526. The [ST] table 14 identifies the scheme being used by each service.

NIAP TD0580

44 **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

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

Scheme	SFR	Service
RSA	FCS_TLSS_EXT.1	Administration
ECDH	FCS_SSHC_EXT.1	Audit Server
Removed: ~~DiffieHellman (Group 14)~~	~~FCS_SSHC_EXT.1~~	~~Backup Server~~

ECDH	FCS_IPSEC_EXT.1	Authentication Server

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

Findings:	The [ST] table 14 identifies the scheme being used by each service.

3.2.2.2 Guidance Documentation

47 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:	How to configure the TOE to use selected key establishment schemes during session establishment is discussed in the associated protocol sections: [AGD] section 2.2.7 describes how to configure the TOE to use the key generation schemes and key sizes used in IPSec. [AGD] section 2.2.9 describes how to configure the TOE to use the key generation schemes and key sizes used in SSH. [AGD] section 2.2.10 describes “No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.” for HTTPS/TLS.

3.2.2.3 Tests

Key Establishment Schemes

48 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

49 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

50 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.

51 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.

52 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.

53 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.

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

Validity Test

55 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.

56 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).

57 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 the CAVP certificates A2690 and A2689 for ECC Key Establishment. This is described in [ST] Table 13.

RSA-based key establishment schemes

58 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 "RSA-based key establishment."
Findings: N/A

NIAP TD0580 Removed:

~~Diffie-Hellman Group 14~~

59 ~~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
N/A
Findings: Removed per TD0580

FFC Schemes using "safe-prime" groups

60 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
Verify the TOE can successfully perform key exchanges with a known good FFC scheme using Diffie-Helman group 14.
Findings: PASS – FTP_TRP.1/Admin and FTP_ITC.1 claim SSH, TLS, and IPsec. Only the IPsec protocol uses safe-prime Diffie-Hellman Group 14. Refer to the test case for FCS_IPSEC_EXT.1.11. The test cases uses an independent, known-good interoperable cryptographic implementation.

3.2.3 FCS_CKM.4 Cryptographic Key Destruction

3.2.3.1 TSS

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

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

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

identification and description of the interfaces that the TOE uses to destroy keys (e.g., file system APIs, key store APIs).

Findings:	Within [ST] section 6.2.3 the TSS references Table 16: Keys. This table includes all relevant keys and their storage location (in flash memory or in RAM). Based on the chart, keys in volatile and non-volatile storage are deleted by means of zeroization. Keys are protected as described in Table 16 of the ST. In all cases, plaintext keys cannot be viewed through an interface designed specifically for that purpose.

62 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:	Selection "destruction of reference" not chosen in the [ST]. For non-volatile memory the [ST] Table 16 identifies the command to destroy the keys as: write erase all Other keys are destroyed by rebooting the module.

63 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:	Table 16: "Keys" in section 6.5.1 of the [ST] has a column labelled "Storage" which provides the evidence of how the key is stored. For keys stored encrypted, the table identifies how the key is encrypted.

64 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 additional configurations claimed.

65 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 specify the use of "A value that does not contain any CSP".

3.2.3.2 Guidance Documentation

66 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.

67 For example, when the TOE does not have full access to the physical memory, it is possible that the storage may be implementing wear-levelling and garbage collection. This may result in additional copies of the key that are logically inaccessible but persist physically. Where available, the TOE might then describe use of the TRIM command[2] and garbage collection to destroy these persistent copies upon their deletion (this would be explained in TSS and Operational Guidance).

Findings:	In [AGD] section 2.2.3 the guidance states that: No configuration required. During runtime, all CSPs (Critical Security Parameters) will be zeroized automatically when no longer needed. To erase all CSPs stored in flash memory (as well as software images and configuration files), issue the command 'zeroize-tpm-keys' (for hardware). This command will overwrite the entire flash with an alternating pattern. The Conductor must be restored through TFTP after this process. In addition, files in the flash can be zeroized using the 'write erase all' command.

3.2.3.3 Tests

68 None

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

3.2.4.1 TSS

69 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:	Within [ST] section 6.2.4 the TSS states that the TOE provides symmetric encryption and decryption capabilities using 128 and 256 bit AES in CBC, CTR and GCM mode. AES is implemented in the following protocols: TLS, SSH and IPsec. The relevant NIST CAVP certificate numbers are listed Table 4 of the ST.

3.2.4.2 Guidance Documentation

70 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:	[AGD] Section 2.2.4 states, Ensure that the Advanced Cryptography License is installed for all required cryptographic algorithms to be enabled. Ensure the Conductor has FIPS mode enabled so that cryptographic requirements are met. *(config)# fips enable*

[2] Where TRIM is used then the TSS and/or guidance documentation is also expected to describe how the keys are stored such that they are not inaccessible to TRIM, (e.g. they would need not to be contained in a file less than 982 bytes which would be completely contained in the master file table).

	How to configure the TOE to use selected AES modes: [AGD] section 2.2.7 describes how to configure the TOE to use the selected modes and key sizes used in IPSec. [AGD] section 2.2.9 describes how to configure the TOE to use the selected modes and key sizes used in SSH. [AGD] section 2.2.10 describes “No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.” for HTTPS/TLS.

3.2.4.3 Tests

AES-CBC Known Answer Tests

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

72 **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.

73 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.

74 **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.

75 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.

76 **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].

77 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.

78 **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].

79 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

80 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.

81 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

82 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]
```

83 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.

84 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

85 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.

86 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.

87 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.

88 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

89 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):

90 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.

91 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.

92 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.

93 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].

94 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

95 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

96 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]
```

97 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.

98 There is no need to test the decryption engine.

Findings:	The vendor uses the CAVP certificates A2690 and A2689 for AES. This is described in [ST] Table 13.

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

3.2.5.1 TSS

99 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:	Within [ST] section 6.2.5 the TSS states that the TOE provides cryptographic signature generation and verification services using: a) RSA Signature Algorithm with key size of 2048 bits b) ECDSA Signature Algorithm with NIST curves P-256, P-384. Signature verification services are used in the TLS, SSH and IPsec protocols. Additionally, RSA signature verification is used for trusted updates. The relevant NIST CAVP certificate numbers are listed in Table 4 of the [ST].

100

3.2.5.2 Guidance Documentation

101 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:	[AGD] section 2.2.4 states, “Ensure that the Advanced Cryptography License is installed in order for all required cryptographic algorithms to be enabled. Ensure the controller has FIPS mode enabled so that cryptographic requirements are met.” How to configure the TOE to use selected cryptographic algorithm modes: [AGD] section 2.2.7 describes how to configure the TOE to use the selected cryptographic algorithm and key sizes used in IPSec. [AGD] section 2.2.9 describes how to configure the TOE to use the selected cryptographic algorithm and key sizes used in SSH. [AGD] sections 2.2.10 describes “No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.” for HTTPS/TLS.

102

3.2.5.3 Tests

ECDSA Algorithm Tests

ECDSA FIPS 186-4 Signature Generation Test

103 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

104 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

105 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.

106 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

107 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.

108 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 the CAVP certificates A2690 and A2689 for RSA and ECDSA signature generation and verification. The vendor also uses CAVP certificate A2688 for RSA signature verification of trusted updates. This is described in [ST] Table 13.

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

3.2.6.1 TSS

109 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:	Within [ST] section 6.2.6 the TSS states that the TOE provides cryptographic hashing services using SHA-1, SHA-256, SHA-384 and SHA-512. SHA is implemented in the following parts of the TSF: a) TLS, SSH and IPsec; and b) Digital signature verification as part of trusted update validation The relevant NIST CAVP certificate numbers are listed in Table 4 of the [ST].

3.2.6.2 Guidance Documentation

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

Findings:	[AGD] section 2.2.4 states, “Ensure that the Advanced Cryptography License is installed in order for all required cryptographic algorithms to be enabled. Ensure the controller has FIPS mode enabled so that cryptographic requirements are met.” How to configure the TOE to use the required hash sizes: [AGD] section 2.2.7 describes how to configure the TOE to use the required hash sizes used in IPSec. [AGD] section 2.2.9 describes how to configure the TOE to use the required hash sizes used in SSH. [AGD] section 2.2.10 describes “No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.” for HTTPS/TLS.

3.2.6.3 Tests

111 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.

112 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

113 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

114 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

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

Selected Long Messages Test - Byte-oriented Mode

116 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

117 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 the CAVP certificates A2690 and A2688 for Hashing. This is described in [ST] Table 13.

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

3.2.7.1 TSS

118 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:	Within section 6.2.7 the TSS states that the TOE provides keyed-hashing message authentication services using HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384. HMAC is implemented in the following protocols: TLS, IPsec and SSH. The HMAC key lengths, block sizes and digest sizes are identified as Table 15: HMAC Characteristics in the [ST].

3.2.7.2 Guidance Documentation

119 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:	[AGD] section 2.2.4 states, “Ensure that the Advanced Cryptography License is installed in order for all required cryptographic algorithms to be enabled. Ensure the Conductor has FIPS mode enabled so that cryptographic requirements are met.” How to configure the TOE to use the values used by the HMAC function: [AGD] section 2.2.7 describes how to configure the TOE to use the values used by the HMAC function in IPSec. [AGD] section 2.2.9 describes how to configure the TOE to use the values used by the HMAC function in SSH.

	[AGD] section 2.2.10 describes “No configuration is required to set the permitted cipher suites or the associated key agreement parameters once ‘fips enable’ has been entered on the Conductor.” for HTTPS/TLS.

3.2.7.3 Tests

120 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 the CAVP certificates A2690 and A2689 for HMAC. This is described in [ST] Table 13.

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

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

3.2.8.1 TSS

122 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:	Within [ST] section 6.2.11 the TSS states that the TOE contains a CTR_DRBG that is seeded from one software entropy source. Entropy from the noise source is used to seed the DRBG with 256 bits of full entropy. Additional detail is provided in the proprietary Entropy Description.

3.2.8.2 Guidance Documentation

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

Findings:	In section 2.2.8 the guidance states that “No configuration required.”

3.2.8.3 Tests

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

125 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).

126 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.

127 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 the CAVP certificate A2690 for DRBG. This is described in [ST] Table 13.

3.3 Identification and Authentication (FIA)

3.3.1 FIA_AFL.1 Authentication Failure Management

3.3.1.1 TSS

128 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:	Within [ST] section 6.3.5 the TSS states that the TOE is capable of detecting and tracking authentication failures of local and remote administrators attempting to authenticate with a password via the GUI or CLI interfaces. After an administrator specified number of consecutive failed authentication attempts, the TOE will lockout the offending remote administrator account and log the event. The account will remain locked out until the administrator-defined period of time has elapsed. The administrator can configure the maximum number of failed attempts and the lockout time threshold using the web GUI or CLI.

129 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:	Within [ST] section 6.3.5 the TSS states that to ensure that an administrator cannot be fully locked out of the TOE, a local user (with the same username) that is accessed via the local console and is not configured to authenticate against the remote authentication server, may continue to log in.

3.3.1.2 Guidance Documentation

130 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.

131 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:	In [AGD] section 2.3.1 the guidance states that: All configuration related to administrative login is configured using "aaa password-policy mgmt". Note that if the remote authentication server locks out a user, the local account with the same name will not be marked as locked. However, the local user will not be able to authenticate when configured authenticate against the remote authentication server. To configure failed authentication lockout that will lock an administrative account for five minutes, when five failed login attempts occur in a three-minute period, use the following commands: *(config) #aaa password-policy mgmt* *(Mgmt Password Policy) #password-lock-out 5* *(Mgmt Password Policy) #password-lock-out-time 5* *(Mgmt Password Policy) #enable:* When an account has been locked out for a specified duration, the process of unlocking the account may take up to 60 seconds beyond the configured lockout period that has been configured.

3.3.1.3 Tests

132 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 – The evaluator confirmed that the TOE blocks login attempts to a user after the configured threshold of invalid login attempts is met at both the Web UI and the remote CLI.

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
The TOE does not claim manual unlocking functionality.
Findings: N/A

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 2m00s 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 2m00s 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 – The evaluator confirmed that, using correct credentials, access was denied prior to the lockout duration expiring and access was granted after the lockout duration had expired.

3.3.2 FIA_PMG_EXT.1 Password Management

3.3.2.1 TSS

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

Findings:	Within [ST] section 6.3.1 the TSS states that the TOE supports the local definition of users with corresponding passwords. The passwords can be composed of any combination of upper and lower case letters, numbers, and special characters “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, [“`”, “~”, “-“, “_”, “=”, “+”, “[“, “]”, “{“, “}”, “\”, “|”, “;”, “:”, “’”, “”””, “,”, “.”, “<”, “>”, “/” The minimum password length is settable by the Administrator and can range from 8 to 32 characters. The maximum password length supported by the TOE is 128 characters.

3.3.2.2 Guidance Documentation

134 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:	In [AGD] section 2.3.2 the guidance states that: Administrative password policies are configured under “aaa password-policy mgmt”. *(config) #aaa password-policy mgmt* *(Mgmt Password Policy) #password-min-length 8* *(Mgmt Password Policy) #password-min-lowercase-characters 1* *(Mgmt Password Policy) #password-min-uppercase-characters 1* *(Mgmt Password Policy) #password-min-special-characters 1* *(Mgmt Password Policy) #password-min-digit 1* *(Mgmt Password Policy) #enable* The following special characters can be used when configuring passwords: `, ~, !, @, #, $, %, ^, &, *, (,), -, _, =, +, [,], {, }, \, |, ;, :, ‘, “, comma, <, >, period, / Once configured, the TOE only permits the use of strong passwords.

3.3.2.3 Tests

135 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.

136

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 to include all valid characters and show that it can be used to login to the Web GUI. 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.

a) 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.

3.3.3 FIA_UIA_EXT.1 User Identification and Authentication

3.3.3.1 TSS

137 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:	Within [ST] section 6.3.2 the TSS states that the Administrative access to the TOE is facilitated through one of several interface: a) Directly connecting to the TOE appliance (locally) b) Remotely connecting to the TOE appliance via SSHv2 c) Remotely connecting to appliance GUI via HTTPS The remote administrators will achieve a successful authentication by providing a valid username and password via Web GUI, and valid username/password or recognized public key via SSH. Direct console to the CLI only supports username/password.

138 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:	Within [ST] section 6.3.2 the TSS states that 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.

139 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:	TOE is not distributed.

140 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:	TOE is not distributed.

3.3.3.2 Guidance Documentation

141 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:	In [AGD] section 2.3.5 the guidance states that: The TOE permits authentication by an administrator through SSH or Web UI over TLS. Authentication is permitted through username/password and public key (for SSH) authentication via local authentication or by a remote authentication server (RADIUS/TACACS+). Authentication to the TOE through a wireless connection does

not permit administration by default.

[AGD] section 2.2.9 describes the preparatory steps required for establishing pre-shared keys for SSH connections.

No user can perform any actions prior to successful authentication to the TOE outside of viewing the warning banner as defined under FTA_TAB.1.

3.3.3.3 Tests

142 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.

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.

High-Level Test Description
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
The TOE is not a distributed TOE.
Findings: N/A

3.3.4 FIA_UAU_EXT.2 Password-based Authentication Mechanism

143 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.

3.3.5 FIA_UAU.7 Protected Authentication Feedback

3.3.5.1 TSS

144 None.

3.3.5.2 Guidance Documentation

145 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:	[AGD] section 2.3.3 states, “No configuration is necessary to obscure feedback of passwords during login. Nothing will be echoed back on the console.”

146

3.3.5.3 Tests

147 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 Security management (FMT)

3.4.1 FMT_MOF.1/ManualUpdate

3.4.1.1 TSS

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

Findings:	Within [ST] section 6.4.1 the TSS states that the TOE restricts the ability to perform software updates to Security Administrators.

3.4.1.2 Guidance Documentation

149 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:	In [AGD] section 2.5.5 the guidance states that: Use the “copy” command to download new firmware images from an FTP or TFTP server and to select the system partition to which the image file is copied. Note that the administrator should first ensure that the boot system partition <partition_id> command is correctly set to specify the system partition number that the controller should use during the next reboot. The following CLI commands transfer the ArubaOS image file: *copy tftp:<tftphost><filename>system:partition[0|1]}* *copy ftp:<ftphost><user><filename>system:partition{0|1}* An option is provided to reboot the device with the transferred image file. From the WebUI, navigate to Maintenance>Software Management>Upgrade page to upload an ArubaOS image from a local filesystem. Specify the system partition to which the image file is copied and choose whether the device should be rebooted when the image file is transferred. Click Upgrade. ArubaOS images are integrity-protected through two evaluated methods: 1. ArubaOS images are digitally signed using RSA 2048-bit signature validation. The Mobility Conductor will check the digital signature immediately after downloading a new firmware image and will refuse to install an image whose digital signature does not match. 2. Mobility Conductors also check the digital signature of an ArubaOS image when booting. The Conductor will refuse to boot a corrupted ArubaOS image file. No configuration is needed to achieve this requirement. If the digital signature verification succeeds, the TOE console will note that the signature has been verified and note that the integrity check on the partition is ‘[OK]’ (Passed). The TOE will continue through initial power on self-tests and after successful completion prompt for authentication.

	If digital signature verification fails, 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.

150 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:	TOE is not distributed.

3.4.1.3 Tests

151 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.

152 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.
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 Security Administrator is able to install legitimate updates.

3.4.2 FMT_MTD.1/CoreData Management of TSF Data

3.4.2.1 TSS

153 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:	Within [ST] section 6.4.3 the TSS states that the users are required to login before being provided with access to any administrative functions.

154 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:	Within [ST] section 6.4.3 the TSS states that the access to TSF data and functions, including managing the TOE's trust store, is restricted to Security Administrators as described by FMT_SMR.2.

3.4.2.2 Guidance Documentation

155 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 c PP is identified, and that configuration information is provided to ensure that only administrators have access to the functions.

Findings:	In [AGD] section 2.4.3 the guidance states that: An administrator with the management role of “root” has full privileges to modify, add, and delete configuration and user accounts. The “root” role maps to the Security Administrator role.

156 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:	[AGD] 2.3.6 describes that the TOE contains a trust store and supports loading of CA certificates. Section 2.4.3 in the guidance states that the trust anchor can be either self-signed or custom certificates installed on the Mobility Conductor. Full details on how the Security Administrator can configure CA certificates as the trust store can be found in the ArubaOS 8.10.0.0 User Guide chapter Management Access, particularly the section regarding Managing Certificates. [GUIDE-1] section “Managing Certificates” in the topic “Importing Certificates” describes the following command for importing certificates to the trust store: The following CLI command imports CSR certificates: *crypto pki-import {der|pem|pfx|pkcs12|pkcs7} {PublicCert|ServerCert|TrustedCA} <name>* Section 2.3.6 of the [AGD] provides sufficient information for the administrator to securely load CA certificates into the trust store. *crypto-local pki TrustedCA intermediate-ca ecdsa-intermediate.cer* *crypto-local pki TrustedCA root-ca ecdsa-root-ca.cer* *crypto-local pki OCSPResponderCert ocsp-root ecdsa-root-ca.cer*

3.4.2.3 Tests

157 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.

3.4.3 FMT_SMF.1 Specification of Management Functions

158 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.

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

159 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: Within [ST] section 6.4.6 the TSS states that the TOE may be managed via the CLI (console & SSH) or GUI (HTTPS). The specific management capabilities include:

a) Ability to administer the TOE locally and remotely

b) Ability to configure the access banner

c) Ability to configure the session inactivity time before session termination or locking

d) Ability to update the TOE and to verify the updates

e) Ability to configure the authentication failure parameters

f) Ability to configure audit behavior (enable/disable remote logging)

g) Ability to configure the cryptographic functionality;

h) Ability to configure the lifetime for IPsec SAs;

i) Ability to configure the reference identifier for the peer;

j) Ability to configure the NTP settings

k) Ability to manage the cryptographic keys, including import and management of X.509v3 certificates

l) Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors

m) Ability to import X.509v3 certificates to the TOE’s trust store

n) Ability to manage the trusted public keys database.

The evaluator reviewed the [AGD] and confirmed that the management functions specified in FMT_SMF.1 are provided by the TOE.

160 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: Within [ST] section 6.4.6 the TSS states that the TOE may be managed via the CLI (local console & SSH) or GUI (HTTPS).

Section 2.3.5 of the [AGD] states “Local console access is directly via the serial console port only.”

161 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:	TOE is not distributed.

3.4.3.2 Guidance Documentation

162 See section 2.4.4.1.

3.4.3.3 Tests

163 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.

3.4.4 FMT_SMR.2 Restrictions on security roles

3.4.4.1 TSS

164 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:	Within [ST] section 6.4.4 the TSS states that the TOE implements role-based access control based on pre-defined profiles that are assigned when creating a user. The 'administrator' role is synonymous with the Security Administrator role defined in this document. Management of TSF data via the CLI or web GUI is restricted to Security Administrators.

3.4.4.2 Guidance Documentation

165 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:	[AGD] section 2.4.5 states, "Please reference the Aruba OS User Guide for a full list of configuration instructions through the CLI and Web GUI." The evaluator reviewed the [AGD] and [GUIDE-1] documentation and ensured that it contains instructions for administering the TOE both locally and remotely. The [AGD] includes CLI instructions which can be used locally through the serial console and remotely through the SSH interface. The [GUIDE-1] includes instructions for performing the same management functions through the WebGUI. [AGD] sections 2.2.9 and 2.2.10 provide configuration instructions required to configure the remote interfaces.

3.4.4.3 Tests

166 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.

3.5 Protection of the TSF (FPT)

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

3.5.1.1 TSS

167 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:	Within [ST] section 6.5.1 the TSS states that the keys are protected as described in "Table 16: Keys". In all cases, plaintext keys cannot be viewed through an interface designed specifically for that purpose. Zeroization consists of a single pass overwrite of zeroes (0).

3.5.2 FPT_APW_EXT.1 Protection of Administrator Passwords

3.5.2.1 TSS

168 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:	Within [ST] section 6.5.2 the TSS states that in all cases plaintext passwords cannot be viewed through an interface designed specifically for that purpose. Password protection is detailed in the chart which is labelled "Table 17: Passwords" in the ST.

3.5.3 FPT_TST_EXT.1 TSF testing

3.5.3.1 TSS

169 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: Within [ST] section 6.5.3 the TSS states that the TOE runs 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 Conductor runs 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 and immediately after generation of a key (FIPS self-tests, including the continuous RNG test).

The following cryptographic KAT's are performed:

a) ArubaOS OpenSSL Module:

i) Algorithm Known Answer Tests

ii) ECDSA (sign/verify)

b) ArubaOS Cryptographic Module

i) Algorithm Known Answer Tests

ii) RSA (sign/verify)

iii) ECDSA (sign/verify)

The following firmware integrity tests are performed:

a) ArubaOS Uboot BootLoader Module

i) Firmware Integrity Test: RSA 2048-bit Signature Validation

The TOE also performs Aruba Hardware Known Answer Tests to ensure the integrity of hardware components.

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

a) 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.

b) 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.

c) 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.

d) ECDSA Pairwise Consistency test. See above RSA pairwise consistency test description.

e) 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.

Known-answer tests (KAT) involve operating the cryptographic algorithm on data for which the correct output is already known and comparing the calculated output with the previously generated output (the known answer). If the calculated output does not equal the known answer, the known-answer test shall fail.

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). The Conductor will reboot after a self-test failure. During reboot, memory is re-initialized, which wipes all keys and user data. If a self-test failure continues to occur, the Conductor will continue to reboot repeatedly and will require return to manufacturer.

The above 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.

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: TOE is not distributed.

3.5.3.2 Guidance Documentation

170 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: In [AGD] section 2.5.4 the guidance states that:

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). The Conductor will reboot after a self-test failure. During reboot, memory is re-initialized, which wipes all keys and user data. If a self-test failure continues to occur, the Conductor will continue to reboot repeatedly and will require return to manufacturer. 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.

171 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:	The TOE is not distributed.

3.5.3.3 Tests

172 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.

173 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.

174 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.

High-Level Test Description
Reboot the Mobility Conductor and watch the output on the serial console as the device boots. Witness that cryptographic and firmware-integrity self-tests are executed successfully.
Findings: PASS

175 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
The TOE is not a distributed TOE.
Findings: N/A

3.5.4 FPT_TUD_EXT.1 Trusted Update

3.5.4.1 TSS

176 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:	Within [ST] section 6.5.4 the TSS states that the TOE allows administrators to query the currently executing version of its firmware/software by issuing the “show version” command, and the most recently loaded but inactive version of the TOE firmware/software by issuing the “show image version” command. The TOE allows administrators to manually initiate firmware/software updates. Prior to installing any

update, the administrator can verify the digital signature of the update.

When an update is initiated, the TOE verifies the digital signature with the stored public key (stored in Boot ROM). Upon successful verification, the TOE boots using the new image.

177 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: Within [ST] section 6.5.4 the TSS states that the TOE allows administrators to query the currently executing version of its firmware/software by issuing the “show version” command, and the most recently loaded but inactive version of the TOE firmware/software by issuing the “show image version” command.

The TOE allows administrators to manually initiate firmware/software updates. Prior to installing any update, the administrator can verify the digital signature of the update.

Administrators can update the TOE executable code using image files manually downloaded from the Aruba support portal. The administrator may perform 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.

A SHA-256 hash of each update image is digitally signed using Aruba’s code signing certificate (RSA 2048 bit). The signing certificate is issued by Aruba’s internal CA. Aruba’s code signing public key is programmed into the Boot ROM of all Aruba products at the time of manufacturing.

When an update is initiated, the TOE verifies the digital signature with the stored public key (stored in Boot ROM). Upon successful verification, the TOE boots using the new image. Should verification fail, 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.

178 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: Automatic updates not supported.

179 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:	TOE not distributed.

180 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:	A published hash is not used. Rather, the TOE uses a digital signature mechanism.

3.5.4.2 Guidance Documentation

181 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:	In [AGD] section 2.5.5 the guidance states that: Use the command “show version” to view the active version and “show image version” to view the active version and loaded but inactive version.

182 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:	In [AGD] section 2.5.5 the guidance states that: ArubaOS images are integrity-protected through two evaluated methods: 1. ArubaOS images are digitally signed using RSA 2048-bit signature validation. The Mobility Conductor will check the digital signature immediately after downloading a new firmware image and will refuse to install an image whose digital signature does not match. 2. Mobility Conductors also check the digital signature of an ArubaOS image when booting. The Conductor will refuse to boot a corrupted ArubaOS image file. No configuration is needed to achieve this requirement. If the digital signature verification succeeds, the TOE console will note that the signature has been verified and note that the integrity check on the partition is ‘[OK]’ (Passed). The TOE will continue through initial power on self-tests and after successful completion prompt for authentication. If digital signature verification fails, 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.

183 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:	The TOE does not support published hash checking for updates.

184 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:	TOE is not distributed.

185 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:	TOE is not distributed.

186 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 [ST] does not claim a certificate-based mechanism for software updates. [ST] section 6.5.4 the TSS states, “Aruba’s code signing public key is programmed into the Boot ROM of all Aruba products at the time of manufacturing.”

3.5.4.3 Tests

187 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.

High-Level Test Description
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.

188 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.

High-Level Test Description
The TOE does not support verification of published hashes.

High-Level Test Description
Findings: N/A

189 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).

High-Level Test Description
The TOE only claims manual updates as previously tested.
Findings: N/A

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

High-Level Test Description
The TOE is not a distributed TOE.
Findings: N/A

3.5.5 FPT_STM_EXT.1 Reliable Time Stamps

3.5.5.1 TSS

191 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.

NIAP TD0632

192 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:	Within [ST] section 6.5.5 the TSS states that the TOE has an internal battery-backed hardware clock that provides reliable time stamps used for auditing, session timeouts, and certificate validation. The internal clock can be synchronized with a time signal obtained from an external NTP server. The TOE makes use of time for the following: a) Audit record timestamps b) Session timeouts (lockout enforcement) c) Certificate validation

3.5.5.2 Guidance Documentation

193 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.

NIAP TD0632

194 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:	[AGD] section 2.2.7 provides instructions on establishing an IPsec communication path with an external IT entity. This is used to configure the trusted channel between the TOE and the NTP server. [AGD] section 2.2.6 provides instructions for configuring the NTP server through the GUI: ...the Security Administrator can navigate to the WebUI Configuration > System > General > Clock page. Click the '+' under NTP servers and fill in the information to set additional time sources. In section [AGD] 2.5.3 the guidance provides instructions for configuring the NTP server through CLI: Mobility Conductors require clock synchronization using NTPv4 in order to generate reliable timestamps. To specify an NTP server: *(config) # ntp server <IP address>* *(config) # ntp server <IP address>* More NTP options, including authentication, are available. The ArubaOS User Guide can be consulted for more information. The administrator must ensure the connection to the time server is secured with IPsec. The TOE is not a vND and does not support obtaining time from the underlying VS.

3.5.5.3 Tests

195 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
The TOE does not support direct setting of time by a security administrator.
Findings: N/A

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.
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.

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 virtual Network Device and does not obtain time from the underlying VS.
Findings: N/A

196 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
The TOE does not support independent time information.
Findings: N/A

3.6 TOE Access (FTA)

3.6.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

3.6.1.1 TSS

197 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:	Within [ST] section 6.6.1 the TSS states that the Security Administrator may configure the TOE to terminate an inactive local interactive session (CLI) following a specified period of time.

198

3.6.1.2 Guidance Documentation

199 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:	In [AGD] section 2.6.3 the guidance states that: For local administrative sessions, an idle timeout may be set to disconnect idle sessions. The default value is 300 seconds (5 minutes). To configure the timer value, use the following at the CLI: *(config) #loginsession timeout <value> sec* In the above command, *<value>* can be any number of seconds from 30 to 3600, inclusive. Additionally, the administrator can choose to configure the value from 1-60 minutes by excluding the 'seconds' parameter: *(config) #loginsession timeout <value>* Following configuration of the timeout command, the security administrator should enter the following command to ensure the configuration takes effect:*(config) #write memory*

3.6.1.3 Tests

200 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 Conductor. 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.

3.6.2 FTA_SSL.3 TSF-initiated Termination

3.6.2.1 TSS

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

Findings:	Within [ST] section 6.6.2 the TSS states that the Security Administrator may configure the TOE to terminate an inactive remote interactive session (CLI and Web UI) following a specified period of time.

3.6.2.2 Guidance Documentation

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

Findings:	[AGD] section 2.6.1 provides instructions for configuring the inactivity time period for remote interactive sessions: For remote administrative sessions, an idle timeout may be set to disconnect idle sessions. The default value is 300 seconds (5 minutes). To configure the timer value, use the following: For the SSH CLI: *(config) #loginsession timeout <value> sec* In the above command, <value> can be any number of seconds from 30 to 3600, inclusive. Additionally, the administrator can choose to configure the value from 1-60 minutes by excluding the 'seconds' parameter: *(config) #loginsession timeout <value>* Following configuration of the timeout command, the security administrator should enter the following command to ensure the configuration takes effect: *(config) #write memory* For the WebUI: *(config) #web-server profile* *(Web server configuration) #session-timeout <val>* In the above command, <val> can be any number of seconds from 30 to 3600, inclusive. In addition to the CLI, the administrator can configure the WebUI Idle Session timeout by navigating to Configuration > System > Admin >Admin Authentication Options. This parameter can be configured as either seconds or minutes (1-60 minutes, or 30-3600 seconds).

3.6.2.3 Tests

203 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 Conductor. 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

High-Level Test Description
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.

3.6.3 FTA_SSL.4 User-initiated Termination

3.6.3.1 TSS

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

Findings:	Within [ST] section 6.6.3 the TSS states that administrative users may terminate their own sessions at any time by providing a logout command (CLI) and logout option under user menu (GUI).

205

3.6.3.2 Guidance Documentation

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

Findings:	In [AGD] section 2.6.2 the guidance states that: To logout from the CLI, an administrator can just enter the 'exit' command. In order to logout from the WebUI session, the administrator should select their username in the top right corner and 'log out' from the dropdown menu.

3.6.3.3 Tests

207 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.

3.6.4 FTA_TAB.1 Default TOE Access Banners

3.6.4.1 TSS

208 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:	Within [ST] section 6.6.4 the TSS states that the TOE displays an administrator configurable message to users prior to login at the CLI (local console and SSH) and web GUI (HTTPS).

3.6.4.2 Guidance Documentation

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

Findings:	In [AGD] section 2.6.4 the guidance states that: See FIA_UIA_EXT.1 above for a description of how to configure a notice and consent banner message. Instructions for FIA_UIA_EXT.1 are located in [AGD] 2.3.5. This section has instructions for setting up a banner. A warning banner may be configured as follows. Ensure that no line is longer than 255 characters. *#configure terminal* *Enter Configuration commands, one per line. End with CNTL/Z* *(config) #banner motd =* *Enter TEXT message [maximum of 4095 characters].* *Each line in the banner message should not exceed 255 characters.* *End with the character '='.* … Following configuration of product functionality through the CLI, the security

	administrator should enter the following command to ensure the configuration takes effect: *(config) #write memory* For configuration of the TOE login banner through the WebUI, the administrator should navigate to Configuration > System > Admin > Admin Authentication Options > Login banner text.

3.6.4.3 Tests

210 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.

3.7 Trusted path/channels (FTP)

3.7.1 FTP_ITC.1 Inter-TSF trusted channel

3.7.1.1 TSS

211 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:	Within [ST] section 6.7.1 the TSS describes that the TOE uses the IPsec protocol with certificates to establish VPN tunnels for trusted channels between external IT entities. The TOE supports secure communication via IPsec with the following IT entities: a) Audit server. The TOE initiates communication and acts as a client. b) Authentication server. The TOE initiates communication and acts as a client. c) Aruba Mobility Controllers. The TOE initiates communication and acts as a client. d) NTP server. The TOE initiates communication and acts as a client. The TOE operates as an IPsec peer and has the ability to act as an initiator.

The evaluator determined that all secure communication mechanisms are described in sufficient detail to match them to the cryptographic protocol SFRs listed in the ST.

3.7.1.2 Guidance Documentation

212 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 [AGD] section 2.7.1 the guidance states that:

ArubaOS supports IPsec as the inter-TSF trusted channel. This channel is to be used between a Mobility Conductor and a) a syslog server, b) an authentication server (RADIUS or TACACS+), c) NTPv4 server and d) VPN Gateway/Mobility Controller.

If a connection is unintentionally broken, the TOE will re-establish it once it is restored. If the timeout period has expired, re-authentication/re-negotiation is required.

[AGD] section 2.2.7 provides instructions for establishing IPsec connections with each authorized IT entity.

3.7.1.3 Tests

213 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.

214 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, NTP server and Aruba Controller. 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
Ensure there is a viable IPsec connection between the TOE and a non-TOE IPSec terminator. Physically disconnect the TOE from the network and plug it back in as soon as the network interface link-light turns off. Then physically disconnect the TOE from the network for 30 minutes. Plug the connection back in and show that the traffic will flow over IPsec again without showing plaintext.
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.

215 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.

Findings:	This is not a distributed TOE.

216 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.

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

3.7.2 FTP_TRP.1/Admin Trusted Path

3.7.2.1 TSS

217 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:	Within [ST] section 6.7.2 the TSS states that the TOE provides the following trusted paths for remote administration: a) CLI over SSH b) Web GUI over HTTPS The evaluator confirmed that all protocols listed in the TSS in support of the TOE administration are consistent with those specified in the requirement, and are included in the requirements in the ST.

3.7.2.2 Guidance Documentation

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

Findings:	In [AGD] section 2.7.2 the guidance states that: Communication between a Mobility Conductor and a remote administrator may be protected by TLS/HTTPS (when using the Web-based interface) or SSH (when using the command-line interface). All remote administration must take place over one of these interfaces. To access the SSH CLI interface: 1. Initialize the SSH client 2. For the hostname, specify the TOE IP and port 22. 3. Begin session establishment. 4. The administrator will be prompted for username and password/public key

upon establishing a successful connection.

To access the local serial interface:
1. Configure the terminal or terminal emulation program to use the following communication settings: Baud: 9600, Data Bits: 8, Parity: None, Stop Bits: 1, Flow Control: None
2. Connect the terminal or PC/workstation to the serial port on the devices using an RS-232 serial cable. RJ-45 cable and DB-9 to RJ-45 adapter is required. The administrator may need a USB adapter to connect the serial cable to the PC.
3. After the connection initialized, the administrator will be able to continue through the CLI after entering valid credentials.

To access the WebUI, the administrator should navigate to their approved web browser and connect to https://<TOEIP:PORT> or the specified FQDN. Once connected, the administrator must authenticate to the device before proceeding with any further access requests.

The following browsers are officially supported for use with the ArubaOS WebUI:
- Microsoft Internet Explorer 11 on Windows 7 and Windows 8
- Microsoft Edge (Microsoft Edge 92.0.902.62 and Microsoft EdgeHTML 18.19041) on Windows 10
- Firefox (91.0) on Windows 7, Windows 8, Windows 10, and macOS
- Apple Safari 8.0 or later on macOS
- Google Chrome (92.0.4515.131) on Windows 7, Windows 8, Windows 10, and macOS

[AGD] section 2.2.9 provides instructions for setting up the TOE for SSH connections. Previously verified in FCS_SSHS_EXT.1 assurance activities.

[AGD] section 2.2.10 provides instructions for setting up the TOE for HTTPS/TLS connections. Previously verified in FCS_TLSS_EXT.1 assurance activities.

3.7.2.3 Tests

219 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).

220 Further assurance activities are associated with the specific protocols.

221 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
The TOE is not a distributed TOE.
Findings: N/A

4 Evaluation Activities for Optional Requirements

222 No optional requirements have been selected by this evaluation.

5 Evaluation Activities for Selection-Based Requirements

5.1 Cryptographic Support (FCS)

5.1.1 FCS_HTTPS_EXT.1 HTTPS Protocol

5.1.1.1 TSS

223 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 6.2.8 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.

5.1.1.2 Guidance Documentation

224 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:	In [AGD] section 2.2.5 the guidance states that: No configuration is required. The TOE will function as an HTTPS server, compliant to RFC 2818, when operating under FIPS mode.

5.1.1.3 Tests

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

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

227 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.

5.1.2 FCS_IPSEC_EXT.1 IPsec Protocol

5.1.2.1 TSS

FCS_IPSEC_EXT.1.1

228 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.

229 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 6.2.9 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

230 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 6.2.9 the TSS states that the TOE includes an implementation of IPsec in accordance with RFC 4301 for security. The TOE supports IPsec for tunnel mode.

FCS_IPSEC_EXT.1.4

231 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 6.2.9 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 SHA-based HMAC algorithms: HMAC-SHA-1-96 and with AES-GCM-128 and AES-GCM-256 (as specified by RFC 4106). Theis is consistent with claim made in FCS_COP.1/KeyedHash for HMAC-SHA1.

FCS_IPSEC_EXT.1.5

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

233 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 6.2.9 the TSS states that the TOE implements IKEv2, with support for NAT traversal, as defined in RFC 5996 and RFC 4868 for hash functions.

FCS_IPSEC_EXT.1.6

234 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 6.2.9 the TSS states that the TOE uses the AES-CBC-128 and AES-CBC-256 algorithms as specified in RFC 3602 to encrypt the payload.

FCS_IPSEC_EXT.1.7

235 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 6.2.9 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. The selections correspond to the selection in FCS_IPSEC_EXT.1.5.

FCS_IPSEC_EXT.1.8

236 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:	See findings for FCS_IPSEC_EXT.1.7 above.

FCS_IPSEC_EXT.1.9

237 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 6.2.9 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

238 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 6.2.9 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.

239 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 6.2.9 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

240 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 6.2.9 the TSS states that the TOE implements IKEv2, with support for NAT traversal, as defined in RFC 5996 and RFC 4868 for hash functions. Diffie-Hellman (DH) Groups 14, 19, and 20 are supported as are RSA and ECDSA certificates and pre-shared key IPsec authentication. 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

241 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 6.2.9 the TSS states that IPsec ESP protocol is implemented in conjunction with AES-CBC-128 and AES-CBC-256 (as specified by RFC 3602) together with the following SHA-based HMAC algorithms: HMAC-SHA-1-96 and with AES-GCM-128 and AES-GCM-256 (as specified by RFC 4106). 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

242 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 6.2.9 the TSS states that Diffie-Hellman (DH) Groups 14, 19, and 20 are supported as are RSA and ECDSA certificates and pre-shared key IPsec authentication. This is consistent with selections made in FCS_COP.1/SigGen.

243 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:	Within [ST] section 6.2.9 the TSS states that the TOE is not capable of generating its own pre-shared keys and requires suitable keys to be entered by an authorized administrator using a Web GUI or CLI function. The pre-shared key is authenticated over IPsec protocol.

FCS_IPSEC_EXT.1.14

244 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 6.2.9 the TSS states that the TOE will only establish a trusted IPsec channel if the presented identifier in the received certificate matches the configure 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.

5.1.2.2 Guidance Documentation

FCS_IPSEC_EXT.1.1

245 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:	In [AGD] section 2.2.7.1 the guidance states that: RFC 4301 references an explicit Security Policy Database (SPD) with rules for DISCARD, BYPASS, and PROTECT. ArubaOS does not implement an explicit SPD, but equivalent behavior may be obtained through the use of firewall policies and "routing" ACLs. In the following configuration, ICMP echo-request traffic from Client A to Client C takes the BYPASS action. All other ICMP traffic between the same hosts takes the PROTECT action. HTTP traffic is dropped. *ip access-list route spd-test*

host <IP address> host <IP address> icmp echo forward
host <IP address> host <IP address> svc-icmp route ipsec-map <IP address>
host <IP address> host <IP address> svc-http route tunnel 10
interface vlan <vlanid>
ip address <IP address> <subnet>
operstate up
ip access-group "spd-test" in

!

Modify these rules as needed if explicit control over tunneled and non-tunneled traffic is needed. Note: Most deployments will not make use of this feature, as ALL traffic to a specific destination will typically be tunneled. The sample config file at the end of this document does NOT contain examples from this section.

The configuration above provides SPD control for inbound wired traffic. For wireless or VPN client users (not tested as part of the Common Criteria evaluation), multiple ACLs may be sequenced with a user-role container, simplifying this configuration.)

The access control lists used by the TOE are read in hierarchical order. When traffic enters or exits the TOE, the first applicable rule in the ACL is applied. Any rule below the initially triggered rule is not applied. 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.

The evaluator determined that the description was consistent with the TSS and is sufficient to allow the administrator to set up the SPD in an unambiguous fashion.

FCS_IPSEC_EXT.1.3

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

Findings: In [AGD] section 2.2.7.2 the guidance states:

For additional assurance that only tunnel mode is used, the following command should be used under the crypto-local ipsec-map to force tunnel mode to be the only option offered.

force-tunnel-mode

With this command present, the crypto map would show the following:

Transform set transform-tunnel: { esp-aes128 esp-sha-hmac }
will negotiate = { Tunnel }

FCS_IPSEC_EXT.1.4

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

Findings: In [AGD] section 2.2.7.3 the guidance states that:

IPsec cipher suites are configured using transform-sets. These are ordered lists of ciphers - the Conductor will attempt each one in order until one is successfully negotiated with the peer. The command "show crypto ipsec transform-set" will display the configured transform sets.

ArubaOS provides pre-configured transforms that meet three of the Common Criteria requirements. Note that the Advanced Cryptography License must be installed to have access to AES-GCM. The default transforms are:

Transform set default-gcm256: { esp-aes256-gcm }

Transform set default-gcm128: { esp-aes128-gcm }

Transform set default-aes: { esp-aes256 esp-sha-hmac }

Note: The TOE's IPsec ESP protocol implementation supports only HMAC-SHA-1. The IKE protocol supports HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384 (see FCS_IPSEC_EXT.1.6 below).

To configure AES-CBC-128, add a new transform set:

(config) #crypto ipsec transform-set aes128 esp-aes128 esp-sha-hmac

FCS_IPSEC_EXT.1.5

248 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: In [AGD] section 2.2.7.4 the guidance states:

IKEv2 is supported for use in the evaluated configuration. NAT Traversal (NAT-T) is supported for both. NAT-T transports packets over UDP port 4500 rather than using IPsec native encapsulation.

For inbound connections where the Conductor is the IKE responder, NAT-T is supported by default. To disable, install a firewall rule that blocks UDP 4500.

For outbound connections in a site-to-site VPN tunnel, NAT-T is configured in the ipsec-map described in FCS_IPSEC_EXT.1.2. To force NAT-T rather than allowing it to be negotiated, issue the following command:

(config) #crypto-local ipsec-map 10.10.20.1 100
(config-ipsec-map)# force-natt enable

To specify the IKEv2 policy:

(config) #crypto isakmp policy <priority>
(config-isakmp) #version v2

249 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: N/A—IKEv1 is not supported in the evaluated configuration.

FCS_IPSEC_EXT.1.6

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

Findings: In [AGD] section 2.2.7.5 the guidance states:

IKE policies are matched in numerical order, with lower numbers having higher priority. Several IKE policies are pre-configured - to view these, issue the command "show crypto isakmp policy".

Default policies may not be deleted but may be disabled. To disable a policy:

(config) #crypto isakmp policy <policy>

(config-isakmp)# disable

It is recommended that when deployed as a VPN gateway, all default IKE policies be disabled, and only user-defined policies configured for use.

To configure an IKEv2 policy that uses AES-256, issue the following commands:

(config) # crypto isakmp policy 100

(config-isakmp)# encryption aes256

(config-isakmp)# hash sha

(config-isakmp)# version v2

To configure AES128, adjust the encryption to 'encryption aes128'.

To configure HMAC-SHA-256 or HMAC-SHA-384, adjust the hash to 'sha2-256-128' or 'sha2-384-192'.

FCS_IPSEC_EXT.1.7

NIAP TD0633

251 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: In [AGD] section 2.2.7.6 the guidance states:

Phase 1 (IKE) lifetimes are configured in the IKE policies. To adjust the previously-created IKE policy for a 24-hour lifetime (this is the default value), issue the following commands:

(config) # crypto isakmp policy 100
(config-isakmp)# lifetime 86400

FCS_IPSEC_EXT.1.8

NIAP TD0633

252 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: In [AGD] section 2.2.7.6 the guidance states:

Phase 2 (IPsec) lifetimes are configured in the ipsec-map (for site-to-site):

(config) #crypto-local ipsec-map 10.10.20.1 100
(config-ipsec-map)# set security-association lifetime seconds 28800
…
SA lifetimes may also be configured based on the number of bytes transmitted. Replace the keyword "seconds" with "kilobytes" in the above configuration and supply the lifetime value. Both time-based and volume-based lifetimes may be configured simultaneously.

FCS_IPSEC_EXT.1.11

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

Findings: In [AGD] section 2.2.7.8 the guidance states:

ArubaOS supports DH groups 14, 19, and 20. To configure, modify the IKE policy:

(config) # crypto isakmp policy 100

(config-isakmp)# group 20

FCS_IPSEC_EXT.1.13

254 The evaluator ensures the guidance documentation describes how to set up the TOE to use certificates with RSA and/or ECDSA signatures and public keys.

The evaluator shall check that the guidance documentation describes how pre-shared keys are to be generated and established. The description in the guidance documentation shall also indicate how pre-shared key establishment is accomplished for TOEs that can generate a pre-shared key as well as TOEs that simply use a pre-shared key.

The evaluator will ensure that the guidance documentation describes how to configure the TOE to connect to a trusted CA, and ensure a valid certificate for that CA is loaded into the TOE and marked “trusted”.

Findings: In [AGD] section 2.2.7.9 the guidance states:

ArubaOS supports both RSA and ECDSA certificates. Note that the Advanced Cryptography License must be installed to make use of ECDSA.

Loading certificates onto the Conductor for both authentication to peers and for verification of other peers is described in the User Guide. Minimally, both a "server certificate" and a "trusted root CA" certificate must be loaded onto the Conductor to perform IPsec operations. Once these certificates are loaded on the Conductor, configure them for use in IPsec. For use with dynamic VPN clients:

(config) #crypto-local isakmp server-certificate "server-cert"

(config) #crypto-local isakmp ca-certificate "trusted-root-ca-cert"

For a site-to-site VPN tunnel:

(config) #crypto-local ipsec-map 10.10.20.1 100

(config-ipsec-map)# set server-certificate server-cert

(config-ipsec-map)# set ca-certificate root-ca

To configure an IKE policy to authenticate RSA certificates sent by peers, use the following command:

(config) #crypto isakmp policy 100

(config-isakmp)# authentication rsa-sig

To configure an IKE policy for ECDSA-384 authentication, use the following command:

(config) #crypto isakmp policy 100

(config-isakmp)# authentication ecdsa-384

ECDSA-256 may be supported by replacing "384" with "256".

Administrators should take care to configure IKE/IPsec policies so that the strength of the IKE association is greater than or equal to the strength of the IPsec tunnel (for example, by always using AES-256). However, if a misconfiguration is made, the Conductor will reject the security association along with generating an audit log message.

When the IPsec connection is configured to use pre-shared keys, the administrator can follow the following steps to configure a pre-shared key on the TOE:

To configure the key, pick one of the following options:

(config) #crypto-local isakmp key DEADBEEF01010202abc!@# address 0.0.0.0 netmask 0.0.0.0

When configuring the pre-shared key, the administrator must ensure that the PSK is at least 22 characters, contains at least one uppercase character, one lowercase character, one special character, and one digit. If the PSK is configured as a bit-based key, the ‘key-hex’ field should be used instead.

(config) #crypto-local isakmp key-hex DEADBEEF01010202ABA010 address 0.0.0.0 netmask 0.0.0.0

The TOE does not connect to an external CA for any PKI operations except for OCSP revocation checking. In section 2.3.6 the guidance documentation states:

When a root CA or intermediate CA certificate is loaded on the Conductor, an automatic Revocation Check Point (RCP) section is created in the configuration file. These may be shown using “show crypto-local pki rcp”. For each RCP, the revocation check method may be configured, and may be set to none, crl, or ocsp.

If OCSP has been specified, then an OCSP responder URL and OCSP responder certificate must be specified. In the evaluated configuration, the TOE does not accept certificates if an OCSP responder is unreachable.

FCS_IPSEC_EXT.1.14

255 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: In [AGD] section 2.2.7.10 the guidance states:

The TOE does not support the SAN extension.

To configure the TOE reference identifier for the distinguished name of the peer, an administrator may use the following commands:

(config) #crypto-local ipsec-map testmap 1

(config-submode)#peer-cert-dn

<peer-dn> Subject-Name DN string of the Peer's Certificate

To ensure appropriate compliance within the evaluated configuration, the administrator should generate a CA chain with one Root CA and two Intermediate CAs. The OCSP configuration and information on this can be found under Sections 2.3.6 and 2.3.7.

5.1.2.3 Tests

FCS_IPSEC_EXT.1.1

256 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
Configure NTP to point to the remote NTP server via the VPN IP addressing and show the packets are encrypted.

High-Level Test Description
Configure NTP to point to the remote NTP server directly and show the packets are not encrypted. Configure an IP ACL to drop specific NTP 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

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

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

259 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 NTP to point to the remote NTP directly and show the packets are not encrypted. Configure an IP ACL to permit all NTP traffic only for that destination IP and apply it to the interface. Configure NTP to point to the remote NTP server directly using a different destination IP address and show the packets are not delivered and are dropped by the TOE.
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

260 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

261 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

262 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

263 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.
Findings: PASS – The evaluator confirmed that a successful connection with each of claimed IKEv2 algorithms and HMAC functions.

FCS_IPSEC_EXT.1.7

264 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."

265 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

266 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."

267 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 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

268 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.

Findings:	[ST] section 6.2.9 states, "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." The evaluator confirmed the TSS indicates that the random number generated meets the requirements in the PP and the length of the nonces meet the stipulations in the requirement.

b) Test 2: If the second selection is chosen, the evaluator shall check to ensure that, for each PRF hash supported, the TSS describes the process for generating each nonce. The evaluator shall verify that the TSS indicates that the random number

generated that meets the requirements in this PP is used, and that the length of the nonces meet the stipulations in the requirement.

Findings:	[ST] section 6.2.9 states, "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." The evaluator confirmed the TSS indicates that the random number generated meets the requirements in the PP and the length of the nonces meet the stipulations in the requirement.

FCS_IPSEC_EXT.1.11

269 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

270 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 test 1. The TOE only claims IKEv2.
Findings: PASS – The evaluator confirmed in FCS_IPSEC_EXT.1.6 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.

High-Level Test Description
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

271 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

272 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.

273 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.
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).

5.1.3 FCS_NTP_EXT.1 NTP Protocol

5.1.3.1 TSS

FCS_NTP_EXT.1.1

274 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.

Findings:	Within [ST] section 6.2.10 the TSS states that the TOE supports NTP v4 by implementing the ntpd Linux daemon in client mode. The TOE supports IPsec between itself and the NTP server to provide trusted communications.

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:	See above findings in FCS_NTP_EXT.1.1.

5.1.3.2 Guidance Documentation

FCS_NTP_EXT.1.1

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

Findings:	In [AGD] section 2.2.6 the guidance documentation states that the TOE supports the usage of NTPv4 by default. Mobility Conductors require clock synchronization using NTPv4 in order to generate reliable timestamps. To specify an NTP server: *(config) # ntp server <IP address>* *(config) # ntp server <IP address>* *(config) # ntp server <IP address>* To remove any of the configured servers, the following command can be used: *(config) # no ntp server <IP address>* The TOE supports configuration of 3 NTP time sources. Multiple time servers can be configured with the use of the 'ntp server' command shown above.

FCS_NTP_EXT.1.2

276 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:	In [AGD] section 2.5.3 the guidance documentation describes what protocol to use to ensure the integrity of the timestamp: The administrator must ensure the connection to the time server is secured with IPsec. [AGD] Section 2.2.7 provides instructions on how the administrator can configure the TOE to use the IPsec protocol. Please see AAR section 5.1.2.2 for AGD activities related to FCS_IPSEC_EXT.1.

FCS_NTP_EXT.1.3

277 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:	In [AGD] section 2.2.6 the guidance documentation states “The TOE by default does not accept broadcast and multicast NTP packets.”

5.1.3.3 Tests

FCS_NTP_EXT.1.1

278 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

279 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.

High-Level Test Description
The TOE claims support for cryptographic channel transport using IPSec.
Findings: PASS – IPsec was tested as part of FCS_IPSEC_EXT.1 Evaluation Activities.

[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
The ST does not claim support for the use of message digest algorithms in element 1.2.
Findings: N/A

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. 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

280 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.

5.1.4 FCS_SSHS_EXT.1 SSH Server

5.1.4.1 TSS

FCS_SSHS_EXT.1.2

NIAP TD0631

281 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).

282 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.

283 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 6.2.12 the TSS states that the TOE supports password-based or public key (ssh-rsa, rsa-sha2-256, and rsa-sha2-512) authentication which is consistent with the signature verification algorithms selected in FCS_COP.1/SigGen.

FCS_SSHS_EXT.1.3

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

Findings:	Within [ST] section 6.2.12 the TSS states that the TOE examines the size of each received SSH packet. If the packet is greater than 256KB, it is automatically dropped.

FCS_SSHS_EXT.1.4

285 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 6.2.12 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

286 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:	Within [ST] section 6.2.12 the TSS states that the TOE uses ssh-rsa, rsa-sha2-256, and rsa-sha2-512 for server (host) public key authentication. This is identical to claims in the SFR.

FCS_SSHS_EXT.1.6

287 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 6.2.12 the TSS states that the TOE provides data integrity for SSH connections via HMAC-SHA1 and HMAC-SHA2-256. The list corresponds to the selections made for the component.

FCS_SSHS_EXT.1.7

288 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 6.2.12 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

289 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 an aggregate or 1 gig of data has been exchanged (whichever occurs first).

5.1.4.2 Guidance Documentation

FCS_SSHS_EXT.1.4

290 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:	In [AGD] section 2.2.9 the guidance states: SSH access requires that you configure an IP address and a default gateway. No configuration is needed to specify the permitted algorithms after 'fips enable' has been set. The conductor will attempt negotiations using AES128-CBC, AES256-CBC, AES128-CTR, and AES256-CTR. To ensure correct configuration, un-claimed macs and key exchange algorithms must be disabled, leaving only: Hmac-sha1, Hmac-SHA2-256 ECDH-sha2-nistp256, ECDH-sha2-nisp384 To view configuration for SSH, the following command can be used: *show ssh* To configure the SSH server, the following commands should be used: *ssh disable_dsa* *Ssh disable-kex dh* *Ssh disable-mac hmac-sha1-96* *ssh mgmt-auth {public-key [username/password]|username/password [public-key]}* To configure authentication for SSH using public key, the following commands can be used:

ssh mgmt-auth public-key

mgmt-user ssh-pubkey client-cert ssh-pubkey cli-admin root

Full instructions on how to upload a X.509-containerized public key for SSH authentication are included in the ArubaOS 8.10.0.0 User Guide.

FCS_SSHS_EXT.1.5

291 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: [AGD] Section 2.2.9 of the guidance states:
To configure the SSH server, the following commands should be used:
ssh disable_dsa
ssh mgmt-auth {public-key [username/password]|username/password [public-key]}

To configure authentication for SSH using public key, the following commands can be used:

ssh mgmt-auth public-key
mgmt-user ssh-pubkey client-cert ssh-pubkey cli-admin root

No configuration is needed to specify the permitted algorithms after 'fips enable' has been set.

FCS_SSHS_EXT.1.6

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

Findings: [AGD] Section 2.2.9 of the guidance states:
To ensure correct configuration, un-claimed macs and key exchange algorithms must be disabled, leaving only:

Hmac-sha1, Hmac-SHA2-256

…

To view configuration for SSH, the following command can be used:

show ssh

To configure the SSH server, the following commands should be used:

ssh disable_dsa
Ssh disable-kex dh
Ssh disable-mac hmac-sha1-96
…
Note: The TOE does not support the "none" MAC algorithm.

FCS_SSHS_EXT.1.7

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

Findings:	[AGD] Section 2.2.9 of the guidance states: To ensure correct configuration, un-claimed macs and key exchange algorithms must be disabled, leaving only: Hmac-sha1, Hmac-SHA2-256 ECDH-sha2-nistp256, ECDH-sha2-nisp384 To view configuration for SSH, the following command can be used: show ssh To configure the SSH server, the following commands should be used: ... *Ssh disable-kex dh* …

FCS_SSHS_EXT.1.8

294 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:	[AGD] Section 2.2.9 of the guidance states: SSH rekey intervals are non-configurable and are set to a maximum time interval of one (1) hour or 512M, whichever occurs first.

5.1.4.3 Tests

FCS_SSHS_EXT.1.2

NIAP TD0631

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

296 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
Verify TOE allows SSH client to authenticate using all supported public-key algorithms.
Findings: PASS – The evaluator confirmed all supported public-key algorithms were able to authenticate to the TOE.

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
Verify that using an SSH key pair that has not been configured on the TOE results in an authentication failure.
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

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.

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

297 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
Transmit a packet larger than allowed by the TOE SSH implementation and verify that the TOE rejects the packet.
Findings: PASS - The evaluator confirmed the TOE rejects SSH packets larger than 256KB.

FCS_SSHS_EXT.1.4

298 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
Connect to the TOE using each claimed SSH cipher. Verify the TOE only proposes the claimed ciphers.
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

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

300 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, connect to the TOE server using each hostkey algorithm and verify each connection succeeds.
Findings: PASS - The evaluator confirmed the TOE successfully identifies itself with each hostkey algorithm specified in the ST.

~~301 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.~~

302 Has effectively been moved to FCS_SSHS_EXT.1.2.

303 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.

304 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, connect to the TOE server using the specified public key algorithms in turn. This requires the TOE to be loaded with a public key corresponding to the key pair.
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

305 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.

306 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
Attempt to negotiate each claimed integrity algorithm and show that each algorithm is used in a successful connection.
Findings: PASS - The evaluator confirmed the TOE successfully establishes an SSH connection with each claimed integrity algorithm.

307 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.

308 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 HMAC 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

309 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
Attempt to negotiate an SSH connection with diffie-hellman-group1-sha1 and verify the connection fails.
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.

310 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
Attempt to negotiate each claimed key exchange method and show that each method is used in a successful connection.
Findings: PASS - The evaluator confirmed the TOE successfully establishes a connection using ecdh-sha2-nistp256 and ecdh-sha2-nistp384.

FCS_SSHS_EXT.1.8

311 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.

312 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).

313 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.
Findings: PASS - The evaluator confirmed the TOE initiates a rekey before 1 hour is exceeded.

314 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).

315 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).

316 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, trickle data to the SSH server and detect a rekey initiated by the TOE SSH server.

High-Level Test Description
Findings: PASS - The evaluator confirmed the TOE initiates a rekey before 512MB of data has been encrypted or decrypted using a key.

317 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.

318 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.

5.1.5 FCS_TLSS_EXT.1 Extended: TLS Server Protocol

5.1.5.1 TSS

FCS_TLSS_EXT.1.1

319 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 6.2.13 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

• TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

The ciphersuites are identical to those listed for this component.

FCS_TLSS_EXT.1.2

320 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 6.2.13 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

321 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 6.2.13 the TSS states that the TOE performs key establishment using ECDHE curves secp256r1.

FCS_TLSS_EXT.1.4

322 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 6.2.13 the TSS states that the TOE supports session resumption based on session IDs according to RFC 5246 and session tickets according to RFC 5077.

323 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 6.2.13 states that session tickets are protected by implementing symmetric encryption algorithms as described in FCS_COP.1/DataEncryption and [ST] section 6.2.4. Session tickets are encrypted according to the TLS negotiated symmetric encryption algorithm.

324 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 6.2.13 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 6.2.13 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.

5.1.5.2 Guidance Documentation

FCS_TLSS_EXT.1.1

325 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:	[AGD] Section 2.2.10 of the guidance states: No configuration is required to set the permitted cipher suites or the associated key agreement parameters once 'fips enable' has been entered on the Conductor. The Conductor negotiates using the following ciphersuites: 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, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289 To view configuration for TLS, the following command can be used: *show web-server profile* *show web-server statistics* The following commands can be used to configure the TLS web-server profile: *web-server profile* *absolute-session-timeout <30-3600>* *ciphers high* *mgmt-auth username/password* *session-timeout <30-3600>* *ssl-protocol tlsv1.2* *web-max-clients <25-320>* *web-https-port-443* *switch-cert <name>*

FCS_TLSS_EXT.1.2

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

Findings:	[AGD] Section 2.2.10 in the guidance states: The following commands can be used to configure the TLS web-server profile: … *ssl-protocol tlsv1.2* …

FCS_TLSS_EXT.1.3

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

Findings:	[AGD] Section 2.2.10 in the guidance states: No configuration is required to set the permitted cipher suites or the associated key agreement parameters once 'fips enable' has been entered on the Conductor.

NIAP TD0569

FCS_TLSS_EXT.1.4

328 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.

5.1.5.3 Tests

FCS_TLSS_EXT.1.1

329 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.

330 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.

High-Level Test Description
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.

331 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

332 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

333 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 sent a Server Hello message and the connection is not established when the client sends an unsupported elliptic curve.

334 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

335 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

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).

336 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

337 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.

338 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 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 3a, and the TOE did not resume a session when an altered/invalid session ticket was presented as described in test 3b.

5.2 Identification and Authentication (FIA)

5.2.1 FIA_X509_EXT.1/Rev X.509 Certificate Validation

5.2.1.1 TSS

339 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 6.3.6 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) TLS Client certificates must have the Client Authentication purpose in the extendedKeyUsage field

iii) OCSP certificates must have the OCSP signing purpose in the extendedKeyUsage field

Certificate revocation checking is performed using OCSP.

340 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 6.3.6 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.

5.2.1.2 Guidance Documentation

341 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: [AGD] Section 2.3.6 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 does not identify any rules for extendedKeyUsage fields in FIA_X509_EXT.1.1 that are not supported by the TOE.

The TOE performs revocation checking on all certificates in the chain. The [AGD] section 2.3.6 states:

In the evaluated configuration, ArubaOS supports certificate revocation checking

using OCSP. OCSP is the recommended method of revocation checking.

Note, OCSP does not support multiple levels of certificate chaining for delegated trust, so the direct issuer of the OCSP responder's signing certificate must be configured in the RCP. If multiple levels of certificate checking will be performed (e.g. for a peer's IPsec certificate and one level up to an Intermediate CA) then a separate RCP must be configured for each, along with an appropriate OCSP responder certificate.
…
To configure delegated trust on the TOE for OCSP verification of each CA, ensure that CA certificates are uploaded as bundles. The following procedures should be followed:

1. Create a full CA bundle, from the leaf's issuing CA to the rootca.
2. Upload that as a trustedCA bundle.
3. Upload the same CA bundle as an OCSP responder cert.
4. Click on the RCP for the full CA bundle.
5. Ensure that the correct OCSP responder cert is selected.
6. Input the OCSP responder URL for the top most intermediary CA in the bundle.
7. For the next CA bundle, remove the top most intermediary CA and save it as a new bundle.
8. Repeat above steps until you're left with just the rootca.

5.2.1.3 Tests

342 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.

High-Level Test Description
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.
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 restarted 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 restarted the OCSP server. Verify that a certificate now results in a successful connection.

High-Level Test Description
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.

NIAP TD0527 (REVISED 1 December 2020)

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.

343 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.

344 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).

345 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.

346 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.

5.2.2 FIA_X509_EXT.2 X.509 Certificate Authentication

5.2.2.1 TSS

347 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:	Within [ST] section 6.3.7 the TSS states that the TOE has a trust store where root CA and intermediate CA certificates can be stored. The trust store is not cached: if a certificate is deleted, it is immediately untrusted. If a certificate is added to the trust store, it is immediately trusted for its given scope. Instructions for configuring the trusted IT entities to supply appropriate X.509 certificates are captured in the guidance documents.

	As part of the verification process, OCSP is used to determine whether the certificate is revoked or not.

348 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:	Within [ST] section 6.3.7 the TSS states that as part of the verification process, OCSP is used to determine whether the certificate is revoked or not. When a connection cannot be established to determine the validity of a certificate, the certificate is not accepted.

5.2.2.2 Guidance Documentation

349 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:	For to use the certificates, the operating environment requires an OCSP Responder and the OCSP Responder certificate must be loaded onto the TOE as per [AGD] section 2.3.6: For the purpose of verifying OCSP responses, ArubaOS requires that the responses be signed, and requires that the nonce extension be supported by the OCSP responder. Signed responses are verified using the "OCSP Responder" certificate. Two methods are supported: direct trust and delegated trust. For direct trust, the signing certificate of the OCSP responder must be loaded onto the Conductor through the WebUI Certificate Management section, and its name configured in the relevant RCP. When used, the Conductor makes a direct comparison between the signer certificate included in the OCSP response, and the OCSP responder certificate that was loaded - they must be exactly the same certificate. Direct Trust is cumbersome in environments where the OCSP responder certificate expires frequently. An alternative is Delegated Trust. In this method, the "OCSP Responder" type certificate must still be loaded into the Conductor, in the same way just described. However, the certificate should be the Issuing CA certificate for the CA that issues a signing certificate to the OCSP responder. When this type of configuration is performed, ArubaOS will examine the certificate in the OCSP response, then chain one level up to see if that certificate was issued by the CA configured in the RCP. For use with the IPsec trusted channel [AGD] section 2.2.7.9 states: Loading of certificates onto the Conductor for both authentication to peers and for verification of other peers is described in the (“Managing Certificates” section of the ArubaOS User Guide). Minimally, both a "server certificate" and a "trusted root CA" certificate must be loaded onto the Conductor in order to perform IPsec operations. Once these certificates are loaded on the Conductor, configure them for use in IPsec. For use with dynamic VPN clients: *(config) #crypto-local isakmp server-certificate "server-cert"* *(config) #crypto-local isakmp ca-certificate "trusted-root-ca-cert"*

	For a site-to-site VPN tunnel: *(config) #crypto-local ipsec-map 10.10.20.1 100* *(config-ipsec-map)# set server-certificate server-cert* *(config-ipsec-map)# set ca-certificate root-ca* To configure an IKE policy to authenticate RSA certificates sent by peers, use the following command: *(config) #crypto isakmp policy 100* *(config-isakmp)# authentication rsa-sig* To configure an IKE policy for ECDSA-384 authentication, use the following command: *(config) #crypto isakmp policy 100* *(config-isakmp)# authentication ecdsa-384* ECDSA-256 may be supported by replacing "384" with "256".

5.2.2.3 Tests

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

351 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.

5.2.3 FIA_X509_EXT.3 Extended: X509 Certificate Requests

5.2.3.1 TSS

352 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:	The ST author has not selected “device-specific information.” Within [ST] section 6.3.8 the TSS states that the TOE generates Certificate Request Messages and includes the following information: public key, common name, organization, organizational unit, country. Upon receiving the CA Certificate response, the TOE will validate the chain of certificates from the Root CA.

5.2.3.2 Guidance Documentation

353 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:	[AGD] Section 2.3.7 of the guidance state: An example of the commands that can be used to generate a certificate sign request are provided below: *crypto pki* *csr rsa* *key_len 2048* *common_name <common_val>* *country <country>* *organization <org>* *unit <org_unit>* To export the request, you may show the CSR with the follow command: *Show crypto pki csr* Before creating a CSR, the administrator must ensure that the CN, country, O, and OU have been set as identified above.

5.2.3.3 Tests

354 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.

5.3 Security management (FMT)

5.3.1 FMT_MOF.1/Functions Management of security functions behaviour

5.3.1.1 TSS

355 For distributed TOEs see chapter 2.4.1.1.

Findings:	TOE is not distributed.

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

Findings:	Within [ST] section 6.4.2 the TSS states that the TOE restricts the ability to modify (enable/disable) transmission of audit records to an external audit server to Security Administrators. [ST] Section 6.1.3 of the TSS states The Security Administrator can configure the TOE to send logs to a Syslog server. Log events are sent in real-time. Logs are sent via IPsec.

5.3.1.2 Guidance Documentation

357 For distributed TOEs see chapter 2.4.1.2.

Findings:	TOE is not distributed.

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

Findings:	[AGD] Section 2.4.2 of the guidance states: An administrator with the management role of “root” has full privileges to modify, add, and delete configuration settings on the TOE. The “root” role maps to the Security Administrator role. [AGD] section 2.1.3 describes how the Security Administrator modifies the behaviour of transmitting audit data to an external IT entity:

To configure an external syslog server:

(config)# logging <ip address>

The connection between the Mobility Conductor and the syslog server must be protected using IPsec. Configure a site-to-site VPN tunnel to carry this traffic. The syslog server must use a different IP address for the syslog receiver process than it uses for IPsec termination. Alternatively, a VPN gateway (such as an Aruba Mobility Controller) may front-end the syslog server to provide the IPsec tunnel. The following is an example of an IPsec tunnel which assumes that the syslog receiver process listens on 192.168.1.1, and the IPsec tunnel terminates on 192.168.2.1 – these IP addresses may be on the same server, or on different systems.

```
crypto-local ipsec-map <name> 10
  version v2
  set ikev2-policy <policy>
  peer-ip <ip address>
  src-net <ip address> <subnet>
  dst-net <ip address> <subnet>
  set transform-set "<transform-set>"
  set security-association lifetime seconds <seconds>
  set security-association lifetime kilobytes <kilobytes>
  pre-connect enable
  trusted enable
  uplink-failover disable
  force-natt disable
  set ca-certificate root-ca
  set server-certificate server-cert
```

Adjust the above ipsec-map as appropriate, following instructions in the ArubaOS User Guide. The peer-ip and dst-net addresses cannot be the same. Note that bi-directional communication is not necessary – syslog is sent using UDP, so the only requirement is that packets are able to flow from the Mobility Conductor to the syslog server.

5.3.1.3 Tests

359 Test 1 (if 'transmission of audit data to external IT entity' is selected from the second selection together with 'modify the behaviour of' in the first selection): The evaluator shall try to modify all security related parameters for configuration of the transmission protocol for transmission of audit data to an external IT entity without prior authentication as Security Administrator (by authentication as a user with no administrator privileges or without user authentication at all). Attempts to modify parameters without prior authentication should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt to modify the security related parameters can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
Using an unprivileged 'readonly' user, attempt to disable the remote syslog logging mechanism and show the attempt is unsuccessful.
Findings: PASS – The evaluator confirmed the unprivileged user is unable to modify the configuration of the remote logging mechanism.

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

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

High-Level Test Description
Using a privileged user, attempt to disable and re-enable the remote syslog logging mechanism and show that the attempt is successful.
Findings: PASS – The evaluator confirmed the privileged administrator has the authorization to successfully modify the configuration of the remote logging mechanism.

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

High-Level Test Description
The TOE does not claim ‘handling of audit data’ functionality.
Findings: N/A

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

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

High-Level Test Description
The TOE does not claim ‘handling of audit data’ functionality.
Findings: N/A

365 Test 1 (if 'audit functionality when Local Audit Storage Space is full' is selected from the second selection together with 'modify the behaviour of' in the first selection): The

evaluator shall try to modify the behaviour when Local Audit Storage Space is full without prior authentication as Security Administrator (by authentication as_a user with no administrator privileges or without user authentication at all). This attempt should fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
The TOE does not claim ‘audit functionality when Local Audit Storage Space is full’ functionality.
Findings: N/A

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

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

High-Level Test Description
The TOE does not claim ‘audit functionality when Local Audit Storage Space is full’ functionality.
Findings: N/A

368 Test 3 (if in the first selection 'determine the behaviour of' has been chosen together with for any of the options in the second selection): The evaluator shall try to determine the behaviour of all options chosen from the second selection without prior authentication as Security Administrator (by authentication as a user with no administrator privileges or without user authentication at all). This can be done in one test or in separate tests. The attempt(s) to determine the behaviour of the selected functions without administrator authentication shall fail. According to the implementation no other users than the Security Administrator might be defined and without any user authentication the user might not be able to get to the point where the attempt can be executed. In that case it shall be demonstrated that access control mechanisms prevent execution up to the step that can be reached without authentication as Security Administrator.

High-Level Test Description
The TOE does not claim ‘determine the behaviour of’ functionality.
Findings: N/A

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

High-Level Test Description
The TOE does not claim ‘determine the behaviour of’ functionality.
Findings: N/A

5.3.2 FMT_MTD.1/CryptoKeys Management of TSF Data

5.3.2.1 TSS

370 For distributed TOEs see chapter 2.4.1.1.

Findings:	TOE is not distributed.

371 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:	Within [ST] section 6.4.5 the TSS states that the TOE restricts the ability to manage SSH, TLS, IPsec and any configured X.509 private keys to Security Administrators.

5.3.2.2 Guidance Documentation

372 For distributed TOEs see chapter 2.4.1.2.

Findings:	TOE is not distributed.

373 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:	[AGD] Section 2.4.4 of the guidance states: An administrator with the management role of “root” has full privileges to modify, add, and delete configuration and user accounts. The “root” role maps to the Security Administrator role. [AGD] Section 2.3.6 of the guidance describes that the TOE is able to generate, import, modify and delete RSA and ECDSA public and private keys as part of X.509 certificates used both for IPsec communication and TLS communications. [AGD] section 2.2.7.9 provides instructions on using the X.509 certificates in a IPsec channel for both authentication of the TOE to IPsec peers and to authenticate IPsec peers to the TOE. Section 2.2.9 describes that the TOE is able to generate RSA Host keys and import SSH client public keys for SSH authentication. Full instructions on how to upload a X.509-containerized public key for SSH authentication are included in the ArubaOS 8.10.0.0 User Guide. The host key is generated at install time. To regenerate an SSH host key, the administrator must gain support access to the shell and manually do so, or zeroize the appliance.

Please see relevant sections for instructions on how these operations are performed.

5.3.2.3 Tests

374 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.

375 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.

6 Evaluation Activities for Security Assurance Requirements

6.1 ASE: Security Target

6.1.1 General ASE

376 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.

377 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:	N/A, the TOE is not a distributed TOE.

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

6.2 ADV: Development

6.2.1 Basic Functional Specification (ADV_FSP.1)

379 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.

380 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.

381 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.

382 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.

6.2.1.1 Evaluation Activity:

383 *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.*

384 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.

385 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 [SD] cannot be completed because the [ST] or the AGD is incomplete, then the functional specification is not complete and observations are required. During the evaluator's use of the product and its interfaces (the Web GUI, SSH CLI, local serial port), there were no areas that were deficient.

6.2.1.2 Evaluation Activity

386 *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.

6.2.1.3 Evaluation Activity:

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

388 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.

389 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.

390 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.

6.3 AGD: Guidance Documents

391 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.

392 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)

6.3.1 Operational User Guidance (AGD_OPE.1)

393 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.

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

6.3.1.1 Evaluation Activity:

395 *The evaluator shall ensure the Operational guidance documentation is distributed to Security Administrators and users (as appropriate) as part of the TOE, so that there is a reasonable guarantee that Security Administrators and users are aware of the existence and role of the documentation in establishing and maintaining the evaluated configuration.*

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

6.3.1.2 Evaluation Activity

396 *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:	There is only one operational environment claimed in the [ST]. All TOE platforms claimed in [ST] are covered by the operational guidance. This is evidenced by the platform equivalency.

6.3.1.3 Evaluation Activity

397 *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 [AGD] states: Ensure the controller has FIPS mode enabled so that cryptographic requirements are met. *(config)# fips enable*

6.3.1.4 Evaluation Activity

398 *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:	The [AGD] document covers configuration of the in-scope functionality where additional configuration might be required.

6.3.1.5 Evaluation Activity

399 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 [PP] 5.3.1.3 for configuration of the cryptographic engine. See work unit in section 3.4.1.2 for FMT_MOF.1/ManualUpdate for instructions on verifying updates to the TOE. See work unit [PP] 5.3.1.4 for details as to what was covered by the EAs.

6.3.2 Preparative Procedures (AGD_PRE.1)

400 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.

401 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.

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

6.3.2.1 Evaluation Activity

403 *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).*

404 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:	[AGD] Section 1.5 – 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.

6.3.2.2 Evaluation Activity

405 *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:	There is only one operational environment claimed in the [ST]. All TOE platforms claimed in [ST] are covered by the preparative procedures in the operational guidance. This is evidenced by the platform equivalency.

6.3.2.3 Evaluation Activity

406 *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.

6.3.2.4 Evaluation Activity

407 *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 helps instil a culture of secure manageability within a larger operational environment.

6.3.2.5 Evaluation Activity

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

409 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:	[AGD] section 1.5 “Preparatory Guidance” includes a description of client requirements to connect with protected administrative interfaces. Sections 2.2.5, 2.2.9 and 2.2.10 provides instructions for configuring the TOE to support HTTPS/TLS and SSH administrative interfaces. The [AGD] section 2.3.1 only identifies the default password recovery account as containing a default value. The section further describes that it must be disabled in the evaluated configuration.

6.4 ALC: Life-cycle Support

6.4.1 Labelling of the TOE (ALC_CMC.1)

410 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.

411

6.4.2 TOE CM coverage (ALC_CMS.1)

412 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.

6.5 ATE: Tests

6.5.1 Independent Testing – Conformance (ATE_IND.1)

413 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.

414 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.

415 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.

416 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 not a distributed TOE.

6.6 Vulnerability Assessment

6.6.1 Vulnerability Survey (AVA_VAN.1)

417 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.

418 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.

419 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.

6.6.1.1 Evaluation Activity (Documentation):

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

421 *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

422 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).

423 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.

6.6.1.2 Evaluation Activity:

424 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 June 16, 2023 and included the following search terms:

- ArubaOS 8.10;

- Aruba Mobility Conductor;

- Aruba OpenSSL Module;

- Aruba Crypto Module;

- Aruba Bootloader Module;

- MCR-HW-1K-F1

- MCR-HW-5K-F1;

- MCR-HW-10K-F1;

- MM-HW-1K-F1;

- MM-HW-5K-F1;

- MM-HW-10K-F1;

- Intel Xeon E5-2609v4;

- Intel Xeon E5-2620v4;

- Intel Xeon E5-2650v4;

- 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.