Aruba Mobility Conductor with ArubaOS 8.10

Security Target

Version 1.2

June 2023

Document prepared by

www.lightshipsec.com

Document History

Version	Date	Author	Description
1.0	23 May 2023	Garrett Nickel	Release for certification
1.1	02 June 2023	Garrett Nickel	Guidance Document version references, QA updates.
1.2	19 June 2023	Garrett Nickel	Address ECR Comments

Table of Contents

List of Tables

1 Introduction

1.1 Overview

1 This Security Target (ST) defines the Aruba Mobility Conductor with ArubaOS 8.10 Target of Evaluation (TOE) for the purposes of Common Criteria (CC) evaluation.

2 The Aruba Mobility Conductor simplifies the management of multiple Aruba controllers running ArubaOS 8 or later. Key features include a centralized dashboard to easily see and manage controllers deployed in multiple sites, a hierarchical configuration tool to pre-stage network deployments, and the ability to perform live firmware and feature upgrades during active user sessions. The addition of licensing pools simplifies the transfer of licenses between different controllers to quickly address expanded deployment needs.

1.2 Identification

Table 1: Evaluation identifiers

Target of Evaluation	Aruba Mobility Conductor with ArubaOS 8.10 Evaluated Build: 8.10.0.6-FIPS-CC_BUILD
Security Target	Aruba Mobility Conductor with ArubaOS 8.10 Security Target, v1.2

1.3 Conformance Claims

3 This ST supports the following conformance claims:

a) CC version 3.1 revision 5

b) CC Part 2 extended

c) CC Part 3 conformant

d) collaborative Protection Profile for Network Devices, v2.2e (referenced within as NDcPP)

e) NIAP Technical Decisions per Table 2

Table 2: NIAP Technical Decisions

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	

TD #	Name	Exclusion Rationale
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	

TD #	Name	Exclusion Rationale
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	

1.4 Terminology

Table 3: Terminology

Term	Definition
AP	Access Point (wireless)
ArubaOS	The ArubaOS is a suite of applications that runs on Mobility Conductor and Mobility Controllers that allows administrators to configure and manage the wireless and mobile user environment.
Aruba Mobility Conductor	The Aruba Mobility Conductor allows for centralized management of multiple Aruba controllers.
Aruba Mobility Controller	Aruba Mobility Controllers serve as a gateway between wired and wireless networks and provides command and control over Aruba APs within an Aruba dependent wireless network.
CC	Common Criteria
GUI	Graphical User Interface
KAT	Known Answer Test
NDcPP	collaborative Protection Profile for Network Devices
PP	Protection Profile

Term	Definition
TOE	Target of Evaluation
TSF	TOE Security Functionality

2 TOE Description

2.1 Type

4 The TOE is a network device that provides centralized management of multiple Aruba Mobility Controllers.

2.2 Usage

2.2.1 Deployment

5 The TOE is deployed within a network that provides connectivity to the managed Aruba controllers. The TOE's web-based GUI is the primary user interface, as shown in Figure 1.

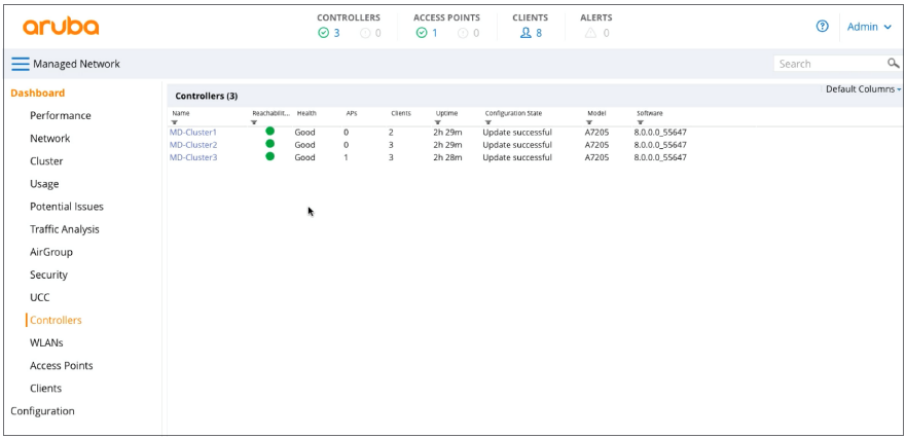

Figure 1: TOE User Interface

2.2.2 Interfaces

6 The TOE interfaces within the scope of evaluation are shown in Figure 2.

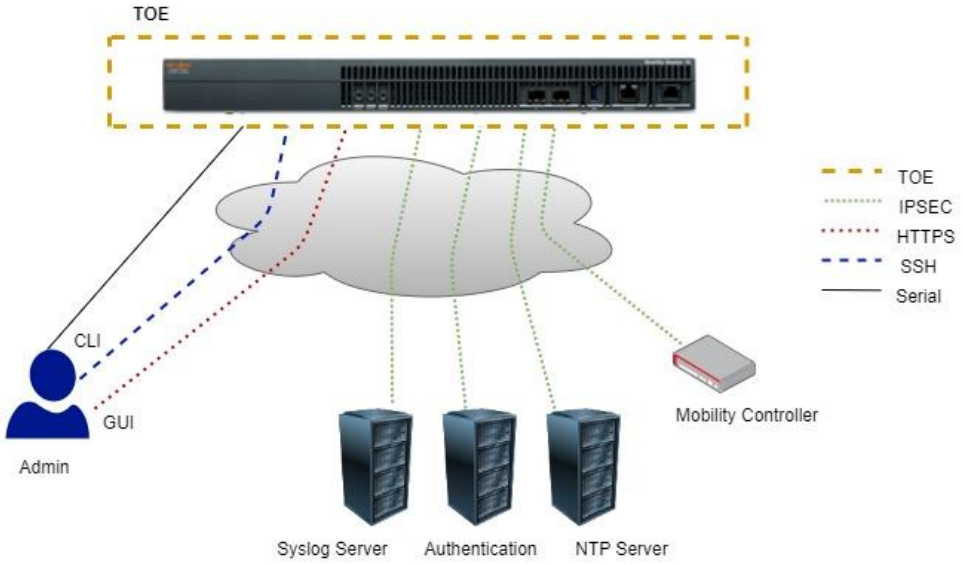

Figure 2: TOE interfaces

7 The TOE interfaces are as follows:

a) **CLI.** Administrative CLI via direct serial connection or SSH.

b) **GUI.** Administrative web GUI via HTTPS / TLS.

c) **RADIUS/TACACS+.** Authentication with a remote server via IPsec.

d) **Logs.** Logs are sent to an external syslog server via IPsec.

e) **NTP.** Time synchronization with an NTP server via IPsec.

f) **Mobility Controller.** Management of Aruba Mobility Controllers via IPsec.

2.3 Security Functions / Logical Scope

8 The TOE provides the following security functions:

a) **Protected Communications.** The TOE protects the integrity and confidentiality of communications as noted in section 2.2 above.

b) **Secure Administration.** The TOE enables secure management of its security functions, including:

   i) Administrator authentication with passwords

   ii) Configurable password policies

   iii) Role Based Access Control

   iv) Access banners

   v) Management of critical security functions and data

   vi) Protection of cryptographic keys and passwords

c) **Trusted Update.** The TOE ensures the authenticity and integrity of software updates through digital signatures.

d) **System Monitoring.** The TOE generates logs of security relevant events. The TOE stores logs locally and is capable of sending log events to a remote audit server.

e) **Self-Test.** The TOE performs a suite of self-tests to ensure the correct operation and enforcement of its security functions.

f) **Cryptographic Operations.** The TOE implements a cryptographic module. Relevant Cryptographic Algorithm Validation Program (CAVP) certificates are shown in Table 4 and Table 5.

Table 4: CAVP Certificates

Module Name	Services	Certificates
ArubaOS OpenSSL Module	Performs cryptographic functions to support all SSH, HTTPS, and TLS operations.	A2690
ArubaOS Crypto Module	Provides cryptographic functions to support all IPsec/IKE session operations	A2689
ArubaOS Bootloader Module	Provides cryptographic functions to support firmware integrity testing	A2688

2.4 Physical Scope

9 The physical boundary of the TOE includes the appliance models shown in Table 5 executing ArubaOS 8.10 software. The TOE hardware is shipped to users via commercial courier and the TOE software is available via Aruba customer portal.

10 The ArubaOS consists of a base software package with add-on software modules that can be activated by installing the appropriate licenses. The following modules are required to be licensed and installed in the CC evaluated configuration:

a) **Advanced Cryptography.** Required for Commercial National Security Algorithms Suite, AES-GCM and ECDSA functionality.

Table 5: 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)		

2.4.1 Guidance Documents

11 The evaluation includes the following guidance documents (PDF):

a) ArubaOS 8.10.0.0 User Guide Revision 02

b) ArubaOS 8.10.0.0 Getting Started Guide, Revision 01

c) ArubaOS 8.x Command-Line Reference Guide

d) ArubaOS 8.10.0.0 Syslog Reference Guide

e) Aruba OS 8.10 Supplemental Guidance (Common Criteria Configuration Guidance for Aruba Mobility Conductor with ArubaOS 8.10-FIPS) Version 2.6.

2.4.2 Non-TOE Components

12 The TOE operates with the following components in the environment:

a) **Audit Server.** The TOE is capable of sending audit events to a Syslog server.

b) **NTP Server.** The TOE synchronizes time with an NTP server.

c) **Authentication Server.** The TOE is able to utilize RADIUS and TACACs+ servers to authenticate users.

d) **Mobility Controllers.** The TOE manages Aruba controllers running ArubaOS 8 or later. Note: The TOE security functions are not reliant on the presence of Mobility Controllers.

2.5 Logical Scope

13 The logical scope of the TOE comprises the security functions defined in section 2.3.

2.5.1 Functions not included in the TOE Evaluation

14 The TOE is capable of a variety of functions which are not covered by the NDcPP. Only the security functions defined in section 2.3 have been examined as part of this evaluation.

15 IKEv1 must not be used in the evaluated configuration.

3 Security Problem Definition

16 The Security Problem Definition is reproduced from section 4 of the NDcPP. Non-applicable Security Problem Definitions have been removed.

3.1 Threats

Table 6: Threats

Identifier	Description
T.UNAUTHORIZED_ ADMINISTRATOR_ ACCESS	Threat agents may attempt to gain Administrator access to the Network Device by nefarious means such as masquerading as an Administrator to the device, masquerading as the device to an Administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between Network Devices. Successfully gaining Administrator access allows malicious actions that compromise the security functionality of the device and the network on which it resides.
T.WEAK_ CRYPTOGRAPHY	Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.
T.UNTRUSTED_ COMMUNICATION_ CHANNELS	Threat agents may attempt to target Network Devices that do not use standardized secure tunnelling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the Network Device itself.
T.WEAK_ AUTHENTICATION_ ENDPOINTS	Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints – e.g. a shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the Administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the Network Device itself could be compromised.
T.UPDATE_ COMPROMISE	Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.
T.UNDETECTED_ ACTIVITY	Threat agents may attempt to access, change, and/or modify the security functionality of the Network Device without Administrator awareness. This could result in the attacker finding an avenue (e.g.,

Identifier	Description
	misconfiguration, flaw in the product) to compromise the device and the Administrator would have no knowledge that the device has been compromised.
T.SECURITY_ FUNCTIONALITY_ COMPROMISE	Threat agents may compromise credentials and device data enabling continued access to the Network Device and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or device credentials for use by the attacker.
T.PASSWORD_ CRACKING	Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the device. Having privileged access to the device provides the attacker unfettered access to the network traffic, and may allow them to take advantage of any trust relationships with other Network Devices.
T.SECURITY_ FUNCTIONALITY_ FAILURE	An external, unauthorized entity could make use of failed or compromised security functionality and might therefore subsequently use or abuse security functions without prior authentication to access, change or modify device data, critical network traffic or security functionality of the device.

3.2 Assumptions

Table 7: Assumptions

Identifier	Description
A.PHYSICAL_ PROTECTION	The Network Device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, the cPP does not include any requirements on physical tamper protection or other physical attack mitigations. The cPP does not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device. For vNDs, this assumption applies to the physical platform on which the VM runs.
A.LIMITED_ FUNCTIONALITY	The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the device should not provide a computing platform for general purpose applications (unrelated to networking functionality). If a virtual TOE evaluated as a pND, following Case 2 vNDs as specified in Section 1.2, the VS is considered part of the TOE with only one vND instance for each physical hardware platform. The exception being where components of a distributed TOE run inside more than one virtual machine (VM) on a single VS. In Case 2 vND, no non-TOE guest VMs are allowed on the platform.

Identifier	Description
A.NO_THRU_ TRAFFIC_ PROTECTION	A standard/generic Network Device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the Network Device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the Network Device, destined for another network entity, is not covered by the NDcPP. It is assumed that this protection will be covered by cPPs and PP-Modules for particular types of Network Devices (e.g., firewall).
A.TRUSTED_ ADMINISTRATOR	The Security Administrator(s) for the Network Device are assumed to be trusted and to act in the best interest of security for the organization. This includes appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The Network Device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are expected to fully validate (e.g. offline verification) any CA certificate (root CA certificate or intermediate CA certificate) loaded into the TOE's trust store (aka 'root store', 'trusted CA Key Store', or similar) as a trust anchor prior to use (e.g. offline verification).
A.REGULAR_ UPDATES	The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.ADMIN_ CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.
A.RESIDUAL_ INFORMATION	The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

3.3 Organizational Security Policies

Table 8: Organizational Security Policies

Identifier	Description
P.ACCESS_BANNER	The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.

4 Security Objectives

17 The security objectives are reproduced from section 5 of the NDcPP.

Table 9: Security Objectives for the Operational Environment

Identifier	Description
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_ PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE. Note: For vNDs the TOE includes only the contents of the its own VM, and does not include other VMs or the VS.
OE.NO_THRU_ TRAFFIC_ PROTECTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.
OE.TRUSTED_ADMIN	Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner. For vNDs, this includes the VS Administrator responsible for configuring the VMs that implement ND functionality. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE's trust store in case such certificate can no longer be trusted.
OE.UPDATES	The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_ CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.
OE.RESIDUAL_ INFORMATION	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment. For vNDs, this applies when the physical platform on which the VM runs is removed from its operational environment.

5 Security Requirements

5.1 Conventions

18 This document uses the following font conventions to identify the operations defined by the CC:

a) **Assignment.** Indicated with italicized text.

b) **Refinement.** Indicated with bold text and strikethroughs.

c) **Selection.** Indicated with underlined text.

d) **Assignment within a Selection:** Indicated with italicized and underlined text.

e) **Iteration.** Indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").

19 **Note:** Selection and assignment operations performed within the Security Target are denoted within brackets []. Operations shown without brackets are reproduced from the NDcPP.

5.2 Extended Components Definition

20 Refer to Annex A: Extended Components Definition.

5.3 Functional Requirements

Table 10: Summary of SFRs

Requirement	Title
FAU_GEN.1	Audit Data Generation
FAU_GEN.2	User Identity Association
FAU_STG_EXT.1	Protected Audit Event Storage
FCS_CKM.1	Cryptographic Key Generation
FCS_CKM.2	Cryptographic Key Establishment
FCS_CKM.4	Cryptographic Key Destruction
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)
FCS_COP.1/SigGen	Cryptographic Operation (Signature Generation and Verification)
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)
FCS_HTTPS_EXT.1	HTTPS Protocol
FCS_IPSEC_EXT.1	IPsec Protocol

Requirement	Title
FCS_NTP_EXT.1	NTP Protocol
FCS_RBG_EXT.1	Random Bit Generation
FCS_SSHS_EXT.1	SSH Server Protocol
FCS_TLSS_EXT.1	TLS Server Protocol
FIA_AFL.1	Authentication Failure Management
FIA_PMG_EXT.1	Password Management
FIA_UIA_EXT.1	User Identification and Authentication
FIA_UAU_EXT.2	Password-based Authentication Mechanism
FIA_UAU.7	Protected Authentication Feedback
FIA_X509_EXT.1/Rev	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FIA_X509_EXT.3	X.509 Certificate Requests
FMT_MOF.1/ManualUpdate	Management of Security Functions Behavior
FMT_MOF.1/Functions	Management of Security Functions Behavior
FMT_MTD.1/CoreData	Management of TSF Data
FMT_MTD.1/CryptoKeys	Management of TSF Data
FMT_SMF.1	Specification of Management Functions
FMT_SMR.2	Restrictions on Security Roles
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
FPT_APW_EXT.1	Protection of Administrator Passwords
FPT_TST_EXT.1	TSF Testing
FPT_TUD_EXT.1	Extended: Trusted update
FPT_STM_EXT.1	Reliable Time Stamps
FTA_SSL_EXT.1	TSF-initiated Session Locking
FTA_SSL.3	TSF-initiated Termination

Requirement	Title
FTA_SSL.4	User-initiated Termination
FTA_TAB.1	Default TOE Access Banners
FTP_ITC.1	Inter-TSF trusted channel
FTP_TRP.1/Admin	Trusted Path

5.3.1 Security Audit (FAU)

FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1 The TSF shall be able to generate an audit record of the following auditable events:

a) Start-up and shutdown of the audit functions;

b) All auditable events for the not specified level of audit;

c) All administrative actions comprising:

- *Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).*
- *Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).*
- *Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).*
- *Resetting passwords (name of related user account shall be logged).*
- *[no other actions];*

d) *Specifically defined auditable events listed in **Table 11.***

Table 11: Audit Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG_EXT.1	None.	None.
FCS_CKM.1	None.	None.
FCS_CKM.2	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure
FCS_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure
FCS_NTP_EXT.1	Configuration of a new time server Removal of configured time server	Identity if new/removed time server
FCS_RBG_EXT.1	None.	None.
FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate. Any addition, replacement or removal of trust anchors in the TOE's trust store.	Reason for failure of certificate validation. Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store.

Requirement	Auditable Events	Additional Audit Record Contents
FIA_X509_EXT.2	None.	None.
FIA_X509_EXT.3	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MOF.1/Functions	None.	None.
FMT_MTD.1/CoreData	None.	None.
FMT_MTD.1/CryptoKeys	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1 (if “terminate the session” is selected)	The termination of a local session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FTP_ITC.1	Initiation of the trusted channel. Termination of the trusted channel. Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FTP_TRP.1/Admin	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	None.

FAU_GEN.1.2 The TSF shall record within each audit record at least the following information:

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the cPP/ST, *information specified in column three of* ***Table 11.***

FAU_GEN.2 User Identity Association

FAU_GEN.2.1 For audit events resulting from actions of identified users, the TSF shall be able to associate each auditable event with the identity of the user that caused the event.

FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1.

FAU_STG_EXT.1.2 The TSF shall be able to store generated audit data on the TOE itself. In addition [

- TOE shall consist of a single standalone component that stores audit data locally]

FAU_STG_EXT.1.3 The TSF shall [overwrite previous audit records according to the following rule: [*overwrite oldest record first*], [*no other action*]] when the local storage space for audit data is full.

5.3.2 Cryptographic Support (FCS)

FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1 The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm: [

- RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3;
- ECC schemes using “NIST curves” [P-256, P-384] that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4;
- FFC Schemes using ‘safe-prime’ groups that meet the following: “NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and [RFC 3526].

]~~and specified cryptographic key sizes [assignment: cryptographic key sizes] that meet the following: [assignment: list of standards]~~.

FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1 The TSF shall **perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [

- Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;
- FFC Schemes using “safe-prime” groups that meet the following: ‘NIST Special Publication 800-56A Revision 3, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography” and [groups listed in RFC 3526].

] ~~that meets the following: [assignment: list of standards].~~

Application note: This SFR was changed by TD0580 and TD0581.

FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [

- *For plaintext keys in volatile storage, the destruction shall be executed by a [single overwrite consisting of [zeroes]];*
- *For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that [*
  - *logically addresses the storage location of the key and performs a [single overwrite consisting of [zeroes]];*

] that meets the following: *No Standard*.

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

FCS_COP.1.1/DataEncryption The TSF shall perform *encryption/decryption* in accordance with a specified cryptographic algorithm *AES used in [CBC, CTR, GCM] mode* and cryptographic key sizes *[128 bits, 256 bits]* that meet the following: *AES as specified in ISO 18033-3, [CBC as specified in ISO 10116, CTR as specified in ISO 10116, GCM as specified in ISO 19772*].

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

FCS_COP.1.1/SigGen The TSF shall perform *cryptographic signature services (generation and verification)* in accordance with a specified cryptographic algorithm [

- RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [*2048 bits*],
- Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [*256 bits, 384 bits*]

] that meet the following: [

- For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,
- For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [P-256, P-384]; ISO/IEC 14888-3, Section 6.4].

FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)

FCS_COP.1.1/Hash The TSF shall perform *cryptographic hashing services* in accordance with a specified cryptographic algorithm [SHA-1, SHA-256, SHA-384, SHA-512] and **message digest sizes [160, 256, 384, 512] bits** that meet the following: *ISO/IEC 10118-3:2004*.

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

FCS_COP.1.1/KeyedHash The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm [HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384] and cryptographic key sizes *[160, 256, 384]* **and message digest sizes [160, 256, 384] bits** that meet the following: *ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*.

FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 If a peer certificate is presented, the TSF shall [not establish the connection] if the peer certificate is deemed invalid.

FCS_IPSEC_EXT.1 IPsec Protocol

FCS_IPSEC_EXT.1.1 The TSF shall implement the IPsec architecture as specified in RFC 4301.

FCS_IPSEC_EXT.1.2 The TSF shall have a nominal, final entry in the SPD that matches anything that is otherwise unmatched, and discards it.

FCS_IPSEC_EXT.1.3 The TSF shall implement [tunnel mode].

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms [AES-CBC-128 (RFC 3602), AES-CBC-256 (RFC 3602), AES-GCM-128 (RFC 4106), AES-GCM-256 (RFC 4106)] together with a Secure Hash Algorithm (SHA)-based HMAC [HMAC-SHA-1].

FCS_IPSEC_EXT.1.5 The TSF shall implement the protocol: [

- IKEv2 as defined in RFC 5996 and [with mandatory support for NAT traversal as specified in RFC 5996, section 2.23)], and [RFC 4868 for hash functions]

].

FCS_IPSEC_EXT.1.6 The TSF shall ensure the encrypted payload in the [IKEv2] protocol uses the cryptographic algorithms [AES-CBC-128, AES-CBC-256 (specified in RFC 3602].

FCS_IPSEC_EXT.1.7 The TSF shall ensure that [

- IKEv2 SA lifetimes can be configured by a Security Administrator based on [
    - length of time, where the time values can be configured within [*1-24*] hours

    ]

].

FCS_IPSEC_EXT.1.8 The TSF shall ensure that [

- IKEv2 Child SA lifetimes can be configured by a Security Administrator based on [
    - number of bytes;
    - length of time, where the time values can be configured within [*1-8*] hours;

    ]

].

FCS_IPSEC_EXT.1.9 The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange (“x” in g^x mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least [*160, 256, 384*] bits.

FCS_IPSEC_EXT.1.10 The TSF shall generate nonces used in [IKEv2] exchanges of length [

- according to the security strength associated with the negotiated Diffie-Hellman group;
- 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 The TSF shall ensure that IKE protocols implement DH Group(s) [

- [14 (2048-bit MODP)] according to RFC 3526,
- [19 (256-bit Random ECP), 20 (384-bit Random ECP)] according to RFC 5114.

].

FCS_IPSEC_EXT.1.12 The TSF shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [IKEv2 IKE_SA] connection is greater than or equal to the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [IKEv2 CHILD_SA] connection.

FCS_IPSEC_EXT.1.13 The TSF shall ensure that all IKE protocols perform peer authentication using [RSA, ECDSA] that use X.509v3 certificates that conform to RFC 4945 and [Pre-shared Keys].

FCS_IPSEC_EXT.1.14 The TSF shall only establish a trusted channel if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following fields and types: [Distinguished Name (DN)] and [no other reference identifier type].

FCS_NTP_EXT.1 NTP Protocol

FCS_NTP_EXT.1.1 The TSF shall use only the following NTP version(s) [NTP v4 (RFC 5905)].

FCS_NTP_EXT.1.2 The TSF shall update its system time using [

- [IPsec] to provide trusted communication between itself and an NTP time source.

].

FCS_NTP_EXT.1.3 The TSF shall not update NTP timestamp from broadcast and/or multicast addresses.

FCS_NTP_EXT.1.4 The TSF shall support configuration of at least three (3) NTP time sources.

FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [CTR_DRBG (AES)].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [[*One*] software based noise source] with a minimum of [256 bits] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for Hash Functions”, of the keys and hashes that it will generate.

FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1 The TSF shall implement the SSH protocol in accordance with RFC(s) 4251, 4252, 4253, 4254, [4344, 5656, 6668, 8308 section 3.1, 8332].

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following user authentication methods as described in RFC 4252: public key-based, [password based].

FCS_SSHS_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*256k*] bytes in an SSH transport connection are dropped.

FCS_SSHS_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: [aes128-cbc, aes256-cbc, aes128-ctr, aes256-ctr].

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [ssh-rsa, rsa-sha2-256, rsa-sha2-512] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [hmac-sha1, hmac-sha2-256] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7 The TSF shall ensure that [ecdh-sha2-nistp256] and [ecdh-sha2-nistp384] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8 The TSF shall ensure that within SSH connections, the same session keys are used for a threshold of no longer than one hour, and each encryption key is used to protect no more than one gigabyte of data. After any of the thresholds are reached, a rekey needs to be performed.

FCS_TLSS_EXT.1 TLS Server Protocol Without Mutual Authentication

FCS_TLSS_EXT.1.1 The TSF shall implement [TLS 1.2 (RFC 5246)] and reject all other TLS and SSL versions. The TLS implementation will support 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

].

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0 and [TLS 1.1].

FCS_TLSS_EXT.1.3 The TSF shall perform key establishment for TLS using [ECDHE curves [secp256r1] and no other curves]].

FCS_TLSS_EXT.1.4 The TSF shall support [session resumption based on session IDs according to RFC 4346 (TLS1.1) or RFC 5246 (TLS1.2), session resumption based on session tickets according to RFC 5077].

5.3.3 Identification and Authentication (FIA)

FIA_AFL.1 Authentication Failure Management

FIA_AFL.1.1 The TSF shall detect when an Administrator configurable positive integer within [*1-10*] unsuccessful authentication attempts occur related to *Administrators attempting to authenticate remotely using a password*.

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall [prevent the offending Administrator from successfully establishing remote session using any authentication method that involves a password until an Administrator defined time period has elapsed].

FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special

characters: [“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, “`”, “~”, “-“, “_”, “=”, “+”, “[“, “]”, “{“, “}”, “\”, “|”, “;”, “:”, “”, “”, “,”, “.”, “<”, “>”, “/”];

b) Minimum password length shall be configurable to between [*8*] and [*32] characters.*

FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1.1 The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process:

- Display the warning banner in accordance with FTA_TAB.1;
- [[*no other action*]]

FIA_UIA_EXT.1.2 The TSF shall require each administrative user to be successfully identified and authenticated before allowing any other TSF-mediated actions on behalf of that administrative user.

FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local [password-based] authentication mechanism to perform local administrative user authentication.

FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1 The TSF shall provide only *obscured feedback* to the administrative user while the authentication is in progress **at the local console**.

FIA_X509_EXT.1/Rev X.509 Certificate Validation

FIA_X509_EXT.1.1/Rev The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certification path validation **supporting a minimum path length of three certificates**.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the presence of the basicConstraints extension and that the CA flag is set to TRUE.
- The TSF shall validate the revocation status of the certificate using [the Online Certificate Status Protocol (OCSP) as specified in RFC 6960]
- The TSF shall validate the extendedKeyUsage field according to the following rules:
    - *Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.*

- *Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.*
- *Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.*
- *OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.*

FIA_X509_EXT.1.2/Rev The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [IPsec], and [no additional uses].

FIA_X509_EXT.2.2 When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [not accept the certificate].

FIA_X509_EXT.3 X.509 Certificate Requests

FIA_X509_EXT.3.1 The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [Common Name, Organization, Organizational Unit, Country].

FIA_X509_EXT.3.2 The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

5.3.4 Security Management (FMT)

FMT_MOF.1/ManualUpdate Management of security functions behaviour

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to enable the functions *to perform manual updates* to *Security Administrators*.

FMT_MOF.1/Functions Management of security functions behaviour

FMT_MOF.1.1/Functions The TSF shall restrict the ability to [modify the behaviour of] the functions [transmission of audit data to an external IT entity] to *Security Administrators*.

FMT_MTD.1/CoreData Management of TSF Data

FMT_MTD.1.1/CoreData The TSF shall restrict the ability to *manage* the *TSF data* to *Security Administrators*.

FMT_MTD.1/CryptoKeys Management of TSF data

FMT_MTD.1.1/CryptoKeys The TSF shall restrict the ability to *manage* the *cryptographic keys to Security Administrators*.

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1 The TSF shall be capable of performing the following management functions:

- *Ability to administer the TOE locally and remotely;*
- *Ability to configure the access banner;*
- *Ability to configure the session inactivity time before session termination or locking;*
- *Ability to update the TOE, and to verify the updates using [digital signature] capability prior to installing those updates;*
- *Ability to configure the authentication failure parameters for FIA_AFL.1;*
- [
    - Ability to modify the behaviour of the transmission of audit data to an external IT entity;
    - Ability to manage the cryptographic keys;
    - Ability to configure the cryptographic functionality;
    - Ability to configure the lifetime for IPsec SAs;
    - Ability to configure the reference identifier for the peer;
    - Ability to configure NTP;
    - Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;
    - Ability to import X.509v3 certificates to the TOE's trust store;
    - Ability to manage the trusted public keys database.]

Application Note: This SFR has been modified by TD0631.

FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

- *Security Administrator*.

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions

- *The Security Administrator role shall be able to administer the TOE locally;*
- *The Security Administrator role shall be able to administer the TOE remotely*

are satisfied.

5.3.5 Protection of the TSF (FPT)

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

FPT_SKP_EXT.1.1 The TSF shall prevent reading of all pre-shared keys, symmetric keys, and private keys.

FPT_APW_EXT.1 Protection of Administrator Passwords

FPT_APW_EXT.1.1 The TSF shall store administrative passwords in non-plaintext form.

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext administrative passwords.

FPT_TST_EXT.1 TSF testing

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [during initial start-up (on power on)] to demonstrate the correct operation of the TSF: [

- *Cryptographic known answer tests;*
- *Firmware integrity tests;*
- *Aruba hardware known answer tests;*
- *Conditional self-tests*].

FPT_TUD_EXT.1 Trusted update

FPT_TUD_EXT.1.1 The TSF shall provide *Security Administrators* the ability to query the currently executing version of the TOE firmware/software and [the most recently installed version of the TOE firmware/software].

FPT_TUD_EXT.1.2 The TSF shall provide *Security Administrators* the ability to manually initiate updates to TOE firmware/software and [no other update mechanism].

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [digital signature] prior to installing those updates.

FPT_STM_EXT.1 Reliable Time Stamps

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [synchronise time with an NTP server].

5.3.6 TOE Access (FTA)

FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [

- terminate the session]

after a Security Administrator-specified time period of inactivity.

FTA_SSL.3 TSF-initiated Termination

FTA_SSL.3.1 The TSF shall terminate **a remote** interactive session after a *Security Administrator-configurable time interval of session inactivity*.

FTA_SSL.4 User-initiated Termination

FTA_SSL.4.1 The TSF shall allow **Administrator**-initiated termination of the **Administrator's** own interactive session.

FTA_TAB.1 Default TOE Access Banners

FTA_TAB.1.1 Before establishing **an administrative user** session the TSF shall display **a Security Administrator-specified** advisory **notice and consent** warning message regarding use of the TOE.

5.3.7 Trusted path/channels (FTP)

FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1 The TSF shall **be capable of using [IPsec] to provide** a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server, [authentication server, [*NTP Server, Aruba Mobility Controllers]*]** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from **disclosure and detection of modification of the channel data**.

FTP_ITC.1.2 The TSF shall permit **the TSF or the authorized IT entities** to initiate communication via the trusted channel.

FTP_ITC.1.3 The TSF shall initiate communication via the trusted channel for [*audit, authentication, management of mobility controllers, time synchronization*].

FTP_TRP.1 /Admin Trusted Path

FTP_TRP.1.1/Admin The TSF shall **be capable of using [SSH, HTTPS] to** provide a communication path between itself and **authorized** remote **Administrators** that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from **disclosure and provides detection of modification of the channel data**.

FTP_TRP.1.2 /Admin The TSF shall permit remote **Administrators** to initiate communication via the trusted path.

FTP_TRP.1.3 /Admin The TSF shall require the use of the trusted path for initial *Administrator authentication and all remote administration actions*.

5.4 Assurance Requirements

21 The TOE security assurance requirements are summarized in Table 12.

Table 12: Assurance Requirements

Assurance Class	Assurance Components
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Stated security requirements (ASE_REQ.1)
	Security problem definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance Documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life Cycle Support (ALC)	Labelling of the TOE (ALC_CMC.1)
	TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing – conformance (ATE_IND.1)
Vulnerability Assessment (AVA)	Vulnerability survey (AVA_VAN.1)

22 In accordance with section 7.1 of the NDcPP, the following refinement is made to ASE:

a) **ASE_TSS.1.1C Refinement:** The TOE summary specification shall describe how the TOE meets each SFR. **In the case of entropy analysis, the TSS is used in conjunction with required supplementary information on Entropy.**

6 TOE Summary Specification

23 The following describes how the TOE fulfils each SFR included in section 5.3.

6.1 Security Audit

6.1.1 FAU_GEN.1

24 The TOE generates the audit records specified at Table 11.

25 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.

6.1.2 FAU_GEN.2

26 The TOE includes the user identity in audit events resulting from actions of identified users.

6.1.3 FAU_STG_EXT.1

27 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.

28 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).

29 When the local audit data store is full, the TOE will overwrite audit records starting with the oldest audit record.

30 Only authorized administrators may view audit records and no capability to modify the audit records is provided.

6.2 Cryptographic Support

31 The TOE includes the following FIPS 140-2 Level 2 certified cryptographic modules which provide supporting cryptographic functions: ArubaOS OpenSSL Module, ArubaOS Crypto Module, and ArubaOS Bootloader Module.

32 The ArubaOS OpenSSL Module is used for SSH/HTTPS/TLS cryptographic functions and the ArubaOS Crypto Module is used for IPsec/IKE session cryptography. Both modules implement the low level cryptographic function in support of the protocols and run self-tests. The ArubaOS Bootloader Module provides the cryptographic functionality to perform firmware integrity tests on boot of the device.

Table 13: SFR to CAVP Mapping

SFR	Algorithm Capability	CAVP
FCS_CKM.1 Cryptographic Key Generation	RSA KeyGen (FIPS Pub 186-4) (2048-bit)	A2689 A2690
	ECDSA KeyGen (FIPS Pub 186-4) (P-256, P-384)	
	FFC Safe Prime Groups (NIST SP 800-56A Rev. 3, RFC 3526)	A2689
FCS_CKM.2 Cryptographic Key Establishment	Elliptic Curve-based Schemes (NIST SP 800-56A Rev 3)	A2689 A2690
	FFC Safe Prime Groups (NIST SP 800-56A Rev 3 and Groups Listed in RFC 3526)	A2689
FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption)	AES CBC (128 and 256 bits)	A2689 A2690
	AES GCM (128 and 256 bits)	A2689 A2690
	AES CTR (128 and 256 bits)	A2690
FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification)	RSA SigGen (FIPS 186-4) (modulus 2048 bits) RSA SigVer (FIPS 186-4) (modulus 2048 bits)	A2688 (Firmware/Update SigVer Only) A2690
	ECDSA SigGen (FIPS 186-4) (256, 384 bits)	A2689 A2690

	ECDSA SigVer (FIPS 186-4) (256, 384 bits)	
FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)	SHA-1, SHA-256, SHA-384, SHA-512 (160, 256, 384, and 512 bits respectively)	A2688 (SHA-256 Only) A2690
FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm)	HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384 (192, 256, and 384 bits respectively)	A2689 A2690
FCS_RBG_EXT.1	CTR_DRBG (AES) (256 bits)	A2690

6.2.1 FCS_CKM.1

33 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.

6.2.2 FCS_CKM.2

34 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.

35 Table 14 below identifies the scheme being used by each service.

Table 14: Key Agreement Mapping

Scheme	SFR	Service
ECC	FCS_TLSS_EXT.1	GUI / Administration
	FCS_SSHS_EXT.1	CLI / Administration
	FCS_IPSEC_EXT.1	Audit Server Authentication Server Aruba Mobility Controller NTP
FFC Safe Prime Groups	FCS_IPSEC_EXT.1	Audit Server

Scheme	SFR	Service
		Authentication Server Aruba Mobility Controller NTP

6.2.3 FCS_CKM.4

36 Table 16 shows the origin, storage location and destruction details for cryptographic keys.

6.2.4 FCS_COP.1/DataEncryption

37 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.

38 The relevant NIST CAVP certificate numbers are listed Table 4.

6.2.5 FCS_COP.1/SigGen

39 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.

40 Signature verification services are used in the TLS, SSH and IPsec protocols. Additionally, RSA signature verification is used for trusted updates.

41 The relevant NIST CAVP certificate numbers are listed in Table 4.

6.2.6 FCS_COP.1/Hash

42 The TOE provides cryptographic hashing services using SHA-1, SHA-256, SHA-384, and SHA-512.

43 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

44 The relevant NIST CAVP certificate numbers are listed in Table 4.

6.2.7 FCS_COP.1/KeyedHash

45 The TOE provides keyed-hashing message authentication services using HMAC-SHA-1, HMAC-SHA-256 and HMAC-SHA-384.

46 HMAC is implemented in the following protocols: TLS, IPsec and SSH.

47 The characteristics of the HMACs used in the TOE are given in Table 15.

Table 15: HMAC Characteristics

Algorithm	Block Size	Key Size	Digest Size
HMAC-SHA-1	512 bits	192 bits	160 bits

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

48 The relevant NIST CAVP certificate numbers are listed in Table 4.

6.2.8 FCS_HTTPS_EXT.1

49 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.

50 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.

6.2.9 FCS_IPSEC_EXT.1

51 The TOE includes an implementation of IPsec in accordance with RFC 4301 for security. The TOE supports IPsec for tunnel mode. The IPsec ESP protocol is implemented in conjunction with AES-CBC-128 and AES-CBC-256 (as specified by RFC 3602) together with the following truncated versions of SHA-based HMAC algorithms: HMAC-SHA1-96 and with AES-GCM-128 and AES-GCM-256 (as specified by RFC 4106).

52 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. The TOE uses the AES-CBC-128 and AES-CBC-256 algorithms as specified in RFC 3602 to encrypt the payload. For IKE SA's, the TOE supports the following HMAC algorithms: HMAC-SHA1-96, HMAC-SHA1-160 (HMAC-SHA1), HMAC-SHA2-256 (HMAC_SHA2_256_128), HMAC-SHA2-384 (HMAC_SHA2_384_192).

53 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.

54 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. 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.

55 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.

56 Pre-shared keys used for IPsec can be constructed of essentially any alphabetic character (upper and lower case), numerals, and special characters (e.g., “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, and “)”) and can be anywhere from 6-160 characters in length (e.g., 22 characters). 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.

57 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.

58 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.

6.2.10 FCS_NTP_EXT.1

59 The TOE supports NTP v4 by implementing the ntpd Linux daemon in client mode and can synchronize with at least three NTP time sources. The TOE supports IPsec between itself and the NTP server to provide trusted communications.

6.2.11 FCS_RBG_EXT.1

60 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.

61 Additional detail is provided in the proprietary Entropy Description.

6.2.12 FCS_SSHS_EXT.1

62 The TOE implements SSH in compliance with RFCs 4251, 4252, 4253, 4254, 4344, 5656, 6668, 8308 section 3.1 and 8332.

63 The TOE supports password-based or public key authentication and supports ssh-rsa, rsa-sha2-256, and rsa-sha2-512 for server authentication. The TOE supports ssh-rsa, rsa-sha2-256, and rsa-sha2-512 for client authentication.

64 During authentication, the TOE establishes a user identity by either verifying that the SSH client’s current public key matches the one stored within the TOE’s SSH authorized keys file, or by confirming the validity of the presented username and matching password within its database.

65 The TOE examines the size of each received SSH packet. If the packet is greater than 256KB, it is automatically dropped.

66 The TOE utilises AES-CBC-128, AES-CBC-256, AES-128-CTR and AES-256-CTR for SSH encryption.

67 The TOE provides data integrity for SSH connections via HMAC-SHA1 and HMAC-SHA2-256.

68 The TOE supports ecdh-sha2-nistp256 and ecdh-sha2-nistp384 for SSH key exchanges.

69 The TOE will re-key SSH connections after 1 hour or after an aggregate of 1 gig of data has been exchanged (whichever occurs first).

6.2.13 FCS_TLSS_EXT.1

70 The TOE operates as a TLS server for the web GUI trusted path.

71 The TOE restricts the use of TLS protocols to version 1.2 exclusively and rejects any other protocol version.

72 The TOE supports remote administration via the Web GUI is protected usingTLS/HTTPS. 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

73 Ciphersuites are not user-configurable.

74 The TOE performs key establishment for TLS using ECDHE curves secp256r1.

75 The TOE supports session resumption based on session IDs according to RFC 5246 and session tickets according to RFC 5077. 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.

76 Session tickets are protected by implementing symmetric encryption algorithms as described in FCS_COP.1/DataEncryption and section 6.2.4. Session tickets are encrypted according to the TLS negotiated symmetric encryption algorithm. Session tickets adhere to the structural format provided in section 4 of RFC 5077.

6.3 Identification and Authentication

6.3.1 FIA_PMG_EXT.1

77 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 “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, [“`”, “~”, “-“, “_”, “=”, “+”, “[“, “]”, “{“, “}”, “\”, “|”, “;”, “:”, “‘”, “”, “,”, “.”, “<”, “>”, “/”

78 The minimum password length is settable by the Administrator and can range from 8 to 32 characters.

79 The maximum password length supported by the TOE is 128 characters.

6.3.2 FIA_UIA_EXT.1

80 The TOE requires all users to be successfully identified and authenticated. The TOE warning banner may be viewed prior to authentication. 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.

81 Administrative access to the TOE is facilitated through one of several interfaces:

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

82 In addition to the local authentication mechanism described by FIA_UAU_EXT.2, the TOE supports RADIUS and TACACS+ authentication servers. A trusted channel using IPsec is established between the TOE and the external authentication server.

83 Remote administrators are configured as users who have privileges to access the CLI and Web GUI administration interfaces and who are authenticated as users using the local database or an external authentication server.

84 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 connection to the CLI only supports username/password.

6.3.3 FIA_UAU_EXT.2

85 Regardless of the interface at which the administrator interacts, the TOE prompts the user for a credential. Only after the administrative user presents the correct authentication credentials will they be granted access to the TOE administrative functionality. No TOE administrative access is permitted until an administrator is successfully identified and authenticated.

86 The TOE provides a local password-based authentication mechanism (in addition to remote authentication servers described at FIA_UIA_EXT.1).

87 The process for authentication is the same for administrative access whether administration is occurring via direct connection or remotely. At initial login, the administrative user is prompted to provide a username. After the user provides the username, the user is prompted to provide the administrative credential associated with the user account (e.g. password or SSH public/private key response). The TOE then either grants administrative access (if the combination of username/password

and/or public key is correct) or indicates that the login was unsuccessful. The TOE does not provide a reason for failure in the cases of a login failure.

6.3.4 FIA_UAU.7

88 The TOE provides obscured feedback while administrators are authenticating. Authentication via the local console will display no echoed characters when characters are entered. Authentication via the remote interfaces (such as the Web UI) will only display black dots when characters are entered into the password field.

6.3.5 FIA_AFL.1

89 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.

90 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.

91 The administrator can configure the maximum number of failed attempts and the lockout time threshold using the web GUI or CLI.

92 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.

6.3.6 FIA_X509_EXT.1/Rev

93 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

94 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 extendedKeyUsagefield

95 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.

96 The X.509 certificates for each of the given scenarios are validated using the certificate path validation algorithm defined in RFC 5280, which can be summarized as follows:

a) The public key algorithm and parameters are checked

b) The current date/time is checked against the validity period revocation status is checked

c) Issuer name of X matches the subject name of X+1

d) Name constraints are checked

e) Policy OIDs are checked

f) Policy constraints are checked; issuers are ensured to have CA signing bits

g) Path length is checked

h) Critical extensions are processed

97 If at any point a certificate under review fails the trust chain verification activity or revocation status check, then the entire trust chain is deemed untrusted.

6.3.7 FIA_X509_EXT.2

98 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.

99 Instructions for configuring the trusted IT entities to supply appropriate X.509 certificates are captured in the guidance documents.

100 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.

6.3.8 FIA_X509_EXT.3

101 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.

6.4 Security Management

6.4.1 FMT_MOF.1/ManualUpdate

102 The TOE restricts the ability to perform software updates to Security Administrators.

6.4.2 FMT_MOF.1/Functions

103 The TOE restricts the ability to modify (enable/disable) transmission of audit records to an external audit server to Security Administrators.

6.4.3 FMT_MTD.1/CoreData

104 Users are required to login before being provided with access to any administrative functions. Access to TSF data and functions, including managing the TOE's trust store, is restricted to Security Administrators as described by FMT_SMR.2 below.

6.4.4 FMT_SMR.2

105 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.

106 Management of TSF data via the CLI or web GUI is restricted to Security Administrators.

6.4.5 FMT_MTD.1/CryptoKeys

107 The TOE restricts the ability to manage SSH, TLS, IPsec and any configured X.509 private keys to Security Administrators.

6.4.6 FMT_SMF.1

108 The TOE may be managed via the CLI (local 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 modify audit transmission behaviour (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.

6.5 Protection of the TSF

6.5.1 FPT_SKP_EXT.1

109 Keys are protected as described in Table 16. 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).

Table 16: Keys

Reference	Type	Generation and Use	Storage	Zeroization
DRBG entropy input	SP800-90a CTR_DRBG (512 bits)	Entropy inputs to the DRBG function used to construct the DRBG seed.	Stored in SDRAM memory (plaintext)	Zeroized by rebooting the module
DRBG seed	SP800-90a CTR_DRBG (384-bits)	Input to the DRBG that determines the internal state of the DRBG. Generated using DRBG derivation function.	Stored in SDRAM memory (plaintext)	Zeroized by rebooting the module
DRBG Key	SP800-90a CTR_DRBG (256 bits)	This is the DRBG key used for SP800-90a CTR_DRBG.	Stored in SDRAM memory (plaintext)	Zeroized by rebooting the module
DRBG V	SP800-90a CTR_DRBG V (128 bits)	Internal V value used as part of SP800-90a CTR_DRBG	Stored in SDRAM memory (plaintext)	Zeroized by rebooting the module
Diffie-Hellman private key	Diffie-Hellman Group 14 (224 bits)	Generated internally during Diffie-Hellman Exchange. Used for establishing DH shared secret.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
Diffie-Hellman public key	Diffie-Hellman Group 14 (2048 bits)	Generated internally during Diffie-Hellman Exchange. Used for establishing DH shared secret.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
Diffie-Hellman shared secret	Diffie-Hellman Group 14 (2048 bits)	Established during Diffie-Hellman Exchange. Used for deriving IPSec/IKE cryptographic keys.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
EC Diffie-Hellman private key	EC Diffie-Hellman (Curves: P-256 or P-384).	Generated internally during EC Diffie-Hellman Exchange. Used for establishing ECDH shared secret.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module

Reference	Type	Generation and Use	Storage	Zeroization
EC Diffie-Hellman public key	EC Diffie-Hellman (Curves: P-256 or P-384).	Generated internally during EC Diffie-Hellman Exchange. Used for establishing ECDH shared secret.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
EC Diffie-Hellman shared secret	EC Diffie-Hellman (Curves: P-256 or P-384)	Established during EC Diffie-Hellman Exchange. Used for deriving IPSec/IKE cryptographic keys.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
RADIUS server shared secret	8-128 characters shared secret	Entered by Security Administrator. Used for RADIUS server authentication.	Stored in Flash memory (plaintext).	Zeroized by using command 'write erase all' or by overwriting with a new secret
SKEYSEED	Shared Secret (160/256/384 bits)	A shared secret known only to IKE peers. It was derived via key derivation function defined in SP800-135 KDF (IKEv2) and it will be used for deriving IKE session authentication key.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
IKE session authentication key	HMAC-SHA-1/256/384 (160/256/384 bits)	The IKE session (IKE Phase I) authentication key. This key is derived via key derivation function defined in SP800-135 KDF (IKEv2). Used for IKEv2 payload integrity verification.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
IKE session encryption key	AES (128/192/256 bits, CBC)	The IKE session (IKE Phase I) encrypt key. This key is derived via key derivation function defined in SP800-135 KDF	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module

Reference	Type	Generation and Use	Storage	Zeroization
		(IKEv2). Used for IKE payload protection.		
IPSec session encryption key	AES and AES-GCM (128/256 bits, CBC) NOTE: 192 bit CAVS tested, but not used.	The IPsec (IKE phase II) encryption key. This key is derived via a key derivation function defined in SP800-135 KDF (IKEv2). Used to secure IPsec traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
IPSec session authentication key	HMAC-SHA-1 (160 bits)	The IPsec (IKE Phase II) authentication key. This key is derived via using the KDF defined in SP800-135 KDF (IKEv2). Used to verify the integrity of IPsec traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
IPSec Pre-Shared Key	6-160 characters pre-shared secret	Administrator generated.	Stored in Flash memory (ciphertext). AES encrypted with KEK.	Zeroized by using command 'write erase all' or by overwriting with a new secret
SSHv2 session key	AES (128/192/256 bits)	This key is derived via a key derivation function defined in SP800-135 KDF (SSHv2). Used to secure SSHv2 traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
SSHv2 session authentication key	HMAC-SHA-1 (160-bit)	This key is derived via a key derivation function defined in SP800-135 KDF (SSHv2). Used to verify the integrity of SSHv2 traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
TLS pre-master secret	48 bytes secret	This key is transferred into the module, protected	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module

Reference	Type	Generation and Use	Storage	Zeroization
		by TLS RSA public key.		
TLS session encryption key	AES 128/192/256 bits	This key is derived via a key derivation function defined in SP800-135 KDF (TLS). Used to secure TLS traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
TLS session authentication key	HMAC-SHA-1/256/384 (160/256/384 bits)	This key is derived via a key derivation function defined in SP800-135 KDF (TLS). Used to verify the integrity of TLS traffic.	Stored in SDRAM memory (plaintext).	Zeroized by rebooting the module
RSA Private Key	RSA 2048 bit private key	This key is generated in the module. Used for IKEv2, TLS, OCSP (signing OCSP messages) and EAP-TLS peers authentication.	Stored in Flash memory (plaintext).	Zeroized by using command 'write erase all'
RSA public key	RSA 2048 bits public key	This key is generated in the module. This Key can also be entered by the CO via SSH (CLI) and/or TLS (for the GUI). Used for IKEv2, TLS, OCSP (verifying OCSP messages) and EAP-TLS peers authentication.	Stored in Flash memory (plaintext).	RSA public key
ECDSA Private Key	ECDSA suite B P-256 and P-384 curves	This key is generated in the module. Used for IKEv2, TLS and EAP-TLS peers authentication.	Stored in Flash memory (plaintext).	Zeroized by using command 'write erase all'
ECDSA Public Key	ECDSA suite B P-256 and P-384 curves	This key is generated in the module. This Key can also be entered	Stored in Flash memory (plaintext).	Zeroized by using command 'write erase all'.

Reference	Type	Generation and Use	Storage	Zeroization
		by the CO via SSH (CLI) and/or TLS (for the GUI). Used for IKEv2, TLS and EAP-TLS peers authentication.		
Factory CA Public Key	RSA (2048 bits)	This is RSA public key. Loaded into the module during manufacturing. Used for Firmware verification.	Stored in Flash (plaintext).	Zeroized by using command 'write erase all'
Key Encryption Key (KEK)	Triple-DES (192 bits)	Hardcoded during manufacturing. Used only to protect keys stored in the flash, not for key transport.	Stored in Flash memory (plaintext).	Zeroized by using command 'write erase all'

6.5.2 FPT_APW_EXT.1

110 Passwords are protected as describe in Table 17. In all cases plaintext passwords cannot be viewed through an interface designed specifically for that purpose.

Table 17: Passwords

Reference	Type	Generation	Storage	Zeroization
Enable secret	8-32 characters password	Entered by Security Administrator.	Stored in Flash memory (ciphertext) encrypted with KEK	Zeroized by using command 'write erase all' or by overwriting with a new secret
User Password	8-32 characters password	Entered by Security Administrator.	Stored in Flash memory (ciphertext) encrypted with KEK	Zeroized by using command 'write erase all' or by overwriting with a new secret

6.5.3 FPT_TST_EXT.1

111 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).

112 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)

113 The following firmware integrity tests are performed:

a) ArubaOS Uboot BootLoader Module

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

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

115 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.

116 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.

117 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.

118 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.

6.5.4 FPT_TUD_EXT.1

119 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.

120 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.

121 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.

122 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.

123 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.

6.5.5 FPT_STM_EXT.1

124 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.

125 The TOE makes use of time for the following functions:

a) Audit record timestamps

b) Session timeouts (lockout enforcement)

c) Certificate validation

6.6 TOE Access

6.6.1 FTA_SSL_EXT.1

126 The Security Administrator may configure the TOE to terminate an inactive local interactive session (CLI) following a specified period of time.

6.6.2 FTA_SSL.3

127 The Security Administrator may configure the TOE to terminate an inactive remote interactive session (CLI and Web UI) following a specified period of time.

6.6.3 FTA_SSL.4

128 Administrative users may terminate their own sessions at any time by providing a logout command (CLI) and logout option under user menu (GUI).

6.6.4 FTA_TAB.1

129 The TOE displays an administrator configurable message to users prior to login at the CLI (local console and SSH) and web GUI (HTTPS).

6.7 Trusted Path/Channels

6.7.1 FTP_ITC.1

130 The TOE uses the IPsec protocol with certificates to establish VPN tunnels for trusted channels between external IT entities.

131 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.

132 The TOE operates as an IPsec peer and has the ability to act as an initiator.

6.7.2 FTP_TRP.1/Admin

133 The TOE provides the following trusted paths for remote administration:

a) CLI over SSH

b) Web GUI over HTTPS

7 Rationale

7.1 Conformance Claim Rationale

134 The following rationale is presented with regard to the PP conformance claims:

a) **TOE type.** As identified in section 2.1, the TOE is network device, consistent with the NDcPP.

b) **Security problem definition.** As shown in section 3, the threats, OSPs and assumptions are reproduced directly from the NDcPP.

c) **Security objectives.** As shown in section 4, the security objectives are reproduced directly from the NDcPP.

d) **Security requirements.** As shown in section 5, the security requirements are reproduced directly from the NDcPP. No additional requirements have been specified.

7.2 Security Objectives Rationale

135 All security objectives are drawn directly from the NDcPP.

7.3 Security Requirements Rationale

136 All security requirements are drawn directly from the NDcPP. Table 18 presents a mapping between threats and SFRs as presented in the NDcPP.

Table 18: NDcPP SFR Rationale

Identifier	SFR Rationale
T.UNAUTHORIZED_ADMINISTRATOR_ACCESS	• The Administrator role is defined in FMT_SMR.2 and the relevant administration capabilities are defined in FMT_SMF.1 and FMT_MTD.1/CoreData, with optional additional capabilities in FMT_MOF.1/Services and FMT_MOF.1/Functions • The actions allowed before authentication of an Administrator are constrained by FIA_UIA_EXT.1, and include the advisory notice and consent warning message displayed according to FTA_TAB.1 • The requirement for the Administrator authentication process is described in FIA_UAU_EXT.2 • Locking of Administrator sessions is ensured by FTA_SSL_EXT.1 (for local sessions), FTA_SSL.3 (for remote sessions), and FTA_SSL.4 (for all interactive sessions) • The secure channel used for remote Administrator connections is specified in FTP_TRP.1/Admin • (Malicious actions carried out from an Administrator session are separately addressed by T.UNDETECTED_ACTIVITY)

Identifier	SFR Rationale
	• (Protection of the Administrator credentials is separately addressed by T.PASSWORD_CRACKING).
T.WEAK_CRYPTOGRAPHY	• Requirements for key generation and key distribution are set in FCS_CKM.1 and FCS_CKM.2 respectively • Requirements for use of cryptographic schemes are set in FCS_COP.1/DataEncryption, FCS_COP.1/SigGen, FCS_COP.1/Hash, and FCS_COP.1/KeyedHash • Requirements for random bit generation to support key generation and secure protocols (see SFRs resulting from T.UNTRUSTED_COMMUNICATION_CHANNELS) are set in FCS_RBG_EXT.1 • Management of cryptographic functions is specified in FMT_SMF.1
T.UNTRUSTED_COMMUNICATION_CHANNELS	• The general use of secure protocols for identified communication channels is described at the top level in FTP_ITC.1 and FTP_TRP.1/Admin; for distributed TOEs the requirements for inter-component communications are addressed by the requirements in FPT_ITT.1 • Requirements for the use of secure communication protocols are set for all the allowed protocols in FCS_DTLSC_EXT.1, FCS_DTLSC_EXT.2, FCS_DTLSS_EXT.1, FCS_DTLSS_EXT.2, FCS_HTTPS_EXT.1, FCS_IPSEC_EXT.1, FCS_SSHC_EXT.1, FCS_SSHS_EXT.1, FCS_TLSC_EXT.1, FCS_TLSC_EXT.2, FCS_TLSS_EXT.1, FCS_TLSS_EXT.2 • Optional and selection-based requirements for use of public key certificates to support secure protocols are defined in FIA_X509_EXT.1, FIA_X509_EXT.2, FIA_X509_EXT.3
T.WEAK_AUTHENTICATION_ENDPOINTS	• The use of appropriate secure protocols to provide authentication of endpoints (as in the SFRs addressing T.UNTRUSTED_COMMUNICATION_CHANNELS) are ensured by the requirements in FTP_ITC.1 and FTP_TRP.1/Admin; for distributed TOEs the authentication requirements for endpoints in inter-component communications are addressed by the requirements in FPT_ITT.1 • Additional possible special cases of secure authentication during registration of distributed TOE components are addressed by FCO_CPC_EXT.1 and FTP_TRP.1/Join.

Identifier	SFR Rationale
T.UPDATE_COMPROMISE	• Requirements for protection of updates are set in FPT_TUD_EXT.1 • Additional optional use of certificate-based protection of signatures can be specified using FPT_TUD_EXT.2, supported by the X.509 certificate processing requirements in FIA_X509_EXT.1, FIA_X509_EXT.2 and FIA_X509_EXT.3 • Requirements for management of updates are defined in FMT_SMF.1 and (for manual updates) in FMT_MOF.1/ManualUpdate, with optional requirements for automatic updates in FMT_MOF.1/AutoUpdate
T.UNDETECTED_ACTIVITY	• Requirements for basic auditing capabilities are specified in FAU_GEN.1 and FAU_GEN.2, with timestamps provided according to FPT_STM_EXT.1 and if applicable, protection of NTP channels in FCS_NTP_EXT.1 • Requirements for protecting audit records stored on the TOE are specified in FAU_STG.1 • Requirements for secure transmission of local audit records to an external IT entity via a secure channel are specified in FAU_STG_EXT.1 • Optional additional requirements for dealing with potential loss of locally stored audit records are specified in FAU_STG_EXT.2/LocSpace, and FAU_STG_EXT.3/LocSpace • If (optionally) configuration of the audit functionality is provided by the TOE then this is specified in FMT_SMF.1, and confining this functionality to Security Administrators is required by FMT_MOF.1/Functions.
T.SECURITY_FUNCTIONALITY_COMPROMISE	• Protection of secret/private keys against compromise is specified in FPT_SKP_EXT.1 • Secure destruction of keys is specified in FCS_CKM.4 • If (optionally) management of keys is provided by the TOE then this is specified in FMT_SMF.1, and confining this functionality to Security Administrators is required by FMT_MTD.1/CryptoKeys • (Protection of passwords is separately covered under T.PASSWORD_CRACKING)

Identifier	SFR Rationale
T.PASSWORD_CRACKING	• Requirements for password lengths and available characters are set in FIA_PMG_EXT.1 • Protection of password entry by providing only obscured feedback is specified in FIA_UAU.7 • Actions on reaching a threshold number of consecutive password failures are specified in FIA_AFL.1 • Requirements for secure storage of passwords are set in FPT_APW_EXT.1.
T.SECURITY_FUNCTIONALITY_FAILURE	• Requirements for running self-test(s) are defined in FPT_TST_EXT.1 • Optional use of certificates to support self-test(s) is defined in FPT_TST_EXT.2 (with support for the use of certificates in FIA_X509_EXT.1, FIA_X509_EXT.2, and FIA_X509_EXT.3),
P.ACCESS_BANNER	• An advisory notice and consent warning message is required to be displayed by FTA_TAB.1

Annex A: Extended Components Definition

137 Refer to the Extended Components Definitions section of the PP as follows:

a) CPP_ND_V2.2E – Appendix ‘C’