Palo Alto Networks
WF-500
WildFire 8.1.11
Security Target

Version 1.0
January 3, 2020

Palo Alto Networks, Inc.
3000 Tannery Way
Santa Clara, CA 95054

Table of Contents

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is a physical appliance running WildFire software version 8.1.11, provided by Palo Alto Networks, Inc.

The physical appliance is the WF-500, which is an on-premise network device that identifies unknown malware, zero-day exploits, and Advanced Persistent Threats (APTs) through dynamic analysis, and automatically disseminates protection in near real-time to help security teams meet the challenge of advanced cyber-attacks. Unknown files are analyzed by WildFire in a scalable sandbox environment where new threats are identified, and protections are automatically developed and delivered in the form of an update. The result is a unique, closed loop approach to controlling cyber threats that begins with positive security controls to reduce the attack surface, inspection of all traffic, ports, and protocols to block all known threats, and rapid detection of unknown threats by observing their actual behavior. The appliance's architecture allows organizations to meet privacy and regulatory requirements for local analysis while still benefiting from shared threat intelligence and protections from other WildFire subscribers.

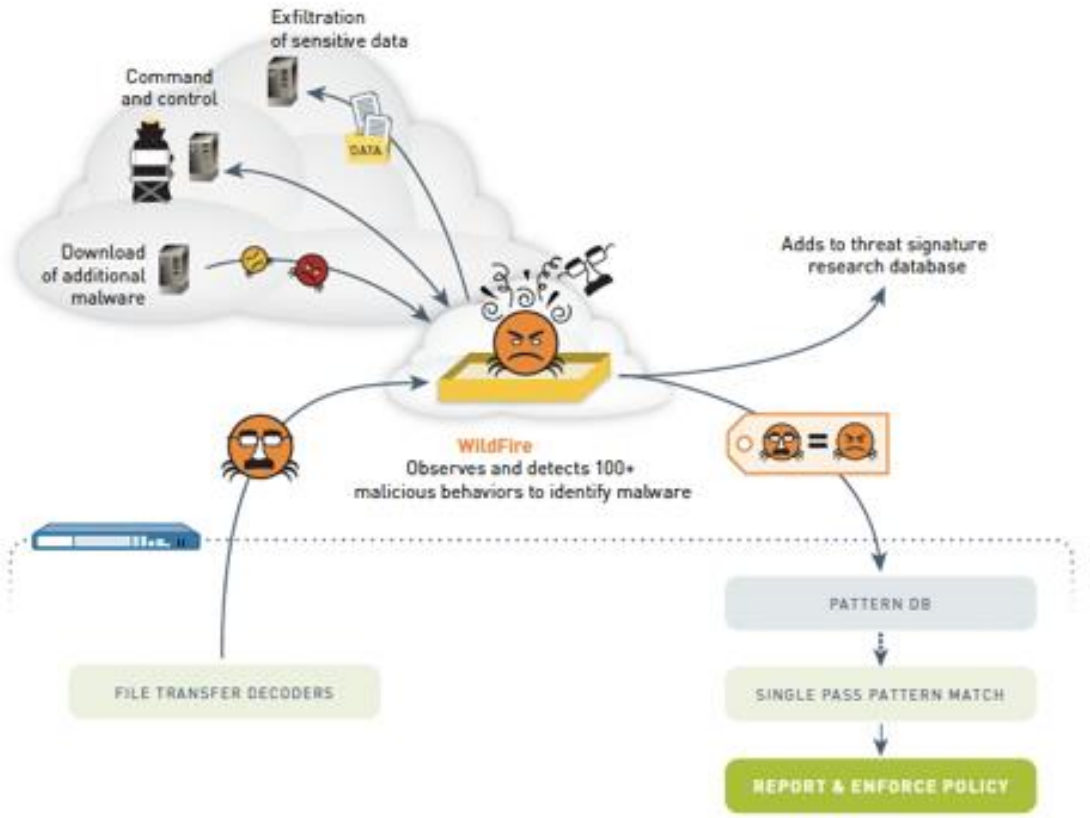

Figure 1 - WildFire File Analysis

The focus of this evaluation is on the TOE functionality supporting the claims in the collaborative Protection Profile for Network Devices. (See Section 1.2 for specific version information).

The only capabilities covered by the evaluation are those specified in the aforementioned Protection Profile, all other capabilities are not covered in the evaluation. The security functionality specified in [NDcPP] includes protection of communications between the TOE and trusted external IT entities (trusted channel), protection of communications between the TOE and remote administrators (trusted path), identification and authentication of administrators, auditing of security-relevant events, ability to verify the source and integrity of updates to the TOE, implementation of session idle timeout, and the restricted use of FIPS Approved algorithms and protocols.

The Security Target contains the following additional sections:

- Product Description
- Security Problem Definition
- Security Objectives
- IT Security Requirements
- TOE Summary Specification
- Protection Profile Claims
- Rationale

1.1 Security Target, TOE and CC Identification

ST Title: Palo Alto Networks WF-500 with WildFire 8.1.11.
ST Version: 1.0
ST Date: 1/3/2020
TOE Identification: Palo Alto Networks WildFire WF-500 running version 8.1.11.
TOE Developer: Palo Alto Networks, Inc.
Evaluation Sponsor: Palo Alto Networks, Inc.
CC Identification: *Common Criteria for Information Technology Security Evaluation, Version 3.1, Release 5, April 2017*

1.2 Conformance Claims

PP Reference: collaborative Protection Profile for Network Devices, Version 2.1, 24-September-2018

The following NIAP Technical Decisions apply to this PP, and have been accounted for in the ST development:

- TD0395: NIT Technical Decision for Different Handling of TLS1.1 and TLS1.2
- TD0396: NIT Technical Decision for FCS_TLSC_EXT.1.1, Test 2
- TD0397: NIT Technical Decision for Fixing AES-CTR Mode Tests
- TD0398: NIT Technical Decision for FCS_SSH*EXT.1.1 RFCs for AES-CTR
- TD0399: NIT Technical Decision for Manual installation of CRL (FIA_X509_EXT.2)
- TD0400: NIT Technical Decision for FCS_CKM.2 and elliptic curve-based key establishment
- TD0402: NIT Technical Decision for RSA-based FCS_CKM.2 Selection
- TD0408: NIT Technical Decision for local vs. remote administrator accounts
- TD0409: NIT decision for Applicability of FIA_AFL.1 to key-based SSH authentication
- TD0410: NIT technical decision for Redundant assurance activities associated with FAU_GEN.1
- TD0412: NIT Technical Decision for FCS_SSHS_EXT.1.5 SFR and AA discrepancy
- TD0423: NIT Technical Decision for Clarification about application of Rfl#201726rev2
- TD0424: NIT Technical Decision for NDcPP v2.1 Clarification – FCS_SSHC/S_EXT1.5
- TD0425: NIT Technical Decision for Cut-and-paste Error for Guidance AA
- TD0448: NIT Technical Decision for Documenting Diffie-Hellman 14 groups
- TD0449: NIT Technical Decision for Identification of usage of cryptographic schemes
- TD0450: NIT Technical Decision for RSA-based ciphers and the Server Key Exchange message
- TD0452: NIT Technical Decision for FCS_(D)TLSC_EXT.X.2 IP addresses in reference identifiers

- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Release 5, April 2017.
    - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Release 5, April 2017.
    - Part 3 Conformant.

1.3 Conventions

The following conventions have been applied to this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
    - Iteration: allows a component to be used more than once with varying operations. In the ST, iteration is indicated by adding a string starting with "/" (e.g. "FCS_COP.1/Hash").
    - Selection wholly or partially completed in the PP: the selection values (i.e. the selection values adopted in the PP or the remaining selection values available for the ST) are indicated with **bolded text**
        - For example, [*selection: SHA-1, SHA-256, SHA-384, SHA-512*] will be [***SHA-1, SHA-256, SHA-384, SHA-512***]
    - Assignment wholly or partially completed in the PP: indicated with *italicized text*;
        - For example, between [*assignment: minimum number of characters supported by the TOE*] and [*assignment: number of characters greater than or equal to 15]* characters will be between [*6*] and [*15*] characters.
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… some **big** things …"). Note that 'cases' that are not applicable in a given SFR have simply been removed without any explicit identification.
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.
- The ST does not change operations that have been completed by the PP and EP authors, nor undo formatting. For example, if the text is italicized by the PP author, the ST author will not undo it. However, if it's a selection or assignment operation made by the ST author, the ST author has bolded the text as well to indicate that a selection/assignment operation was performed.

1.3.1 Terminology

The following terms and abbreviations are used in this ST:

- WF – WildFire appliance
- UID – Unique Identification feature is a combination LED/button that is used to assist a technician in locating a device
- CO – Cryptographic Officer (Administrator or superuser)
- CCECG – Common Criteria Evaluated Configuration Guide used to assist an administrator with steps for configuring the TOE properly

1.3.2 Acronyms

AES	Advanced Encryption Standard
CBC	Cipher-Block Chaining
CC	Common Criteria for Information Technology Security Evaluation
CEM	Common Evaluation Methodology for Information Technology Security
CM	Configuration Management

CLI	Command Line Interface
CPU	Central Processing Unit
DH	Diffie-Hellman
EEPROM	Electrically Erasable Programmable Read-Only Memory
EP	Extended Package
FIA	Identification and Authentication CC Class
FIPS	Federal Information Processing Standard
FMT	Security Management CC Class
FSP	Functional Specification
FTP	File Transfer Protocol
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
HTTP(S)	Hypertext Transfer Protocol (Secure)
IKE	Internet Key Exchange
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
IPsec	Internet Protocol Security
NDcPP	Collaborative Protection Profile for Network Devices
NAT	Network Address Translation
NIST	National Institute of Standards and Technology
PP	Protection Profile
REST	Representational State Transfer
RSA	Rivest, Shamir and Adleman (algorithm for public-key cryptography)
SA	Security Association
SAR	Security Assurance Requirement
SFP	Security Function Policy
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SM	Security Management
SMR	Security Management Roles
SNMP	Simple Network Management Protocol
SSH	Secure Shell
SSL	Secure Socket Layer Protocol
ST	Security Target
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functions
TSP	TOE Security Policy
UDP	User Data Protection
URL	Uniform Resource Locator

2. Product Description

The TOE is the Palo Alto Networks WF-500 appliance, which utilizes the WildFire 8.1.11 software. It receives network traffic samples from individual Palo Alto Networks Firewalls and can be managed by Panorama devices in its Operational Environment over a secure channel.

The evaluation only includes the WF-500 physical device as identified in sections above. Other deployments of WildFire are cloud-based and are not within scope of this evaluation. Palo Alto Networks Next-Generation Firewall and Panorama devices are components of the TOE's Operational Environment and were evaluated previously, and information about them are provided for completeness only.

2.1 TOE Overview

It is required that the TOE be placed in FIPS-CC mode, which ensures that only FIPS/CC approved/allowed algorithms are utilized when interacting with other devices in the operational network. For communications with Palo Alto Networks Firewalls and Panorama appliances, only the secure communication channels are claimed.

The TOE utilizes various protocols in its communication with other devices. IPsec is used for communication between WF-500 devices to send sensitive data to each other in a protected manner while TLS 1.2 is used to secure connections between the TOE and other devices on the network such as the Panorama and Firewall. For any updates that are required on the TOE such as configuration changes that are processed via CLI, SSHv2 is used to secure these connections.

The various protocols implemented by the module include TLS, SSH, and IPsec. The TOE has the ability to handle certificates for their desired purpose as well as being able to generate certificates that can be used for these functions. Administrators are able to setup the TOE to support mutual authentication to improve the security of their appliance as it communicates with other devices.

As a network device, the TOE is required to generate logging events, which can be used by administrators to audit functions of the TOE as well as its interactions with other components. The TOE provides the ability to store logs locally, and also has the ability to send these logs to an external syslog server via a TLS connection.

To protect the TOE from unauthorized access and disruption, the TOE has security features in place that will log out idle administrators, lock the device in the event of too many failed authentication attempts, and has the ability to increase the password length to harden the credentials of administrators.

Operators that provide the correct credentials to the TOE are able to perform all management functions via the CLI, which is protected using SSHv2. Configurations can be updated for the ability of the appliance to communicate with other devices on the network, generate certificates, and perform security actions such as zeroization.

Figure 2 provides an overview of the communication protocols used between the TOE and other devices on the network. The PAN-OS Firewall and Panorama devices are in the operational environment, and only the secure communication channels from the WildFire to those devices are claimed.

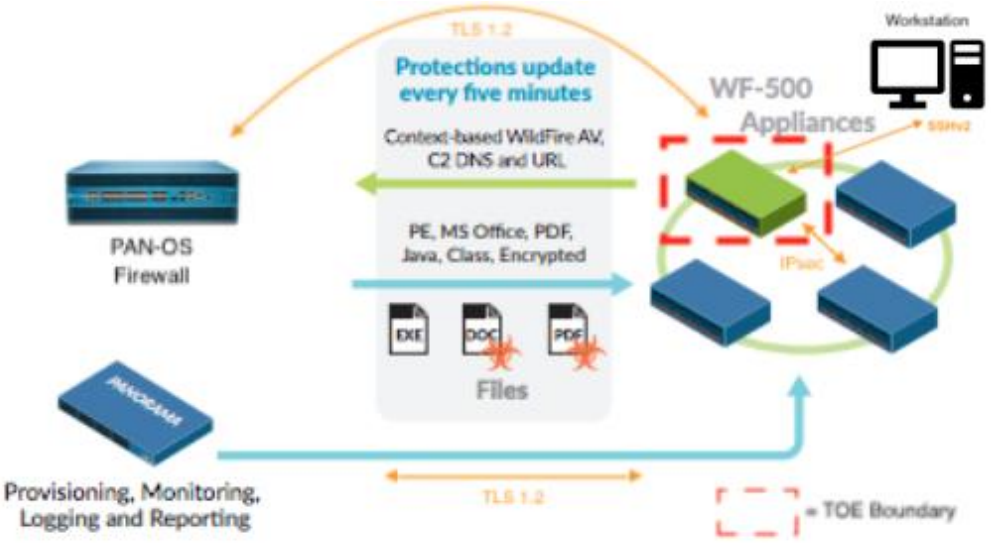

Figure 2 - WildFire Interaction with Systems

2.2 TOE Architecture

The TOE is a hardware and software solution that is comprised of items listed in Section 2.2.1 and 2.2.2. The software comes pre-installed on the device and can be updated by downloading a new version from the Palo Alto Networks support site. The system consists of the following items: system software, database, linux-derived operating system, and the hardware. The database is a repository for audit logs, user logs, and system/configuration data. The system software contains necessary items to support the functionality of the device such as using OpenSSL/OpenSSH, and items necessary for management interfaces (CLI). The operating system is PAN-OS 8.1.11. PAN-OS 8.1.11 is an operating system derived from Linux kernel version 3.10.88 to enforce domain separation, memory management, disk access, file I/O, and communications with the underlying hardware components including memory, network I/O, CPUs, and hard disks. Only services and libraries required by the system software and DB are enabled in the OS.

The following diagram demonstrates the software and hardware architecture of the TOE.

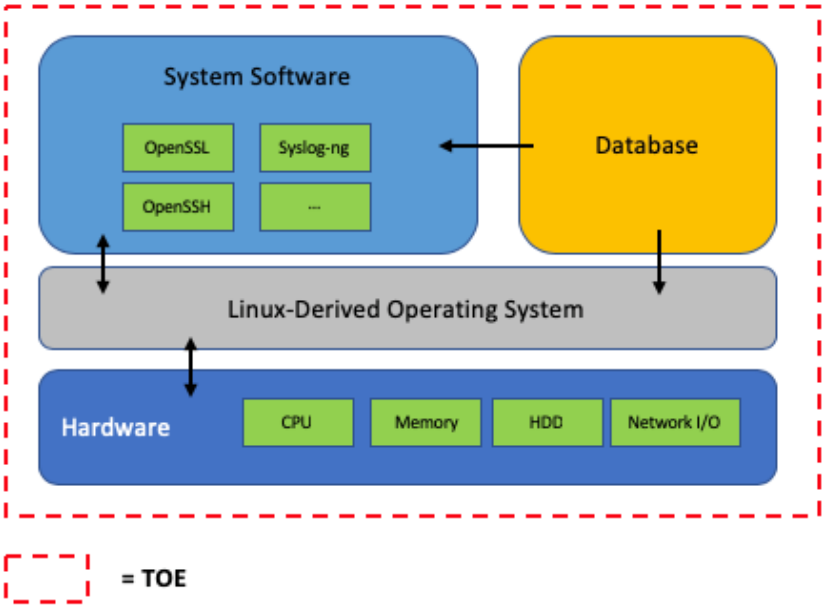

Figure 3 - TOE Architecture

2.2.1 Physical Boundaries

The TOE consists of the following components:

- Palo Alto Networks WF-500 hardware appliance
- WildFire 8.1.11: The software component that runs on the appliance

The WF-500 appliance includes the following ports:

- (Qty. 1) RJ-45 10/100/1000 management port used for managing the device and for data traffic
- (Qty. 3) RJ-45 10/100/1000 ports used for data traffic
- (Qty. 1) Graphics Port (reserved for future use)
- (Qty. 1) UID
- (Qty. 1) DB-9 serial port for console access (disabled in FIPS-CC mode)
- (Qty. 2) Power supplies
- (Qty. 4) USB ports (reserved for future use; disabled in FIPS-CC mode)

The operational environment includes the following:

- Syslog server
- Another Wildfire appliance for High Availability (HA)
- Palo Alto Networks Firewall or Panorama appliances
- Workstation
    - SSHv2 client

Table 1 TOE Platform

Product Identification	Illustration	Description
WF-500		The appliance detects unknown threats through a combination of multiple complementary analysis techniques. Processor: Intel Xeon E5-2620

2.2.2 Logical Boundaries

This section summarizes the security function provided by the TOE:

- Security audit
- Cryptographic support
- Identification and authentication
- Security Management
- Protection of the TSF
- TOE access
- Trusted path/channels

2.2.2.1 Security Audit

The TOE is designed to be able to generate logs for a variety of security relevant events including the events specified in NDcPP. The TOE can be configured to store the logs locally or can be configured to send the logs to a designated external log server.

2.2.2.2 Cryptographic Support

The TOE implements NIST validated cryptographic algorithms that provide key management, random bit generation, encryption/decryption, digital signature and cryptographic hashing and keyed-hash message authentication features in support of cryptographic protocols such as IPsec, TLS, and SSH. In order to utilize these features, the TOE must be configured in FIPS-CC mode.

WF-500 with WildFire 8.1.11 covered by the following CAVP certificates:

- AES – Cert. #5890
- CVL – Cert. #2119
- DRBG – Cert. #2451
- DSA – Cert. #1485
- ECDSA – Cert. #1570
- HMAC – Cert. #3865
- RSA – Cert. #3086
- SHS – Cert. #4641

2.2.2.3 Identification and Authentication

The TOE requires that all users that access the TOE be successfully identified and authenticated before they can have access to any security functions that are available in the TOE. The TOE offers functions through connections using SSH for administrators.

The TOE supports the local definition and authentication of administrators with username, password, SSH keys, and role that it uses to authenticate the operator. These items are associated with an operator and an authorized role for access to the TOE.

2.2.2.4 Security Management

The TOE provides access to the security management features using the CLI. CLI commands are transmitted over SSH for both local and remote connections. Security management commands are limited to administrators and only available after the operator has successfully authenticated themselves to the TOE. The TOE provides access to these services via direct RJ-45 Ethernet connection and remotely using an SSHv2 client. The product also includes a console port, but once FIPS-CC mode is enabled, the console port is disabled.

2.2.2.5 Protection of the TSF

The TOE implements features designed to protect itself, and to ensure the reliability and integrity of its security functions.

Stored passwords and cryptographic keys are protected so that unauthorized access does not result in sensitive data being lost, and the TOE also contains various self-tests so that it can detect if there are any errors with the system or if malicious activity has occurred. The TOE provides its own timing mechanism to ensure that reliable time information is present. The TOE uses digital signature mechanisms when performing trusted updates to ensure installation of software is valid and authenticated properly.

2.2.2.6 TOE Access

The TOE provides the ability for both TOE and user-initiated locking of the interactive sessions for the TOE termination of an interactive session after a period of inactivity is observed. Additionally, the TOE is able to display an advisory message regarding unauthorized use of the TOE before establishing a user session.

2.2.2.7 Trusted Path/Channels

The TOE protects interactive communication with remote administrators using SSH, and also protects communication between itself and other WildFire devices using IPsec. Communication with other devices and services (such as a Syslog server) are protected using TLS.

2.3 TOE Documentation

Palo Alto Networks, Inc. has several documents that provide operators with information regarding the installation, and the included security features.

For WildFire 8.1.11, these documents include the following:

- Palo Alto Networks Common Criteria Evaluated Configuration Guide (CCECG) for WildFire v8.1, Version 1.9, November 15, 2019
- Palo Alto Networks WildFire Administrator Guide Version 8.1, Last Revised: December 5, 2018
- WF-500 WildFire Appliance Hardware Reference Guide, February 29, 2016

3. Security Problem Definition

This security target includes by reference the Security Problem Definition (composed of organizational policies, threat statements, and assumption) from [NDcPP].

In general, the [NDcPP] has presented a Security Problem Definition appropriate for network infrastructure devices, such as firewalls, and as such is applicable to the Palo Alto TOE. NOTE: A.COMPONENTS_RUNNING is not applicable because this is not a distributed TOE.

4. Security Objectives

Like the Security Problem Definition, this security target includes by reference the Security Objectives from the [NDcPP]. The security objectives for the operational environment are reproduced below, since these objectives characterize technical and procedural measures each consumer must implement in their operational environment. NOTE: OE.COMPONENTS_RUNNING is not applicable because this is not a distributed TOE.

4.1 Security Objectives for the Operational 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.
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
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.TRUSTED_ADMIN	Security administrators are trusted to follow and apply all guidance documentation in a trusted manner. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) area assumed to monitor the revocation status of all certificates in the TOE's trust store in case such certificate can no longer be trusted.
OR.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.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.

5. IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the following Protection Profiles (PP):

- *collaborative Protection Profile for Network Devices, Version 2.1, 24 September 2018* [NDcPP],

As a result, refinements and operations already performed in that PP are not identified (e.g., highlighted) here, rather the requirements have been copied from that PP and any residual operations have been completed herein. Of particular note, the [NDcPP] made a number of refinements and completed some of the SFR operations defined in the CC and that PP should be consulted to identify those changes if necessary.

The SARs are the set of SARs specified in [NDcPP].

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from the [NDcPP]. The [NDcPP] defines all the extended SFRs (*_EXT.1) and since they are not redefined in this ST, the [NDcPP] should be consulted for more information in regard to those CC extensions.

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the TOE.

Table 2 TOE Security Functional Components

Requirement Class	Requirement Component
FAU: Security Audit	FAU_GEN.1: Audit Data Generation
	FAU_GEN.2: User Identity Association
	FAU_STG_EXT.1: Protected Audit Event Storage
FCS: Cryptographic Support	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_IPSEC_EXT.1: IPsec Protocol
	FCS_RBG_EXT.1: Random Bit Generation
	FCS_SSHS_EXT.1: SSH Server Protocol
	FCS_TLSC_EXT.1: TLS Client Protocol
	FCS_TLSC_EXT.2: TLS Client Protocol with Authentication
	FCS_TLSS_EXT.1: TLS Server Protocol
	FCS_TLSS_EXT.2: TLS Server Protocol with Mutual Authentication
FIA: Identification and Authentication	FIA_AFL.1: Authentication Failure Management
	FIA_PMG_EXT.1: Password Management
	FIA_UAU_EXT.2: Password-based Authentication Mechanism
	FIA_UAU.7: Protected Authentication Feedback
	FIA_UIA_EXT.1: User Identification and Authentication
	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: Security Management	FMT_MOF.1/ManualUpdate: Management of Security Functions Behavior
	FMT_MTD.1/CoreData: Management of TSF Data
	FMT_SMF.1: Specification of Management Functions
	FMT_SMR.2: Restrictions on Security Roles
FPT: Protection of the TSF	FPT_APW_EXT.1: Protection of Administrator Passwords

Requirement Class	Requirement Component
	FPT_SKP_EXT.1: Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
	FPT_STM_EXT.1: Reliable Time Stamps
	FPT_TST_EXT.1: TSF Testing
	FPT_TUD_EXT.1: Trusted Update
FTA: TOE Access	FTA_SSL_EXT.1: TSF-initiated Session Locking
	FTA_SSL.3: TSF-initiated Termination
	FTA_SSL.4: User-initiated Termination
	FTA_TAB.1: Default TOE Access Banners
FTP: Trusted Path/Channels	FTP_ITC.1: Inter-TSF Trusted channel
	FTP_TRP.1/Admin: Trusted Path

5.2.1 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; and
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 3.***

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 3**.

Table 3 Auditable 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.
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure.
FCS_RBG_EXT.1	None.	None.
FCS_SSHS_EXT.1	Failure to establish an SSH session.	Reason for failure.
FCS_TLSC_EXT.1	Failure to establish a TLS session.	Reason for failure.
FCS_TLSC_EXT.2	Failure to establish a TLS session.	Reason for failure.
FCS_TLSS_EXT.1	Failure to establish a TLS session.	Reason for failure.
FCS_TLSS_EXT.2	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_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g. IP address).
FIA_UAU.7	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g. IP address).
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.
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_MTD.1/CoreData	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_SKP_EXT.1	None.	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).
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure).	None.
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.
FTP_ITC.1	Initiation of the trusted channel.	

Requirement	Auditable Events	Additional Audit Record Contents
	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. Failures of the trusted path functions.	None.

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.

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

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

FAU_STG_EXT.1.3 The TSF shall [***[drop new audit data until the TOE is rebooted]***] when the local storage space for audit data is full.

5.2.2 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, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;***
- ***FFC schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1***
- ***FFC Schemes using Diffie-Hellman group 14 that meet the following: RFC 3526, Section 3***

] ~~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*:* [

- ***RSA-based key establishment schemes that meet the following: RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 8017, "Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.1";***
- ***Elliptic curve-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***
- ***Finite field-based key establishment schemes that meets the following: NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography";***
- ***Key establishment scheme using Diffie-Hellman group 14 that meets the following: RFC 3526, Section 3***

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

Application Note*: This SFR was modified based on TD0402.*

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** [**a pseudo-random pattern using the TSF's RBG**]*];
- *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 [[three or more times] overwrite consisting of [using a different alternating pattern]];***
  - ***instructs a part of the TSF to destroy the abstraction that represents the key***]

that meets the following*: No Standard.*

Application Note*: The TOE does not store plain text keys in non-volatile storage aside from a KEK, which is used to encrypt all stored key data. NIAP TRRT 241 response stated: "The TRRT does not see the need to modify the requirement. If the TOE does not store plaintext keys in one type of memory, that portion of the requirement is met. A statement in the TSS that plaintext keys are not stored in a specific type of memory is sufficient."*

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, 192 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, 3072 bits],***

- ***Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256 bits, 384 bits, 521 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 RSASSAPKCS1v1_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, P-521]; 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 cryptographic key sizes [assignment: cryptographic key sizes]~~ 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, HMAC-SHA-512***] and cryptographic key sizes [***160, 256, 384, 512 bits***] **and message digest sizes [*160, 256, 384, 512*] bits** that meet the following: *ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*.

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, AES-CBC-256 (specified in RFC 3602)***] together with a Secure Hash Algorithm (SHA)-based HMAC [***HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512***] and [***AES-GCM-128, AES-GCM-256 (specified in RFC 4106)***].

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 an Security Administrator based on [***
  - ***length of time, where the time values can be configured within [*1 to 65535*] 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 to 65535] 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 [***224 (for DH Group 14), 256 (for DH Group 19), 384 (for DH Group 20)***] 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***

] .

FCS_IPSEC_EXT.1.11 The TSF shall ensure that IKE protocols implement DH Group(s) [***14 (2048-bit MODP), 19 (256-bit Random ECP), 20 (384-bit Random ECP)***].

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 [***no other method***].

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: [***SAN: IP address, SAN: Fully Qualified Domain Name (FQDN)***] and [***no other reference identifier type***].

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] hardware-based noise source***] with 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 that complies with RFC(s) [***4251, 4252, 4253, 4254, 4344, 5656, 6668***].
Application Note*: This SFR was modified based on TD0398.*

FCS_SSHS_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following 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, aes128-gcm@openssh.com, aes256-gcm@openssh.com***].

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

FCS_SSHS_EXT.1.7 The TSF shall ensure that [***diffie-hellman-group14-sha1, ecdh-sha2-nistp256***] and [***ecdh-sha2-nistp384, ecdh-sha2-nistp521***] 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 no more than one gigabyte of transmitted data. After either of the thresholds are reached a rekey needs to be performed.

FCS_TLSC_EXT.1 - TLS Client Protocol

FCS_TLSC_EXT.1.1 The TSF shall implement [***TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)***] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[

- ***TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***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_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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246***
- ***TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***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***
- ***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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289***

].

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifiers of the following types: [***identifiers defined in RFC 6125, IPv4 address in CN or SAN, IPv4 address in SAN***] are matched to reference identifiers.

Application Note*: This SFR was modified based on TD0452.*

FCS_TLSC_EXT.1.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [

- ***Not implement any administrator override mechanism***

].

FCS_TLSC_EXT.1.4 The TSF shall [**p*resent the Supported Elliptic Curves Extension with the following NIST curves* [*secp256r1, secp384r1, secp521r1*] *and no other curves***]** in the Client Hello.

FCS_TLSC_EXT.2 - TLS Client Protocol with authentication

FCS_TLSC_EXT.2.1 The TSF shall implement [***TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)***] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[

- ***TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***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_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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246***
- ***TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***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***
- ***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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289***
- ***TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289***

].

FCS_TLSC_EXT.2.2 The TSF shall verify that the presented identifiers of the following types: [***identifiers defined in RFC 6125, IPv4 address in CN or SAN, IPv4 address in SAN***] are matched to reference identifiers.

Application Note*: This SFR was modified based on TD0452.*

FCS_TLSC_EXT.2.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [

- **Not implement any administrator override mechanism**

].

FCS_TLSC_EXT.2.4 The TSF shall [***present the Supported Elliptic Curves Extension with the following NIST curves: [secp256r1, secp384r1, secp521r1] and no other curves***] in the Client Hello.

FCS_TLSC_EXT.2.5 The TSF shall support mutual authentication using X.509v3 certificates.

FCS_TLSS_EXT.1 - TLS Server Protocol

FCS_TLSS_EXT.1.1 The TSF shall implement [**TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)**] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[

- ***TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246***
- ***TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***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 **[** ***none*****]**.

FCS_TLSS_EXT.1.3 The TSF shall [***perform RSA key establishment with key size [2048 bits, 3072 bits, 4096 bits]; generate EC Diffie-Hellman parameters over NIST curves [secp256r1, secp384r1] and no other curves; generate Diffie-Hellman parameters of size [2048 bits]***].

FCS_TLSS_EXT.2 - TLS Server Protocol with mutual authentication

FCS_TLSS_EXT.2.1 The TSF shall implement [***TLS 1.2 (RFC 5246), TLS 1.1 (RFC 4346)***] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[

- ***TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268***
- ***TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268***
- ***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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246***
- ***TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246***
- ***TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246***
- ***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.2.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, and [***none***].

FCS_TLSS_EXT.2.3 The TSF shall [***perform RSA key establishment with key size [2048 bits, 3072 bits]; generate EC Diffie-Hellman parameters over NIST curves [secp256r1, secp384r1] and no other curves; generate Diffie-Hellman parameters of size [2048 bits]***].

FCS_TLSS_EXT.2.4 The TSF shall support mutual authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.5 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server certificate is invalid. The TSF shall also [

- ***Not implement any administrator override mechanism***

].

FCS_TLSS_EXT.2.6 The TSF shall not establish a trusted channel if the distinguished name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match the expected identifier for the peer.

5.2.3 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 [unlock] is taken by an Administrator; 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***].

Application Note*: This SFR was modified based on TD0408.*

FIA_PMG_EXT.1 – Password management

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

1. Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [***"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [" ' ", "+", ",", "-", ".", "/", ":", ";", "<", "=", ">", "[", "\", "]", "_", "`", "{", "}", and "~"]***];
2. Minimum password length shall be configurable to between [*6*] and [*31*] characters.

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_UAU_EXT.2 – Extended: Password-based authentication mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local [***password-based, SSH public key-based***, ***[no other authentication mechanism(s)]***] authentication mechanism to perform local administrative user authentication.

Application Note*: This SFR was modified based on TD0408.*

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;
- **[*ICMP*]**.

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_X509_EXT.1 – 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 certificate path validation **supporting a minimum path length of three certificates.**
- The certificate path must terminate with a trusted CA certificate.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag 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, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5**]
- 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, TLS*],** 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 [***allow the Administrator to choose whether to accept the certificate in these cases, not accept the certificate***].

FIA_X509_EXT.3 – X.509 Certificate Requests

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

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

5.2.4 Security Management (FMT)

FMT_MOF.1.1/ManualUpdate - Management of Security Functions Behaviour

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to enable the functions *to perform manual update 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_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 configure the list of TOE-provided services available before an entity is identified and authenticated, as specified in FIA_UIA_EXT.1;***
- ***Ability to configure the cryptographic functionality;***
- ***Ability to re-enable an Administrator account;***
- ***Ability to set the time which is used for time-stamps;***
- ***Ability to generate, import, or delete X.509v3 certificates along with embedded key pairs;***
- ***Ability to configure the lifetime for IPsec SAs;***

].

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.

5.2.5 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 key, and private keys.

FPT_APW_EXT.1 – Protection of Administrator Passwords

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

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext 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), at the request of the authorised user, at the conditions [RSA/ECDSA key generation, loading firmware, use of DRBG or NDRNG]*]** to demonstrate the correct operation of the TSF: [
- ***AES Encrypt Known Answer Test***
- ***AES Decrypt Known Answer Test***
- ***AES GCM Encrypt Known Answer Test***
- ***AES GCM Decrypt Known Answer Test***
- ***AES CCM Encrypt Known Answer Test***
- ***AES CCM Decrypt Known Answer Test***
- ***RSA Sign Known Answer Test***
- ***RSA Verify Known Answer Test***
- ***RSA Encrypt Known Answer Test***
- ***RSA Decrypt Known Answer Test***
- ***ECDSA Sign Known Answer Test***
- ***ECDSA Verify Known Answer Test***
- ***HMAC-SHA-1 Known Answer Test***
- ***HMAC-SHA-256 Known Answer Test***
- ***HMAC-SHA-384 Known Answer Test***
- ***HMAC-SHA-512 Known Answer Test***
- ***SHA-1 Known Answer Test***
- ***SHA-256 Known Answer Test***
- ***SHA-384 Known Answer Test***

- ***SHA-512 Known Answer Test***
- ***DRBG SP800-90A Known Answer Tests***
- ***SP 800-90A Section 11.3 Health Tests***
- ***DH Known Answer Test***
- ***ECDH Known Answer Test***
- ***Firmware Integrity Test***
- ***Conditional Self-Tests***
  - ***Continuous Random Number Generator (RNG) test – Performed on NDRNG and DRBG***
  - ***RSA Pairwise Consistency Test***
  - ***ECDSA Pairwise Consistency Test***
  - ***Firmware Load Test – Verify firmware signatures using RSA 2048 with SHA-256 at time of load***

]

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 [***no other TOE firmware/software version***].

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 mechanism***] 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 [***allow the Security Administrator to set the time***].

5.2.6 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.

5.2.7 Trusted Path/Channels (FTP)

FTP_ITC.1 – Inter TSF Trusted Channel

FTP_ITC.1.1 The TSF shall **be capable of using** [***TLS, IPsec***] **to** provide a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server**, [***WildFire, Firewall, Panorama***] 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 [

- ***transmitting audit records to an audit server using TLS***
- ***communicating with Palo Alto Networks firewall, WildFire appliances, Panorama***].

FTP_TRP.1/Admin– Trusted path

FTP_TRP.1.1/Admin The TSF shall **be capable of using** [***SSH***] **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 administrative actions.*

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference to [NDcPP]

Table 4 Assurance Components

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

Consequently, the assurance activities specified in the following Supporting Documents apply to the TOE evaluation:

- Supporting Document Mandatory Technical Document: Evaluation Activities for Network Device cPP, 24-September-2018, Version 2.1

6. TOE Summary Specification

This chapter describes the security functions:

- Security audit
- Cryptographic support
- Identification and authentication
- Security management
- Protection of the TSF
- TOE access
- Trusted path/channels

6.1 Security Audit

FAU_GEN.1	The TOE is designed to be able to generate log records for security-relevant events as they occur. The events that can cause an audit record to be logged include starting and stopping the audit function (also startup and shutdown of system), any use of an administrator command via the CLI, as well as all of the events identified in Table 3 (which corresponds to the audit events specified in the [NDcPP]. All log records include the following contents: date/time, event type, user ID (i.e., username, IP address) or component (i.e., ssh, syslog), and description of the event including success or failure. For user-initiated actions, the User ID is included in the log records. For cryptographic key operations, the key name—or certificate name if the key is embedded in certificate or certificate request—is also logged.
FAU_GEN.2	The TOE identifies the responsible user for each event based on the specific username and/or network entity (identified by source IP address) that caused the event.
FAU_STG_EXT.1	The audit trail generated by the TOE comprises several logs, which are locally stored in the TOE file system on the hard disk: • Configuration logs—include events such as when an administrator configures the device, user management, cryptographic functions, audit functions (e.g., enable syslog over TLS connection), and when an administrator configures which events gets audited. • System logs—include events such as user login and logout, session establishment, termination, and failures. The TOE stores the audit records locally and protects them from unauthorized deletion by allowing only users in the pre-defined Audit Administrator role to access the audit trail with delete privileges. The pre-defined Audit Administrator role is part of the Security Administrator role as defined by the [NDcPP]. The TOE does not provide an interface where a user can modify the audit records, thus it prevents modification to the audit records. When a log reaches the maximum size, the TOE starts overwriting the oldest log entries with the new log entries. When the log reaches the maximum size, new audit data is dropped until the Administrator reboots the TOE to clear the old log entries. Maximum disk space is dependent on the customer's installation as it depends on the number of hard drives installed on the system. The TOE can be configured to send generated audit records to an external Syslog server in real-time using TLSv1.2. When configured to send audit records to a syslog server, audit records are also written to the external syslog as they are written locally to the internal logs.

6.2 Cryptographic Support

The TOE includes NIST-validated cryptographic algorithms that provide support for the cryptographic functions.

The functions in this section have been certified in accordance with the identified standards that cover the following required components:

- 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_IPSEC_EXT.1 IPsec Protocol
- FCS_RBG_EXT.1: Random Bit Generation
- FCS_ SSHS_EXT.1: SSH Server Protocol
- FCS_TLSC_EXT.1: TLS Client Protocol
- FCS_TLSC_EXT.2: TLS Client Protocol with Authentication
- FCS_TLSS_EXT.1 TLS Server Protocol
- FCS_TLSS_EXT.2 TLS Server Protocol with Mutual Authentication

Table 5 Cryptographic Functions

Functions	Standards	Certificates
Asymmetric key generation		
FFC key pair generation (key size 2048 bits)	FIPS PUB 186-4	RSA #3086 ECDSA #1570 DSA #1485
ECC key pair generation (NIST curves P-256, P-384, P-521)	FIPS PUB 186-4	
RSA key generation (key size 2048, 3072, 4096 bits)	FIPS PUB 186-4	
FFC Schemes using Diffie-Hellman Group 14	RFC 3526, Section 3	N/A (Vendor Assertion)
Cryptographic Key Generation (for IKE Peer Authentication)		
RSA key generation (key size 2048, 3072, 4096 bits)	FIPS PUB 186-4	RSA #3086 ECDSA #1570
ECDSA key pair generation (NIST curves P-256, P-384, P-521)	FIPS PUB 186-4	
Cryptographic Key Establishment		
RSA based key establishment	RSAES-PKCS1-v1_5	N/A (Vendor Assertion)
ECDSA based key establishment	NIST SP 800-56A	KAS Component #2119
FFC based key establishment	NIST SP 800-56A	
Key establishment scheme using Diffie-Hellman Group 14	RFC 3526, Section 3	N/A (Vendor Assertion)
AES Data Encryption/Decryption		
AES CBC, GCM (128, 192, 256 bits)	AES as specified in ISO 18033-3 CBC as specified in ISO 10116 GCM as specified in ISO 19772	AES #5890
Signature Generation and Verification		

Functions	Standards	Certificates
RSA Digital Signature Algorithm (rDSA) (modulus 2048, 3072)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSAPKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	RSA #3086
ECDSA (NIST curves P-256, P-384, and P-521)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" P-256, P-384, P-521 ISO/IEC 14888-3, Section 6.4	ECDSA #1570
Cryptographic hashing		
SHA-1, SHA-256, SHA-384 and SHA-512 (digest sizes 160, 256, 384 and 512 bits)	ISO/IEC 10118-3:2004	SHS #4641
Keyed-hash message authentication		
• HMAC-SHA-1 (block size 512 bits, key size 160 bits and digest size 160 bits) • HMAC-SHA-256 (block size 512 bits, key Size 256 bits and digest size 256 bits) • HMAC-SHA-384 (block size 1024 bits, key Size 384 bits and digest size 384 bits) • HMAC-SHA-512 (block size 1024 bits, key Size 512 bits and digest size 512 bits)	ISO/IEC 9797-2:2011	HMAC #3865
Random bit generation		
CTR_DRBG (AES) from a hardware-based noise source with one independent software-based noise source of 256 bits of non-determinism	ISO/IEC 18031:2011	DRBG #2451

The TOE implements the ISO/IEC 18031:2011 Deterministic Random Bit Generator (DRBG) based on the AES 256 block cipher in counter mode (CTR_DRBG(AES)). The TOE instantiates the DRBG with maximum security strength, obtaining the 256 bits of entropy from a proprietary hardware entropy source to seed the DRBG. The entropy sources are described in the proprietary Entropy Design document.

The TOE generates asymmetric cryptographic keys used for key establishment in accordance with FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes, FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECC schemes, FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.1 for FFC schemes, and Diffie-Hellman Group 14 according to RFC 3526, section 3.

While the TOE generally fulfills all of the FIPS PUB 186-4 requirements without extensions, the following table specifically identifies the "should", "should not", and "shall not" conditions from the publication along with an indication of whether the TOE conforms to those conditions with deviations rationalized. Key generation is among the identified sections.

Table 6 FIPS 186-4 Conformance

FIPS PUB 186-4	"should", "should not", or "shall not"	Implemented accordingly?	Rationale for deviation
FIPS PUB 186-4 Appendix B.1			
B.1.1	should	Yes	N/A
B.1.2	should	Yes	N/A
FIPS PUB 186-4 Appendix B.3			
B.3.1	shall not	Yes	N/A
FIPS PUB 186-4 Appendix B.4			
B.4.1	should	Yes	N/A
B.4.2	should	Yes	N/A

The TOE performs cryptographic RSA-based key establishment in accordance with RSAES-PKCS1-v1_5 as specified in Section 7.2 of RFC 3447, NIST Special Publication 800-56A for elliptic curve-based key establishment schemes, and NIST Special Publication 800-56A for finite field-based key establishment schemes. The TOE acts as both a sender and as a recipient for RSA-based, elliptic curve-based, and finite field-based key establishment schemes.

Table 7 Private Keys and CSPs

CSP #	Key/CSP	Description
1	RSA/ECDSA Private keys	Private keys support establishment of TLS, IPsec/IKE, and SSH host authentication. (RSA 2048, 3072 or 4096 bits; ECDSA P-256, P-384, or P-521)
2	TLS ECDHE/DHE Private Components	Diffie-Hellman private component (DH (L=2048, N >=224), (ECDHE P-256, P-384, P-521)
3	TLS Pre-Master Secret	Secret value used to derive TLS session keys.
4	TLS Encryption keys	AES (128 or 256 bit; CBC or GCM) session keys used in TLS connections.
5	TLS HMAC keys	HMAC session keys (HMAC-SHA-1/SHA-256/SHA-384) used in TLS connections.
6	SSH ECDH/DH Private Components	ECDH and Diffie-Hellman private component (DH 2048 bits, ECDH P-256, P-384, P-521)
7	SSH Session Encryption key	AES session keys used in SSH connections. (128 or 256 bits; CBC, CTR) (128 or 256 bits: GCM)
8	SSH Session Authentication key	HMAC session keys used in SSH connections. (HMAC-SHA-1, HMAC-SHA2-256, HMAC-SHA2-512)
9	CO, User Password	Password for operator authentication (minimum 6 characters).

CSP #	Key/CSP	Description
10	DRBG Seed and State	AES CTR DRBG used in the generation of random values.
11	SNMPv3 Secrets	SNMPv3 Authentication and Privacy Secrets
12	SNMPv3 Keys	AES Session and HMAC-SHA-1 Authentication Keys
13	IPsec/IKE DH Private Components	Diffie-Hellman (Group 14) private components
14	IPsec/IKE ECDH Private Components	EC Diffie-Hellman (Group 19, 20) private components
15	IPsec/IKE Session Keys	Encryption keys for session (128 or 256 bits: AES GCM or CBC)
16	IPsec/IKE Authentication Keys	HMAC keys for authentication (HMAC-SHA-256/384/512)
17	Firmware Content Encryption Key	AES-256-CBC Key Used to encrypt/decrypt firmware, software, and sensitive content.

The TOE performs a key error detection check on each internal, intermediate transfer of a key. The TOE stores persistent secret and private keys in encrypted form (AES encrypted) when not in use. The KEK is the Firmware Content Encryption Key (also known as the Master Key). The TOE zeroizes non-persistent cryptographic keys as soon as their associated session has terminated. In addition, the TOE recognizes when a private key expires and promptly zeroizes the key on expiration. The TOE does not permit expired private signature keys to be archived.

Private cryptographic keys, plaintext cryptographic keys, and all other critical security parameters stored in intermediate locations in volatile memory for the purposes of transferring the key/critical security parameters (CSPs) to another location are zeroized immediately following the transfer. Zeroization is done by overwriting the storage location with a random pattern, followed by a read-verify. Note that plaintext cryptographic keys (e.g., TLS encryption keys, SSH session keys, IKE/IPsec session keys) and CSPs (e.g., TLS Pre-Master secret, ECDHE/DHE private components) are only ever stored in volatile memory.

For non-volatile memory, the only plaintext key that is stored is the KEK. When a new KEK is generated, the old KEK is destroyed via key store APIs that overwrites the old KEK. The KEK is erased when the administrator initiates the zeroization function, which overwrites the KEK three or more times using a differing alternating pattern. Destruction of all encrypted stored keys is accomplished indirectly through destruction of the KEK that encrypted them.

The TOE can be configured as a TLS server for mutual certificate-based authentication for secure connections. The key agreement parameters of the server key exchange message consist of the key establishment parameters generated by the TOE: Diffie-Hellman parameters with a key size 2048 bits, ECDSA implementing NIST curves secp256r1, secp384r1, and secp521r1. The TOE denies connections from clients requesting connections using SSL 2.0, SSL 3.0, or TLS 1.0 and shall not establish a trusted channel if the distinguished name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match the expected identifier for the peer.

The TOE can be configured as a TLS client for mutual certificate-based authentication for secure communications. The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125 and only establishes a trusted channel if the peer certificate is valid. The TOE determines certificate validity by verifying the identifier, certificate path, the expiration date, and the revocation status in accordance with RFC 5280. The TOE includes support for client-side certificates for TLS mutual authentication using X509v3 certificates. The TOE compares the external server's presented identifier to the reference identifier by matching the certificate Common Name (Subject), FQDN (hostname), IP address, User FQDN (email address). The TOE supports IP address reference identifiers and wildcards for peer authentication. Certificate pinning is not supported. The TOE presents

the Supported Elliptic Curves Extension in the Client Hello with the secp256r1, secp384r1, and secp521r1 NIST curves and is enabled by default.

The TOE can be configured as a TLS client for secure communication to an external audit server. The TOE verifies that the presented identifier matches the reference identifier according to RFC 6125 and only establishes a trusted channel if the peer certificate is valid. The TOE compares the external server's presented identifier to the reference identifier by matching the certificate Common Name (Subject), FQDN (hostname), IP address, User FQDN (email address). The CN field is composed of IPv4 addresses that follow the rules described in RFC 3986. The TOE supports IP address reference identifiers and wildcards for peer authentication. Certificate pinning is not supported. The TOE presents the Supported Elliptic Curves Extension in the Client Hello with the secp256r1, secp384r1, and secp521r1 NIST curves and is enabled by default.

The TOE implements TLS 1.2 (RFC 5246) and TLS 1.1 (RFC4346).

TOE (as TLS client) to syslog server (same for mutual authentication) supports the following cipher suites (TLSv1.2):

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- 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_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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- 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
- 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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

TOE (as TLS server) connection to firewall (same for mutual authentication) supports TLSv1.1 or TLSv1.2:

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- 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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- 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

TOE (as TLS client) to Panorama (same for mutual authentication) supports the following cipher suites (TLSv1.1 or TLSv1.2):

- TLS_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268
- TLS_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 3268

- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268
- 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_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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246
- TLS_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_128_CBC_ SHA256 as defined in RFC 5246
- TLS_DHE_RSA_WITH_AES_256_CBC_ SHA256 as defined in RFC 5246
- 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
- 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_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289

The TOE includes an implementation of IPsec in accordance with RFC 4301. The primary cryptographic algorithms used by the TOE include AES-CBC-128, AES-CBC-256 (both specified by RFC 3602); and AES-GCM-128, AES-GCM-256 as specified in RFC 4106; The TOE only supports IKEv2 as defined in RFCs 5996 (with mandatory support for NAT traversal as specified in section 2.23), and 4868 for hash functions. IKEv2 supports AES-128/256 CBC while IPsec supports AES-128/256 CBC or GCM. The administrator is instructed to ensure that the size of key used for ESP must be less than or equal to the key size used to protect the IKE payload. Tunnel mode is supported, and uses the SHA-based HMAC algorithms as specified in FCS_COP.1/KeyedHash Cryptographic Operations (Keyed Hash Algorithm). The TOE supports Diffie-Hellman Group 14, 19, and 20 for the key exchange used for IKE and IPsec. In order to create a connection, the TOE performs peer authentication using X.509v3 certificates that are either RSA or ECDSA. An administrator of the TOE specifies through the configuration what identifier in the presented peer certificate will be checked, which includes either the IP address or Fully Qualified Domain Name (FQDN). The TOE does not support pre-shared keys. For certificates, they can either be generated by the TOE using the CLI, or an administrator can import their desired certificates into the TOE.

When the administrator configures the IPsec setup between the WF-500 appliances, the TOE requires that a specific physical port is defined for the connection to these peer devices that is separate than other ports such as the management port. After the configuration is created, the TOE uses this policy to determine which packets require the PROTECT functionality (encrypting the packets), while others have actions such as BYPASS or DISCARD for inbound/outbound packets. The TOE automatically processes these packets to their appropriate location given the configuration set by the administrator. The TOE has a strict hierarchy such that higher-ranked rules always apply, which ensures that there is no conflict in the processing order.

The lifetimes for IKE and IPsec can be configured by the administrator of the TOE via the CLI. For IKEv2, the lifetime can be configured (in hours/minutes) while the IPsec lifetime allows for configuration based on time (hours/minutes) or number of bytes (kB/MB/GB).

Using the random bit generator (AES-CTR DRBG), the TOE generates secret values (x) for the IKE Diffie-Hellman key exchange with a length of 224, 256, and 384 bits depending on the key exchange selected (DH Group 14, 19, or 20). The TOE and IKE peer will negotiate the highest (most preferred) DH group that is supported by both parties. The nonces used in the IKE exchanges are generated by the TOE's random bit generator according to the security strength associated with the negotiated Diffie-Hellman group.

To set up IPsec between WF-500 appliances, an administrator shall use the CCECG reference document as it provides the applicable commands for the settings/options described above.

The TOE supports SSHv2 (compliant to RFCs 4251, 4252, 4253, 4254, 4344, 5656, 6668) with AES encryption/decryption algorithm (in CBC, CTR, or GCM mode) with key sizes of 128 or 256 bits. The TOE does

not support any other optional characteristics for encryption or public key algorithms. The TOE also supports HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512, aes128-gcm@openssh.com, and aes256-gcm@openssh.com for integrity. Both encryption and integrity algorithms are administrator-configurable and while 3DES, HMAC-MD5, diffie-hellman-group-1 are also supported, they are all disabled when FIPS-CC mode is enabled. Only the Approved encryption and integrity algorithms along with key exchange algorithms diffie-hellman-group14-sha1, ecdh-sha2-nistp256, ecdh-sha2-nistp384, and ecdh-sha2-nistp521 and authentication public-key algorithm ssh-rsa are permitted in the evaluated configuration. If the SSH client (in the operational environment) only supports non-Approved algorithms, the SSH connection will be rejected by the TOE.

The TOE uses OpenSSH implementation to support the SSHv2 connections. The authentication timeout period is 60 seconds allowing clients to retry only 4 times. In addition, both public-key (RSA) and password-based authentication can be configured with password-based being the default method. The SSH packets are limited to 256 Kbytes and any packet over that size will be dropped (i.e., not processed farther and buffer containing the packet will be freed). The TOE manages a tracking mechanism for each SSH session so that it can initiate a new key exchange (rekey) when either a configurable amount of data (10 – 4000 MBs) or time (10 – 3600 seconds) has passed, whichever threshold occurs first. In the evaluated configuration, the administrator should not configure the SSH data rekey threshold to be more than 1024 MBs.

6.3 Identification and Authentication

FIA_UIA_EXT.1, FIA_UAU_EXT.2, FIA_UAU.7

The TOE is designed to require users to be identified and authenticated before they can access any of the TOE functions. The only capabilities allowed prior to users authenticating are the display of an informative (login) banner and responding to ICMP request (e.g., ping or ICMP echo reply).

The TOE maintains user accounts which it uses to control access to the TOE. When creating a new user account, the administrator specifies a username, password, and a role. Only one role is specified in the user account per user. The administrator can also specify an SSH key to be used instead of a password.

The TOE uses the username and password or an SSH key to identify and authenticate the user when the user logs in via the CLI. The TOE does not echo passwords as they are entered, and the private keys are never transmitted. When accessing the CLI, the default authentication method is password. The administrators must configure public-key authentication which is supported for SSH sessions. It uses the role attribute to specify user permissions and control what the user can do with the CLI.

The administrator can logon to CLI by using a secure connection (SSHv2) from an SSH client. The TOE provides access to the CLI locally via direct RJ-45 Ethernet cable connection and remotely using an SSHv2 client. The administrator enters the IP address of the TOE and their username and password or their corresponding SSH key.

A logon successful note is provided when the correct credentials are used (username and password or SSH key) that matches a defined account on the TOE.

FIA_PMG_EXT.1

Passwords can be composed of upper and lower case letters, numbers and special characters “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(”, “)”, [“”, “+”, “,”, “-”, “.”, “/”, “:”, “;”, “<”, “=”, “>”, “[”, “\”, “]”, “_”, “`”, “{”, “}”, and “~”. There are no restrictions on any password field character sets. The minimum password length is configurable by the administrator up to a maximum length of 15 characters. In FIPS-CC mode, the minimum and maximum length is from 6 to 31 characters.

FIA_AFL.1

The TOE logs all unsuccessful authentication attempts in the system log, and tracks the number of failed attempts via internal counters. The TOE can be configured to lock a user or authorized IT entity out after a configurable number (1 – 10) of unsuccessful authentication attempts. The lock can be configured to last a specified amount of

time (0 – 60 minutes) during which providing the correct credentials will still not allow access (i.e. locked out). A setting of "0" will lock out the user indefinitely. In this case, the Administrator must unlock the locked user.

These settings can be configured for SSH remote administration connections but applies to password authentication only. Public-key authentication is not vulnerable to weak passwords that can be brute-forced. In the evaluated configuration, it is required that at least one administrator, preferably the Superuser role (predefined 'admin' account), is configured with public-key authentication for SSH to prevent a denial of service due to locked accounts. This account can then be used to unlock administrator accounts if all other accounts have been locked. The administrator can also wait until the lockout time has expired.

FIA_X509_EXT.1/Rev, FIA_X509_EXT.2, FIA_X509_EXT.3

The TOE uses X.509v3 certificates as defined by RFC 5280 to support authentication for IPsec and TLS connections. Public key infrastructure (PKI) credentials, such as Rivest, Shamir, and Adelman (RSA) keys and certificates are stored in the TOE's underlying file system on the appliance. Certificates and their associated private key are stored in a single container: the Certificate File. The PKCS#12 file consists of an Encrypted Private Key and X509 Certificate. By default, all the private keys are protected since they are always stored in encrypted format using AES-256. The physical security of the appliance (A.PHYSICAL_PROTECTION) protects the appliance and the certificates from being tampered with or deleted. In addition, the TOE identification and authentication security functions protect an unauthorized user from gaining access to the TOE.

The TOE supports Open Certificate Status Protocol (OCSP) and Certificate Revocation List (CRL) status verification for certificate profiles. If both are configured, the devices first try the OCSP method; if the OCSP server is unavailable, the devices use the CRL method (RFCs 5280 and 5759 compliance).

In the case of IPsec connections, the TOE only supports CRL (RFCs 5280 and 5759 compliance).

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

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

The TOE validates a certificate path by ensuring the presence of the basicConstraints extension is present and that the CA flag is set to TRUE for all CA certificates. The TOE forms a Certificate trust path by ensuring that the basic constraints are met, proper key usage parameters exist, the CA flag exists, performing a revocation check of each certificate in the path and performing the validity of the CA certificate. The TOE will not treat a certificate as a CA certificate if the basicConstraints extension is not present or the CA flag is not set to TRUE.

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

When establishing a TLS session, clients can use OCSP to check the revocation status of the authentication certificate. The authenticating client sends a request containing the serial number of the certificate to the OCSP responder (server). The responder searches the database of the certificate authority (CA) that issued the certificate and returns a response containing the status (good, revoked or unknown) to the client. The advantage of the OCSP method is that it can verify status in real-time, instead of depending on the issue frequency (hourly, daily, or weekly) of CRLs.

The TOE downloads and caches the last-issued CRL for every CA listed in the trusted CA list of the TOE. Caching only applies to validated certificates; if a TOE never validated a certificate, the TOE cache does not store the CRL for the issuing CA. Also, the cache only stores a CRL until it expires. The TOE supports CRLs only in Distinguished Encoding Rules (DER) or PEM format.

The TLS session for Syslog is blocked when the certificate status is unknown or cannot be determined. This is the default behavior for syslog connections, and cannot be changed. When configuring the TLS sessions for Firewalls and Panorama, or IPsec connections between WF-500 appliances, the administrator may configure the profile whether or not to block connections when certificate status is unknown or cannot be determined.

The authorized administrator may generate a certificate request as specified in RFC 2986 and provide the following information in the request: Common Name, Organization, and Country. The administrator may also import a certificate and private key into the TOE from an enterprise certificate authority or obtain a certificate from an external CA. When the administrators import a certificate based on the CSR, the TOE will check to make sure the certificate chain are present in the TOE. Otherwise, the TOE will reject the certificate and will not associate it with the CSR.

6.4 Security Management

FMT_MOF.1/ManualUpdate, FMT_MTD.1/CoreData

The TOE provides a CLI to support security management of the TOE. The CLI is accessible via direct connection to the management port on the device (local access), or remotely over SSHv2. The restricted role-based privileges enable only authorized administrators to configure the TOE functions such as updating the TOE, managing X.509v3 certificates in the trust store, and manipulating TSF data.

FMT_SMF.1

The security management functions provided by the TOE including, but not, limited to:

- 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 configure the list of TOE-provided services available before an entity is identified and authenticated;
- Ability to set the time which is used for time-stamps;
- Ability to configure the cryptographic functionality;
- Ability to re-enable an Administrator account;
- Ability to generate, import, or delete X.509v3 certificates along with embedded key pairs;
- Ability to configure the lifetime for IPsec SAs;

The CLI provides the administrator the ability to administer the TOE locally and remotely with all management functions. The management functions above can be configured when the administrator successfully authenticates into the TOE via the CLI. The local interface supports the use of a dedicated Ethernet port that only supports communication with a whitelisted local IP address. Regardless of whether the physical interface is local or remote, the logical interface used to manage the TOE is through SSH CLI.

FMT_SMR.2

The TOE controls user access to commands and resources based on user role. Users are given permission to access a set of commands and resources based on their user role. By default, the TOE has the following pre-defined administrator roles: Superuser and Superuser (Read-Only). The administrator role (Superuser) is considered the Security Administrator as defined in the [NDcPP] for the purposes of this ST. For example, a user with Superuser role can create, modify, or delete user accounts but users with Read-Only role cannot. All roles can administer the TOE both locally and remotely.

- **Superuser**—Full read-write access to WildFire services
- **Superuser (Read-Only)**—Read-only access to WildFire services

6.5 Protection of the TSF

FPT_SKP_EXT.1, FPT_APW_EXT.1

Certificates and their associated private key are stored in a single container: the Certificate File. The PKCS#12 file consists of an Encrypted Private Key and X509 Certificate. By default, all the private keys (including SSH private keys) are protected since they are always stored in encrypted format using AES-256. The TOE prevents the reading of all keys by encrypting them with a Master Key using AES-256. The TOE does not provide an interface to read the Master Key. The TOE is designed specifically to prevent access to locally-stored cryptographically protected passwords and does not disclose any keys stored in the TOE. The TOE protects the confidentiality of user passwords by hashing the passwords using SHA-256. The TOE does not offer any functions that will disclose to any users a stored cryptographic key or password.

FPT_TST_EXT.1

The TOE runs self-tests to ensure that the cryptographic capabilities of the TOE are intact, which include the list below. These tests are performed each time the module is powered on or when the administrator initiates it via CLI.

- AES Encrypt Known Answer Test
- AES Decrypt Known Answer Test
- AES GCM Encrypt Known Answer Test
- AES GCM Decrypt Known Answer Test
- AES CCM Encrypt Known Answer Test
- AES CCM Decrypt Known Answer Test
- RSA Sign Known Answer Test
- RSA Verify Known Answer Test
- RSA Encrypt Known Answer Test
- RSA Decrypt Known Answer Test
- ECDSA Sign Known Answer Test
- ECDSA Verify Known Answer Test
- DH Known Answer Test
- HMAC (HMAC-SHA-1/256/384/512) Known Answer Test
- SHA-1 Known Answer Test
- SHA-256 Known Answer Test
- SHA-384 Known Answer Test
- SHA-512 Known Answer Test
- DRBG Known Answer Test
- ECDH Known Answer Test
- SP 800-90A Section 11.3 Health Tests
- Firmware Integrity Test

The firmware integrity is verified with HMAC-SHA-256 and ECDSA P-256. If the calculate result does not equal the previously generated result, the integrity test shall fail.

A known-answer test involves 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. The TOE performs the following Conditional Self-Tests within the cryptographic module when the conditions specified for the tests occur:

Conditional Self-Tests

- Continuous Random Number Generator (RNG) test – Performed on NDRNG and DRBG
- RSA Pairwise Consistency Test
- ECDSA Pairwise Consistency Test
- Firmware Load Test – Verify firmware signatures using RSA 2048 with SHA-256 at time of load

The RNG continuous random number generator test is performed on each RNG and tests for failure to a constant value as follows:

1. If each call to a RNG produces blocks of n bits (where n > 15), the first n-bit block generated after power-up, initialization, or reset shall not be used, but shall be saved for comparison with the next n-bit block to be generated. Each subsequent generation of an n-bit block shall be compared with the previously generated block. The test shall fail if any two compared n-bit blocks are equal.

2. If each call to a RNG produces fewer than 16 bits, the first n bits generated after power-up, initialization, or reset (for some n > 15) shall not be used, but shall be saved for comparison with the next n generated bits. Each subsequent generation of n bits shall be compared with the previously generated n bits. The test fails if any two compared n-bit sequences are equal.

The TOE performs the following pair-wise consistency tests for public and private keys:

1. If the keys are used to perform an approved key transport method or encryption, then the public key shall encrypt a plaintext value. The resulting ciphertext value shall be compared to the original plaintext value. If the two values are equal, then the test shall fail. If the two values differ, then the private key shall be used to decrypt the ciphertext and the resulting value shall be compared to the original plaintext value. If the two values are not equal, the test shall fail.

2. If the keys are used to perform the calculation and verification of digital signatures, then the consistency of the keys shall be tested by the calculation and verification of a digital signature. If the digital signature cannot be verified, the test shall fail.

If a self-test fails, the TOE enters an error state and outputs an error indicator. The TOE does not perform any cryptographic operations while in the error state. All data output from the TOE is inhibited when an error state exists. Should one or more power-up self-tests fail, the module will reboot and enter a maintenance state in which the reason for the reboot can be determined.

The methods above are sufficient to ensure the correct functionality of the TSF as the self-tests encompass the cryptographic functionality and the integrity of the entire TOE software/firmware executable code.

FPT_TUD_EXT.1

Authorized administrators can query the current version of the TOE's software using the CLI with the command "show system info", and are able to receive updates from https://updates.paloaltonetworks.com using the command

"request system software check" followed by "request system software download <version>". There is no automatic installation of new software; an administrator must manually perform this update. The Palo Alto Networks Update Server supports TLS 1.2 and uses approved cipher suites to ensure that downloads from the server are protected, and are not tampered with in transit.

As an additional precaution, Palo Alto Networks has chosen to sign (using RSA-2048) and encrypt (using AES-256) all content that is downloaded to the TOE. If the TOE is not connected to the internet, the administrators can download the updates and upload it to the TOE (manual update).

When the TOE update package and its corresponding digital signature is downloaded or uploaded; the digital signature is checked automatically by the TOE by verifying the signature using the public key (corresponding to the RSA key used to create the signature). Palo Alto Networks manages the update server and guarantees that images are digitally signed. Public keys are stored and protected in the TOE's file system. If the signature is verified, the update is performed; otherwise the update is not performed.

FPT_STM_EXT.1

The TOE is a hardware appliance that includes a hardware-based real-time clock. The TOE's embedded OS manages the clock and exposes administrator clock-related functions such as set time. The clock is used for audit record time stamps, measuring session activity for termination, certificate validity checking, timing administrator lockout due to excessive failed authentication attempts, and for cryptographic operations based on time/date.

6.6 TOE Access

FTA_SSL_EXT.1, FTA_SSL.3

The TOE will enforce an administrator-defined inactivity timeout value after which the inactive, local or remote, session will be terminated regardless of authentication methods (e.g. password, public-key). The TOE can be configured by an administrator to set an interactive session timeout value (any integer value from 1 to 60 minutes). The function is disabled by default and the administrator must follow the CC AGD to configure the session idle timeout value.

A remote session that is inactive (i.e. no commands issued from the remote client) for the defined timeout value will be terminated. A local session that is similarly inactive for the defined timeout period will be terminated. The users will be required to re-enter their user ID and their password or perform public-key authentication (i.e. restart an SSH connection) in order to establish a new session once the session is terminated.

FTA_SSL.4

The TOE provides the function to logout (or terminate) both local and remote user sessions as directed by the user.

FTA_TAB.1

The TOE can be configured to display an administrator-defined advisory banner prior to authentication when accessing the TOE via a direct or remote connection to the management port in order to access the CLI (SSH).

6.7 Trusted Path/Channels

FTP_ITC.1

The TOE can be configured to send audit records to an external syslog server using TLS in real-time. The TOE permits the TSF to initiate communication with the syslog server or Panorama (TLS client) using the TLS trusted channel. It also permits the TSF to initiate communication with other WF-500 appliances using IPsec. The TOE (TLS server) can also communicate with the Palo Alto Networks firewalls via TLS. Mutual authentication is supported but must be configurable for all TLS channels.

The TOE communicates with its authorized entities over TLS only and all communication are sent over the trusted channel, including the TOE initial communication. The underlying TLS and IKE/IPsec algorithms are supported by CAVP-validated cryptographic mechanisms included in the TOE implementation.

FTP_TRP.1/Admin

The TOE provides SSH to provide a communication path between itself authorized remote administrators. Administrators can initiate a remote session that is secure using CAVP validated cryptographic operations, and all remote security management functions require the use of this security channel. In FIPS-CC mode, Telnet and HTTP are disabled permanently.

7 Protection Profile Claims

This ST is conformant to the [NDcPP].

8 Rationale

This security target includes by reference the [NDcPP] Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the [NDcPP] assumptions. Security functional requirements have been reproduced verbatim with the protection profile operations completed. Operations on the security requirements follow [NDcPP] application notes and assurance activities. The security target did not add or remove any security requirements. Consequently, [NDcPP] rationale applies and is complete.