SECURITY TARGET

FORTIGATE SOC2 APPLIANCES RUNNING FORTIOS 5.4

Reference	FOS-54-SOC2-ST	Status	Released
Version	1.1	Release Date	18 October 2018
Owner	Fortinet, Inc.	Pages	50 (Including Cover)

LIST OF CONTENTS

1 DOCUMENT INFORMATION

Prepared for

Fortinet, Inc.
899 Kifer Road
Sunnyvale, CA 94086
USA
http://www.fortinet.net

Prepared by

BAE SYSTEMS
INSPIRED WORK

BAE Systems Applied Intelligence Pty Ltd
Level 1, 14 Childers Street
Canberra ACT 2601
Australia
http://www.baesystems.com/ai

1.1 Amendment history

Version	Date	Revisions
0.1	15 November 2017	Initial draft.
0.2	15 November 2017	Initial draft release.
0.3	17 November 2017	Updated to address review comments.
1.0	10 April 2018	Release for posting to EPL.
1.1	18 October 2018	Update for CC portal listing.

1.2 Copyright statement

2 SECURITY TARGET INTRODUCTION (ASE_INT)

2.1 ST Identification

ST Title	Security Target - FortiGate SOC2 appliances running FortiOS 5.4
ST Version	**1.1**
ST Release Date	18 October 2018

2.2 TOE Identification

TOE Name	FortiGate SOC2 appliances running FortiOS
TOE Version	5.4 (Build 9791)
Protection Profile(s)	• collaborative Protection Profile for Stateful Traffic Filter Firewalls (FWcPP), Version 1.0, 27 February 2015
CC Identification	• Common Criteria for Information Technology (IT) Security Evaluation, Version 3.1 (Revision 4), September 2012 • Common Methodology for Information Technology Security Evaluation, Evaluation methodology, Version 3.1 (Revision 4), September 2012

2.3 Document organisation

This document is divided into the following sections:

- Section 2 (Security Target Introduction (ASE_INT)) provides the introductory material for the ST;
- Section 3 (Conformance Claims (ASE_CCL)) provides the conformance claims for the evaluation;
- Section 4 (Security Problem Definition (ASE_SPD)) provides the security problem to be addressed by the TOE and the operational environment of the TOE;
- Section 5 (Security Objectives (ASE_OBJ)) defines the security objectives for the TOE and the environment;
- Section 6 (Extended Components Definition (ASE_ECD)) provides a definition and justification for any extended components from CC Parts 2 or 3 that have been developed for the evaluation;
- Section 7 (Security Functional Requirements (ASE_REQ)) contains the functional and assurance requirements derived from the Common Criteria, Part 2 and 3, respectively that must be satisfied by the TOE; and

- Section 8 (TOE summary specification (ASE_TSS)) provides a summary of the TOE specification, identifying the IT security functions provided by the TOE.

2.4 References

[1] Common Criteria Part 1 (Introduction and general model), Version 3.1 Revision 4, September 2012

[2] Common Criteria Part 2 (Security functional components), Version 3.1 Revision 4, September 2012

[3] Common Criteria Part 3 (Security assurance components), Version 3.1 Revision 4, September 2012

[4] Common Criteria Evaluation Methodology (CEM), Version 3.1 Revision 4, September 2012

[5] collaborative Protection Profile for Stateful Traffic Filter Firewalls, Version 1.0, 27 February 2015

2.5 TOE overview

2.5.1 TOE type and usage

The Target of Evaluation (TOE) is FortiGate SOC2 appliances running FortiOS 5.4.

The TOE is designed to provide next-generation firewall services ensuring network protection for Internet Protocol version 4 (IPv4) and Internet Protocol version 6 (IPv6) networks. The TOE is capable of robust filtering based on information contained in IPv4, IPv6, ICMPv4, ICMPv6, TCP and UDP headers as specified by their respective RFC's. Additionally the TOE is capable of content inspection of FTP and H.323 protocols to work with the dynamic nature of these protocols.

The TOE has extensive logging capabilities - these include, but are not limited to the firewall rules described above, administrative actions and logging and tampering or misuse of the trusted cryptographic channels. These audit logs are capable of being exported to an external FortiAnalyzer™ audit server over a cryptographically protected channel for further analysis and inspection.

The TOE implements NIST approved cryptography, validated through numerous FIPS and CAVP validations (see Section **8.3**). This cryptography is used to secure communications to trusted administrators, remote authentication sources and to secure generated audit logs in transit to the FortiAnalyzer server for additional inspection. User administration sessions are capable over the local console or secured over HTTPS to a web based GUI using validated cryptography. To ensure proper random number generation capable of generating keys with 256 bits of strength the TOE has been equipped with a dedicated hardware noise source which provides entropy collected from the ambient environment in which the product operates. This noise source is continually monitored for its ongoing health and proper operation.

The TOE also offers the ability to verify through cryptographic signatures that product updates are valid, and will reject any updates without the appropriate Fortinet signature. The TOE will ensure during boot up that the health of the TOE has not been compromised through a variety of checks. These checks include health testing of the entropy source to ensure that the ambient environment is producing sufficient entropy for the seeding of the cryptographic module.

The TOE implements FIPS 140-2 level 2 hardware requirements for potential tampering and the firmware is inspected on startup as described in the FIPS 140-2 level 1 firmware integrity checks.

2.6 TOE description

This section addresses the physical and logical components of the TOE included in the evaluation.

2.6.1 Physical scope of the TOE

The physical scope of the TOE includes the TOE hardware and firmware as well as an FTR-ENT1 (where required) to provide the hardware noise source. An example of a typical TOE deployment is illustrated in the figure bellow.

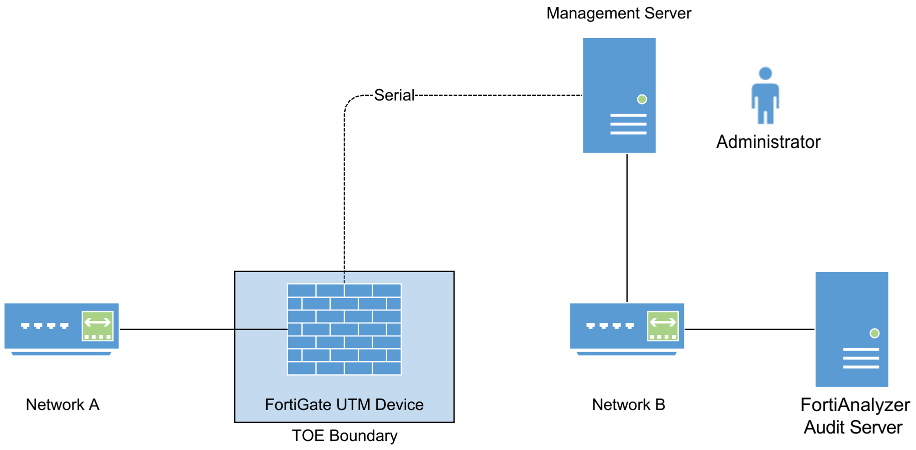

Figure 1 – TOE physical boundary (wired)

The following TOE hardware platforms are claimed for this evaluation.

- FortiGate-30D;
- FortiGate-30D-PoE;
- FortiWiFi-30D;
- FortiGate-60D;
- FortiGate-Rugged-60D;
- FortiWiFi-60D;
- FortiGate-90D;
- FortiGate-90D-PoE;
- FortiWiFi-90D; and
- FortiWiFi-90D-PoE.

Each platform requires the use of an FTR-ENT1 token to seed the TOE cryptographic system with entropy from the ambient environment.

2.6.2 Logical scope of the TOE

The TOE provides the following logical security functions.

Table 1 – Logical scope of the TOE.

TSF	Description
Security audit	The TOE generates logs for auditable events. These logs can be stored locally in protected storage and/or exported to an external audit server via a secure channel.
Cryptographic support	The TOE implements a variety of key generation and cryptographic methods to provide protection of data both in transit and at rest within the TOE.
User data protection	The TOE ensures that data cannot be recovered once deallocated.
Identification and authentication	The TOE implements mechanisms to ensure that users are both identified and authenticated before any access to TOE functionality or TSF data is granted.
Security management	The TOE provides a suite of management functionality, allowing for full configuration of the TOE by an authorised administrator.
Protection of the TSF	The TOE implements a number of protection mechanisms (including authentication requirements, self-tests and trusted update) to ensure the protection of the TOE and all TSF data.
TOE access	The TOE provides session management functions for local and remote administrative sections.
Trusted path/channels	The TOE provides secure channels between itself and local/remote administrators and other devices to ensure data security during transit.
Stateful traffic and packet filtering	The TOE allows for the configuration and enforcement of stateful packet filtering/firewall rules on all traffic traversing the TOE.

2.6.3 Summary of out-of-scope items

The FortiGate™ appliances are capable of a variety of functions and configurations which are not covered by the FWcPP.

The following features have not been examined as part of this evaluation:

- High-Availability;
- FortiExplorer client;
- Anti-spam;
- Anti-virus;
- Content filtering;
- Web filtering;
- Use of syslog;
- FortiToken and FortiSSO Authentication;
- Stream Control Transmission Protocol (SCTP), BGP, RIP and DHCP protocols; and
- Usage of the boot-time configuration menu to upgrade the TOE.

2.7 Product documentation

The TOE (FortiOS version 5.4, build b9791) includes the following product documentation:

- FIPS 140-2 and Common Criteria Compliant Operation for FortiOS™ 5.4, 12 October 2017;
- FortiOS Handbook - The Complete Guide to FortiOS 5.4, 13 September 2017;
- The FortiGate Cookbook 5.4, 19 February 2016;
- FortiOS 5.4.1 CLI Reference, 08 June 2016; and
- FortiOS 5.4.3 Log Reference, 21 December 2016.

3 CONFORMANCE CLAIMS (ASE_CCL)

3.1 CC conformance claim

The ST and TOE are conformant to version 3.1 (Revision 4) of the Common Criteria for Information Technology Security Evaluation.

- **Part 2 extended**. Conformant with Common Criteria for Information Technology Security Evaluation Part 2: Security functional requirements, version 3.1, Revision 4.
- **Part 3 conformant**. Conformant with Common Criteria for Information Technology Security Evaluation Part 3: Security assurance requirements, version 3.1, Revision 4.

3.2 Protection Profile conformance claim

Both the ST and TOE claim **exact** conformance to the:

- collaborative Protection Profile for Stateful Traffic Filter Firewalls (FWcPP), Version 1.0, 27 February 2015.

Augmented with the following NIAP/Network Interpretation Team (NIT) Technical Decisions:

- TD0090: NIT Technical Decision for FMT_SMF.1.1 Requirement in NDcPP;
- TD0107: FCS_CKM - ANSI X9.31-1998, Section 4.1.for Cryptographic Key Generation
- TD0112: NIT Technical Decision for TLS testing in the NDcPP v1.0 and FW cPP v1.0;
- TD0115: NIT Technical Decision for Transport mode and tunnel mode in IPsec communication in NDcPP and FWcPP;
- TD0117: NIT Technical Decision for FIA_X509_EXT.1.1 Requirement in NDcPP;
- TD0125: NIT Technical Decision for Checking validity of peer certificates for HTTPS servers;
- TD0130: NIT Technical Decision for Requirements for Destruction of Cryptographic Keys;
- TD0143: NIT Technical Decision for Failure testing for TLS session establishment in NDcPP and FWcPP;
- TD0150: NIT Technical Decision for Removal of SSH re-key audit events in the NDcPP v1.0 and FW cPP v1.0;
- TD0151: NIT Technical Decision for FCS_TLSS_EXT Testing - Issue 1 in NDcPP v1.0;
- TD0152: NIT Technical Decision for Reference identifiers for TLS in the NDcPP v1.0 and FW cPP v1.0;

- TD0153: NIT Technical Decision for Auditing of NTP Time Changes in the NDcPP v1.0 and FW cPP v1.0;
- TD0154: NIT Technical Decision for Versions of TOE Software in the NDcPP v1.0 and FW cPP v1.0;
- TD0156: NIT Technical Decision for SSL/TLS Version Testing in the NDcPP v1.0 and FW cPP v1.0;
- TD0160: NIT Technical Decision for Transport mode and tunnel mode in IPSEC communications;
- TD0168: NIT Technical Decision for Mandatory requirement for CSR generation;
- TD0169: NIT Technical Decision for Compliance to RFC5759 and RFC5280 for using CRLs;
- TD0189: NIT Technical Decision for SSH Server Encryption Algorithms;
- TD0199: NIT Technical Decision for Elliptic Curves for Signatures;
- TD0223: NIT Technical Decision for "Expected" vs "unexpected" DNs for IPsec Communications;
- TD0225: NIT Technical Decision for Make CBC cipher suites optional in IPsec; and
- TD0226: NIT Technical Decision for TLS Encryption Algorithms.

4 SECURITY PROBLEM DEFINITION (ASE_SPD)

This section describes the nature of the security problem that the TOE is designed to address. The security problem is described through:

- a set of **threats** that the TOE must mitigate,
- specific **assumptions** about the security aspects of the environment (both IT related and non-IT related elements) in which the TOE will operate, and
- relevant **organisational security policies** that specify rules or guidelines that must be followed by the TOE and/or the operational environment.

4.1 Threats

Table 2 – Identified threats (FWcPP)

Threat	Description
T.UNAUTHORIZED_ADMINISTRATOR_ACCESS	Threat agents may attempt to gain administrator access to the firewall by nefarious means such as masquerading as an administrator to the firewall, masquerading as the firewall to an administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between the firewall and a network device. Successfully gaining administrator access allows malicious actions that compromise the security functionality of the firewall and the network on which it resides.
T.WEAK_CRYPTOGRAPHY	Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.
T.UNTRUSTED_COMMUNICATION_CHANNELS	Threat agents may attempt to target firewalls that do not use standardized secure tunnelling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the firewall itself.
T.WEAK_AUTHENTICATION_ENDPOINTS	Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints – e.g., shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the firewall itself could be compromised.
T.UPDATE_COMPROMISE	Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.

T.UNDETECTED_ACTIVITY	Threat agents may attempt to access, change, and/or modify the security functionality of the firewall without administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the administrator would have no knowledge that the device has been compromised.
T.SECURITY_FUNCTIONALITY_COMPROMISE	Threat agents may compromise credentials and firewall data enabling continued access to the firewall and its critical data. The compromise of credentials include replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the administrator or firewall credentials for use by the attacker.
T.PASSWORD_CRACKING	Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the firewall. Having privileged access to the firewall provides the attacker unfettered access to the network traffic, and may allow them to take advantage of any trust relationships with other network devices.
T.SECURITY_FUNCTIONALITY_FAILURE	A component of the firewall may fail during start-up or during operations causing a compromise or failure in the security functionality of the firewall, leaving the firewall susceptible to attackers.
T.NETWORK_DISCLOSURE	An attacker may attempt to "map" a subnet to determine the machines that reside on the network, and obtaining the IP addresses of machines, as well as the services (ports) those machines are offering. This information could be used to mount attacks to those machines via the services that are exported.
T. NETWORK_ACCESS	With knowledge of the services that are exported by machines on a subnet, an attacker may attempt to exploit those services by mounting attacks against those services.
T.NETWORK_MISUSE	An attacker may attempt to use services that are exported by machines in a way that is unintended by a site's security policies. For example, an attacker might be able to use a service to "anonymize" the attacker's machine as they mount attacks against others.
T.MALICIOUS_TRAFFIC	An attacker may attempt to send malformed packets to a machine in hopes of causing the network stack or services listening on UDP/TCP ports of the target machine to crash.

4.2 Assumptions

Table 3 – Assumptions (FWcPP)

Assumption	Description
A.PHYSICAL_PROTECTION	The firewall is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security and/or interfere with the firewall's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the firewall and the data it contains. As a result, the cPP will not include any requirements on physical tamper protection or other physical attack mitigations. The cPP will not expect the product to defend against physical access to the firewall that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the firewall.
A.LIMITED_FUNCTIONALITY	The firewall is assumed to provide networking and filtering functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example the firewall should not provide computing platform for general purpose applications (unrelated to networking/filtering functionality).

A.TRUSTED_ADMINSTRATOR	The authorized administrator(s) for the firewall are assumed to be trusted and to act in the best interest of security for the organization. This includes being appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the firewall. The firewall is not expected to be capable of defending against a malicious administrator that actively works to bypass or compromise the security of the firewall.
A.REGULAR_UPDATES	The firewall firmware and software is assumed to be updated by an administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.ADMIN_CREDENTIALS_SECURE	The administrator’s credentials (private key) used to access the firewall are protected by the host platform on which they reside.

4.2 Organisational security policies

Table 4 – OSPs

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

5 SECURITY OBJECTIVES (ASE_OBJ)

5.1 Security objectives for the TOE

There are no security objectives defined for the TOE in the FWcPP.

5.2 Security objectives for the environment

Table 5 – Security objectives for the environment

Objective	Objective statement
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.TRUSTED_ADMIN	TOE Administrators are trusted to follow and apply all guidance documentation in a trusted manner.
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.

6 EXTENDED COMPONENTS DEFINITION (ASE_ECD)

6.1 Objectives

The following extended components are taken from the FWcPP (Ref. [5])

- FAU_STG_EXT.1;
- FCS_RBG_EXT.1;
- FCS_HTTPS_EXT.1
- FCS_SSHS_EXT.1;
- FCS_TLSC_EXT.2;
- FCS_TLSS_EXT.1;
- FCS_IPSEC_EXT.1;
- FIA_PMG_EXT.1;
- FIA_UIA_EXT.1;
- FIA_UAU_EXT.2;
- FIA_X509_EXT.1;
- FIA_X509_EXT.2;
- FIA_X509_EXT.3;
- FPT_SKP_EXT.1;
- FPT_APW_EXT.1;
- FPT_TUD_EXT.1;
- FTA_SSL_EXT.1; and
- FFW_RUL_EXT.1.

These extended SFRs are defined in the FWcPP (Ref. [5]) and, as such, will not be redefined in this ST.

7 SECURITY FUNCTIONAL REQUIREMENTS (ASE_REQ)

7.1 Overview

This section defines the security requirements satisfied by the TOE. Each requirement has been extracted from version 3.1 (Rev 4) of the CC, Part 2 providing functional requirements and Part 3 providing assurance requirements.

Part 2 of the CC defines an approved set of operations that may be applied to security functional requirements. Following are the approved operations and the document conventions that are used within this ST to depict their application:

- Assignment: The assignment operation provides the ability to specify an identified parameter within a requirement. Assignments are depicted using italicised text and are surrounded by square brackets as follows [*assignment*].
- Assignment within a selection: Indicated with [*italicised and underlined text*]
- Selection: The selection operation allows the specification of one or more items from a list. Selections are depicted using bold text and are surrounded by square brackets as follows [**selection**].
- Refinement: The refinement operation allows the addition of extra detail to a requirement. Refinements are indicated using **bolded text** for additions and ~~strike-through~~ for deletions.
- Iteration: The iteration operation allows a component to be used more than once with varying operations. Iterations are depicted by placing a number at the end of the component identifier (e.g. FCS_COP.1(1) and FCS_COP.1(2)).

The security functional requirements are expressed using the notation stated above and are identified in the table below.

Table 6 – Security functional requirements

Identifier	Title
Security audit (FAU)	
FAU_GEN.1(1)	Audit data generation
FAU_GEN.2	User identity association
FAU_STG_EXT.1	Security audit event storage
Cryptographic support (FCS)	
FCS_CKM.1(1)	Cryptographic key generation
FCS_CKM.2	Cryptographic key establishment
FCS_CKM.4	Cryptographic key destruction
FCS_COP.1(1)	Cryptographic operation (AES data encryption/decryption)
FCS_COP.1(2)	Cryptographic operation (Signature generation and verification)
FCS_COP.1(3)	Cryptographic operation (Hash algorithm)

Identifier	Title
FCS_COP.1(4)	Cryptographic operation (Keyed hash algorithm)
FCS_RBG_EXT.1	Random bit generation
FCS_HTTPS_EXT.1	HTTPS protocol
FCS_SSHS_EXT.1	SSH server protocol
FCS_TLSC_EXT.2	TLS Client protocol with authentication
FCS_TLSS_EXT.1	TLS Server protocol
User data protection (FDP)	
FDP_RIP.2	Full residual data protection
Identification and authentication (FIA)	
FIA_AFL.1	Authentication failure handling
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	X.509 certificate validation
FIA_X509_EXT.2	X.509 certificate authentication
FIA_X509_EXT.3	X.509 certificate requests
Security management (FMT)	
FMT_MOF.1(1)	Management of security functions behaviour (Trusted Update)
FMT_MTD.1(1)	Management of TSF data
FMT_SMF.1(1)	Specification of management functions
FMT_SMR.2	Restrictions on security roles
Protection of the TSF (FPT)	
FPT_SKP_EXT.1	Protection of TSF data (for reading of all symmetric keys)
FPT_APW_EXT.1	Protection of administrator passwords
FPT_TST_EXT.1	TSF testing
FPT_TUD_EXT.1	Trusted updates
FPT_STM.1	Reliable time stamps
TOE access (FTA)	
FTA_SSL_EXT.1	TSF-initiated session locking

Identifier	Title
FTA_SSL.3	TSF-initiated termination
FTA_SSL.4	User-initiated termination
FTA_TAB.1	Default TOE access banners
Trusted path/channels (FTP)	
FTP_ITC.1	Inter-TSF trusted channel
FTP_TRP.1	Trusted path
Stateful traffic filtering (FFW)	
FFW_RUL_EXT.1	Stateful traffic filtering

7.2 Security audit (FAU)

7.2.1 FAU_GEN.1(1) Audit data generation

FAU_GEN.1(1).1	The TSF shall be able to generate an audit record of the following auditable events: a) Start-up and shut-down of the audit functions; b) All auditable events for the not specified level of audit; and c) All administrative actions comprising: i. Administrative login and logout (name of user account shall be logged if individual user accounts are required for administrators). ii. Security related configuration changes (in addition to the information that a change occurred it shall be logged what has been changed). iii. 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). iv. Resetting passwords (name of related user account shall be logged). v. Starting and stopping services (if applicable) vi. [[*seeding from entropy token or CP9, failure to seed, reseeding*]]; d) Specifically defined auditable events listed in **Table 11 – SFRs and associated auditable events**.
FAU_GEN.1(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 PP/ST, information specified in column three of **Table 11 – SFRs and associated auditable events.**

Table 7 – SFRs and associated auditable events

SFR	Auditable event(s)	Additional audit record contents
FAU_GEN.1(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(1)	None	None
FCS_COP.1(2)	None	None
FCS_COP.1(3)	None	None
FCS_COP.1(4)	None	None
FCS_RBG_EXT.1	None	None
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session	Reason for failure
FCS_SSHS_EXT.1	Failure to establish a SSH session	Reason for failure
FCS_TLSS_EXT.1	Failure to establish a TLS session	Reason for failure
FDP_RIP.2	None	None

SFR	Auditable event(s)	Additional audit record contents
FIA_PMG_EXT.1	None	None
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Provided user identity, origin of the attempt (e.g., IP address).
FIA_UAU_EXT.2	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None	None
FIA_X509_EXT.1	Unsuccessful attempt to validate a certificate	Reason for failure
	Session establishment with CA	Entire packet contents of packets transmitted/received during session establishment
FIA_X509_EXT.2	None	None
FIA_X509_EXT.3	None	None
FMT_MOF.1(1)	Any attempt to initiate a manual update	None
FMT_MTD.1	All management activities of TSF data.	None
FMT_SMF.1	None	None
FMT_SMR.2	None	None
FPT_SKP_EXT.1	None	None
FPT_APW_EXT.1	None	None
FPT_TST_EXT.1	None	None
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None
FPT_STM.1	Changes to time.	The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL_EXT.1	Any attempts at unlocking of an interactive session.	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. 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	Initiation of the trusted path. Termination of the trusted path. Failure of the trusted path functions.	Identification of the claimed user identity.
FFW_RUL_EXT.1	Application of rules configured with the ‘log’ operation	Source and destination addresses Source and destination ports Transport Layer Protocol TOE Interface
	Indication of packets dropped due to too much network traffic	TOE interface that is unable to process packets Identifier of rule causing packet drop

7.2.2 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.

7.2.3 FAU_STG_EXT.1 Security audit event storage

FAU_STG_EXT.1.1	The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.1.
FAU_STG_EXT.1.2	The TSF shall be able to store generated audit data on the TOE itself.
FAU_STG_EXT.1.3	The TSF shall [**overwrite previous audit records according to the following rule:** [*delete the oldest stored audit logs*]] when the local storage space for audit data is full.

7.3 Cryptographic support (FCS)

7.3.1 FCS_CKM.1(1) Cryptographic key generation

FCS_CKM.1(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**] ~~and specified cryptographic key sizes [assignment: cryptographic key sizes] that meet the following: [assignment: list of standards].~~

7.3.2 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 meets the following: NIST Special Publication 800-56B, "Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography";** • **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"**] ~~that meets the following: [assignment: list of standards].~~

7.3.3 FCS_CKM.4 Cryptographic key destruction

FCS_CKM.4.1	The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method [• **For plaintext keys in volatile storage, the destruction shall be executed by a [single overwrite consisting of [zeroes]]**. • **For plaintext keys in non-volatile storage, the destruction shall be executed by the invocation of an interface provided by a part of the TSF that [logically addresses the storage location of the key and performs a** [**single**] **overwrite consisting of [zeroes].**] that meets the following: *No Standard*.

7.3.4 FCS_COP.1(1) Cryptographic operation (AES data encryption/decryption)

FCS_COP.1(1).1	The TSF shall perform encryption/decryption in accordance with a specified cryptographic algorithm AES used in *CBC* mode and cryptographic key sizes **128 bits, 256 bits, and [no other key sizes]** that meet the following: *AES as specified in ISO 18033-3, CBC as specified in ISO 10116.*

7.3.5 FCS_COP.1(2) Cryptographic operation (Signature generation and verification)

FCS_COP.1(2).1	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 or greater*] • **Elliptic Curve Digital Signature Algorithm and cryptographic key sizes** [*256 bits or greater*]] that meets 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,** • **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**].

7.3.6 FCS_COP.1(3) Cryptographic operation (Hash algorithm)

FCS_COP.1(3).1	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]~~ that meet the following: ISO/IEC 10118-3:2004.

7.3.7 FCS_COP.1(4) Cryptographic operation (Keyed hash algorithm)

FCS_COP.1(4).1	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*] and message digest sizes [**160, 256, 384, 512**] bits that meet the following: ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2".

7.3.8 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 [[*FTR-ENT 1 USB token, CPU, ASIC*] **hardware-based noise source**] with a minimum of [**256 bits**] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions", of the keys and hashes that it will generate.

7.3.9 FCS_HTTPS_EXT.1 HTTPS protocol

FCS_HTTPS_EXT.1.1	The TSF shall implement the HTTPS protocol that complies with RFC 2818.
FCS_HTTPS_EXT.1.2	The TSF shall implement HTTPS using TLS.
FCS_HTTPS_EXT.1.3	The TSF shall establish the connection only if [**the peer initiates handshake**].

7.3.10 FCS_SSHS_EXT.1 SSH Server protocol

FCS_SSHS_EXT.1.1	The TSF shall implement the SSH protocol that complies with RFCs 4251, 4252, 4253, 4254, and [**5647, 5656, 6187, 6668**].
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 [*32768*] 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**].
FCS_SSHS_EXT.1.5	The TSF shall ensure that the SSH transport implementation uses [**ssh-rsa**] and [**no other public key algorithms**] 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**] and [**no other MAC algorithms**] as its MAC algorithm(s) and rejects all other MAC algorithm(s).
FCS_SSHS_EXT.1.7	The TSF shall ensure that [**diffie-hellman-group14-sha1**] and [**no other methods**] are the only allowed key exchange methods used for the SSH protocol.
FCS_SSHS_EXT.1.8	The TSF shall ensure that the SSH connection be rekeyed after no more than 2^28 packets have been transmitted using that key.

7.3.11 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)**] supporting 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_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_SHA as defined in RFC 3268** • **TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268** • **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_SHA as defined in RFC 4492** • **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492** • **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289**].
FCS_TLSC_EXT.2.2	The TSF shall verify that the presented identifier matches the reference identifier according to RFC 6125.
FCS_TLSC_EXT.2.3	The TSF shall only establish a trusted channel if the peer certificate is valid.
FCS_TLSC_EXT.2.4	The TSF shall present the Supported Elliptic Curves Extension in the Client Hello with the following NIST curves: [**secp256r1**] and no other curves.
FCS_TLSC_EXT.2.5	The TSF shall support mutual authentication using X.509v3 certificates.

7.3.12 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)**] supporting 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_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_SHA as defined in RFC 3268** • **TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268** • **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_SHA as defined in RFC 4492** • **TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492** • **TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289**].
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]; generate EC Diffie-Hellman parameters over NIST curves [secp256r1] and no other curves; generate Diffie-Hellman parameters of size** [**2048**]].

7.4 User data protection (FDP)

7.4.1 FDP_RIP.2 Full residual data protection

FDP_RIP.2.1	The TSF shall ensure that any previous information content of a resource is made unavailable upon the [**allocation of the resource to**] all objects.

7.5 Identification and authentication (FIA)

7.5.1 FIA_AFL.1 Authentication failure handling

FIA_AFL.1.1	**Refinement**: The TSF shall detect when **an Administrator configurable positive integer of successive** unsuccessful authentication attempts occur related to **administrators attempting to authenticate remotely**.
FIA_AFL.1.2	**Refinement**: When the defined number of unsuccessful authentication attempts has been met, the TSF shall [**prevent the offending remote administrator from successfully authenticating until an Administrator defined time period has elapsed**].

7.5.2 FIA_PMG_EXT.1 Password management

FIA_PMG_EXT.1.1	The TSF shall provide the following password management capabilities for administrative passwords: • Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [**“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”**, [*all other standard ASCII characters*]]; • Minimum password length shall be settable by the Security Administrator, and shall support passwords of 15 characters or greater.

7.5.3 FIA_UAU_EXT.2 Password-based authentication mechanism

FIA_UAU_EXT.2.1	The TSF shall provide a local password-based authentication mechanism, [**none**] to perform administrative user authentication.

7.5.4 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

7.5.5 FIA_UIA_EXT.1 User identification and authentication

FIA_UIA_EXT.1.1	The TSF shall allow the following actions prior to requiring the non-TOE entity to initiate the identification and authentication process: • Display the warning banner in accordance with FTA_TAB.1; • [[*No other functionality*]]
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.

7.5.6 FIA_X509_EXT.1 X509 certificate validation

FIA_X509_EXT.1.1	The TSF shall validate certificates in accordance with the following rules: • RFC 5280 certificate validation and certificate path validation. • The certificate path must terminate with a trusted CA certificate. • The TSF shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates. • The TSF shall validate the revocation status of the certificate using [**a 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	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.

7.5.7 FIA_X509_EXT.2 X509 certificate authentication

FIA_X509_EXT.2.1	The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [**TLS, HTTPS**], and [**no additional uses**].
FIA_X509_EXT.2.2	When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [**not accept the certificate**].

7.5.8 FIA_X509_EXT.3 X509 certificate requests

FIA_X509_EXT.3.1	The TSF shall generate a Certificate Request Message as specified by RFC 2986 and be able to provide the following information in the request: public key and [**device-specific information, Common Name, Organization, Organizational Unit, Country**].
FIA_X509_EXT.3.2	The TSF shall validate the chain of certificates from the Root CA upon receiving the CA Certificate Response.

7.6 Security management (FMT)

7.6.1 FMT_MOF.1(1) Management of security functions behaviour (Trusted Update)

FMT_MOF.1(1).1	The TSF shall restrict the ability to <u>enable</u> the functions *perform manual update* to *Security Administrators*.

7.6.2 FMT_MTD.1(1) Management of TSF data

FMT_MTD.1(1).1	The TSF shall restrict the ability to *<u>manage</u>* the TSF data to *Security Administrators.*

7.6.3 FMT_SMF.1(1) Specification of management functions

FMT_SMF.1(1).1	**Refinement**: 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 cryptographic functionality,** • **Ability to configure the IPsec functionality,** • **Ability to import X.509v3 certificates,** • **Ability to enable, disable, determine and modify the behavior of all the security functions of the TOE identified in this EP to the Administrator,** • **Ability to configure all security management functions identified in other sections of this EP.** • Ability to configure firewall rules; [• **No other capabilities.**].

7.6.4 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.

7.7 Protection of the TSF (FPT)

7.7.1 FPT_SKP_EXT.1 Protection of TSF data (for reading of all symmetric keys)

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

7.7.2 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.

7.7.3 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**] to demonstrate the correct operation of the TSF: [• **CPU and Memory BIOS self-tests;** • **Boot loader image verification;** • **FIPS 140-2 Known Answer Tests (KAT); and** • **Noise source tests**].
FPT_TST_EXT.1.2	The TSF shall provide the capability to verify the integrity of stored TSF executable code when it is loaded for execution through the use of the TSF-provided cryptographic service specified in FCS_COP.1(2).

7.7.4 FPT_TUD_EXT.1 Trusted updates

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 software/firmware version**].
FPT_TUD_EXT.1.2	The TSF shall provide Security Administrators the ability to manually initiate updates to TOE firmware/software and [**support automatic checking for updates**].
FPT_TUD_EXT.1.3	The TSF shall provide a means to authenticate firmware/software updates to the TOE using **a** *digital signature mechanism* and [**published hash**] prior to installing those updates.
Application note	This is the base FPT_TUD_EXT.1 taken from the FWcPP, augmented by the VPNEP.

7.7.5 FPT_STM.1 Reliable time stamps

FPT_STM.1.1	The TSF shall be able to provide reliable time stamps.

7.8 TOE access (FTA)

7.8.1 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.

7.8.2 FTA_SSL.3 TSF-initiated termination

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

7.8.3 FTA_SSL.4 User-initiated termination

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

7.8.4 FTA_TAB.1 Default TOE access banners

FTA_TAB.1.1	**Refinement**: 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.

7.9 Trusted path (FTP)

7.9.1 FTP_ITC.1 Inter-TSF trusted channel

FTP_ITC.1.1	**Refinement**: The TSF shall be **capable of using**[**TLS**] **to** provide a trusted communication channel between itself **and authorized IT entities supporting the following capabilities: audit server**, [**no other capabilities**] 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 **<u>the TSF, or the authorized IT entities</u>** to initiate communication via the trusted channel.
FTP_ITC.1.3	The TSF shall initiate communication via the trusted channel for [*logging of audit messages*].

7.9.2 FTP_TRP.1 Trusted path

FTP_TRP.1.1	The TSF shall be **capable of using** [**SSH, TLS, HTTPS**] to provide a communication path between itself and authorized remote administrators that is logically distinct from other communication paths and provides assured identification of its end points and protection of the communicated data from disclosure and provides detection of modification of the channel data.
FTP_TRP.1.2	The TSF shall permit **<u>remote administrators</u>** to initiate communication via the trusted path.
FTP_TRP.1.3	The TSF shall require the use of the trusted path **<u>for initial administrator authentication and all remote administration actions</u>**.

7.10 Stateful traffic filtering (FFW)

7.10.1 FFW_RUL_EXT.1 Stateful traffic filtering

FFW_RUL_EXT.1.1	The TSF shall perform Stateful Traffic Filtering on network packets processed by the TOE.
FFW_RUL_EXT.1.2	The TSF shall allow the definition of Stateful Traffic Filtering rules using the following network protocol fields: • ICMPv4 ○ Type ○ Code • ICMPv6 ○ Type ○ Code • IPv4 ○ Source address ○ Destination Address ○ Transport Layer Protocol • IPv6 ○ Source address ○ Destination Address ○ Transport Layer Protocol ○ [**IPv6 Extension header type** [*Hop-by-Hop Options, Destination Options, Routing, Fragment, Authentication Header and No Next Header*]] • TCP ○ Source Port ○ Destination Port • UDP ○ Source Port ○ Destination Port and distinct interface.
FFW_RUL_EXT.1.3	The TSF shall allow the following operations to be associated with Stateful Traffic Filtering rules: permit or drop with the capability to log the operation.
FFW_RUL_EXT.1.4	The TSF shall allow the Stateful Traffic Filtering rules to be assigned to each distinct network interface.
FFW_RUL_EXT.1.5	The TSF shall: a) accept a network packet without further processing of Stateful Traffic Filtering rules if it matches an allowed established session for the following protocols: TCP, UDP, [**ICMP**] based on the following network packet attributes: 1. TCP: source and destination addresses, source and destination ports, sequence number, Flags; 2. UDP: source and destination addresses, source and destination ports; **3.** [**ICMP: source and destination addresses, type,** [**code**]]. b) remove existing traffic flows from the set of established traffic flows based on the following: [**session inactivity timeout, completion of the expected information flow**].

FFW_RUL_EXT.1.6	The TSF shall enforce the following default Stateful Traffic Filtering rules on all network traffic: a) The TSF shall drop and be capable of [logging] packets which are invalid fragments; b) The TSF shall drop and be capable of [logging] fragmented packets which cannot be re-assembled completely; c) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a broadcast network; d) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a multicast network; The TSF shall drop and be capable of logging network packets where the source address of the network packet is defined as being a loopback address; e) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address “reserved for future use” (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4; f) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as an “unspecified address” or an address “reserved for future definition and use” (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6; g) The TSF shall drop and be capable of logging network packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified; and h) [**no other rules**].
FFW_RUL_EXT.1.7	The TSF shall be capable of dropping and logging according to the following rules: a) The TSF shall drop and be capable of logging network packets where the source address of the network packet is equal to the address of the network interface where the network packet was received; b) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is a link-local address; c) The TSF shall drop and be capable of logging network packets where the source address of the network packet does not belong to the networks associated with the network interface where the network packet was received.
FFW_RUL_EXT.1.8	The TSF shall process the applicable Stateful Traffic Filtering rules in an administratively defined order.
FFW_RUL_EXT.1.9	The TSF shall deny packet flow if a matching rule is not identified.
FFW_RUL_EXT.1.10	The TSF shall be capable of limiting an administratively defined number of half-open TCP connections. In the event that the configured limit is reached, new connection attempts shall be dropped and the drop event shall be [**logged**].

7.13 Security assurance requirements

Table 8 – Security assurance requirements

Assurance class	Assurance component
Security Target (ASE)	Conformance claims (ASE_CCL.1)
	Extended components definition (ASE_ECD.1)
	ST introduction (ASE_INT.1)
	Security objectives for the operational environment (ASE_OBJ.1)
	Stated security requirements (ASE_REQ.1)
	Security Problem Definition (ASE_SPD.1)
	TOE summary specification (ASE_TSS.1)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance documents (AGD)	Operational user guidance (AGD_OPE.1)
	Preparative procedures (AGD_PRE.1)
Life cycle support (ALC)	Labelling of the TOE (ALC_CMC.1)
	TOE CM coverage (ALC_CMS.1)
Tests (ATE)	Independent testing – sample (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability survey (AVA_VAN.1)

8 TOE SUMMARY SPECIFICATION (ASE_TSS)

8.1 Overview

This section provides the TOE summary specification, a high-level definition of the security functions claimed to meet the functional and assurance requirements.

8.2 Security audit

The TOE generates audit records for the following events:

- Start-up and shut-down of the audit functions;
- All auditable events for the not specified level of audit; and
- All administrative actions comprising:
  - Administrative login and logout (name of user account shall be logged if individual user accounts are required for administrators).
  - Security related 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).
  - Starting and stopping services (if applicable)
- Seeding from entropy token or CP9, failure to seed or reseeding events;
- Start-up and shut-down of the IPS functions;
- All IPS auditable events;
- All dissimilar IPS events;
- All dissimilar IPS reactions;
- Totals of similar events occurring within a specified time period;
- Totals of similar reactions occurring within a specified time period; and
- All specifically defined auditable events listed in Table 11 – SFRs and associated auditable events and Table 12 – SFRs and associated auditable events (IPS).

For each auditable event, the TOE records the date and time of the event, subject identity (i.e. administrative user), type of event and/or reaction and (where applicable) the success or failure of the event.

Logs are written to the FortiGate unit hard disk if the unit contains one. Models that do not contain a hard disk log to system memory. The amount of audit data that can be stored is dependent on the capacity of the device (see Appendix A – Hardware platform details).

Local log files can only be deleted via the CLI by an authorised administrator. No editing of log data is permitted.

When the TOE is configured to transmit log data to an external FortiAnalyzer platform, log data is cached prior to transmission. As such, no modification or deletion of the log data is possible.

If the local storage for audit logs is filled, the oldest stored logs will be deleted in a First-In-First-Out (FIFO) order to allow for the saving of new events.

The TOE also permits the transmission of audit data from the TOE to a Fortinet FortiAnalyzer device. This data is transmitted via TLS.

8.3 Cryptographic support

The TOE uses FIPS-approved cryptography that has been implemented in FIPS 140-2 validated cryptographic modules (CMVP certificate #3192). The FIPS-validated cryptographic modules implemented in the TSF meet Security Level 1 overall and meet Security Level 3 for the following: cryptographic module ports and interfaces; roles, services and authentication; and design assurance.

The following certificates have been issued by the CAVP and are implemented accordingly in the TOE.

Table 9 – Implemented key generation methods

Operation	Method	Key size (in bits)	Applicable curve(s)	Applicable standard(s)	CAVP Certificate #
Key generation	RSA	2048 >	N/A	FIPS 186-4, Appendix B.3	2526
	Eliptic-curve	256 384 512	P-256 P-384 P-512	FIPS 186-4, Appendix B.4	1288
	Finite-field	2048 >	N/A	FIPS 186-4, Appendix B.1	1286

Table 10 – Implemented key establishment methods

Operation	Key establishment method	Applicable standard(s)	CAVP Certificate #
Key establishment	RSA-based	NIST SP 800-56B	N/A
	Eliptic-curve based	NIST SP 800-56A	1288
	Finite-field based	NIST SP 800-56A	1286

Table 11 – Implemented cryptographic methods

Operation	Cryptographic algorithm	Key size(s) in bits	Message digest size(s)	Applicable standard(s)	CAVP Certificate #
Encryption and decryption	AES in CBC or GCM modes	128 256	N/A	ISO 18033-3 ISO 10116 ISO 19772	4628 4602
Signature generation and verification	RSA	2048	N/A	FIPS 186-4 ISO/IEC 9796-2	2526
	ECDSA	256	N/A	FIPS 186-4 ISO/IEC 14888-3	1288
Hashing	SHA	160 256 384 512	N/A	ISO/IEC 10118-3:2004	3792 3777
Keyed-hash message authentication	HMAC-SHA	160 256 384 512	160 256 384 512	ISO/IEC 9797-2:2011 Section 7	3063 3050
Random bit generation	CTR_DRBG	N/A	N/A	ISO/IEC 18031:2011	1543

Cryptographic key destruction by the TOE meets the key zeroization requirements of Key Management Security Level 1 from FIPS PUB 140-2. The TOE only stores keys in memory, either in RAM or Flash memory.

The TOE provides the following zeroization methods for cryptographic keys and other material:

- Volatile memory (SDRAM): The TOE performs a single direct overwrite consisting of zeroes, followed by a read-verify. If the read-verification of the overwritten data fails, the process repeats.

- Non-volatile flash memory (Flash RAM): The TOE performs a single, direct overwrite consisting of zeroes, which is followed by a followed by a read-verify. If the read-verification fails, the process repeats.

Zeroisation of cryptographic keys is performed via the OS kernel and invoked via the Command Line Interface (CLI).

The following table lists the keys/CSPs used by the TOE, their storage location and format and their associated zeroisation method, per the description above.

Table 12 – Keys and CSPs

Key/CSP	Storage location and method	Usage	Zeroization
IPSec Manual Encryption Key	Plaintext in RAM	IPsec encryption key.	Overwritten with zeroes when no longer needed.
IPSec Session	Plaintext in RAM	IPsec session management key.	Overwritten with zeroes when no longer needed.
IPsec Authentication Key	Plaintext in RAM	IPsec peer authentication key.	Overwritten with zeroes when no longer needed.
IPSec Session Encryption Key	Plaintext in RAM	IPsec session encryption key.	Overwritten with zeroes when no longer needed.
IKE Authentication Key	Plaintext in RAM	IKE peer authentication key.	Overwritten with zeroes when no longer needed.
IKE Key Generation Key	Plaintext in RAM	IKE key generation key.	Overwritten with zeroes when no longer needed.
IKE Session Encryption Key	Plaintext in RAM	IKE session encryption key.	Overwritten with zeroes when no longer needed.
Diffie-Hellman Keys (IKE)	Plaintext in RAM	Secure key exchange for IKE.	Overwritten with zeroes when no longer needed.
Diffie-Hellman Keys (SSL)	Plaintext in RAM	Secure key exchange for SSL.	Overwritten with zeroes when no longer needed.
HTTPS/TLS Session	Plaintext in RAM	HTTPS/TLS session encryption key.	Overwritten with zeroes when no longer needed.
HTTPS/TLS Authentication Key	Plaintext in RAM	HTTPS/TLS peer authentication key.	Overwritten with zeroes when no longer needed.
HTTPS/TLS Session	Plaintext in RAM	HTTPS/TLS session keys.	Overwritten with zeroes when no longer needed.
HTTPS/TLS Encryption Key	Plaintext in RAM	HTTPS/TLS data encryption key	Overwritten with zeroes when no longer needed.
SSH Session Authentication Key	Plaintext in RAM	SSH session encryption key.	Overwritten with zeroes when no longer needed.
SSH Session Encryption Key	Plaintext in RAM	SSH data encryption key.	Overwritten with zeroes when no longer needed.
IPSec Manual Encryption Key	Plaintext in RAM	IPsec encryption key.	Overwritten with zeroes when no longer needed.
IPSec Session	Plaintext in RAM	IPsec session management key.	Overwritten with zeroes when no longer needed.
IPsec Authentication Key	Plaintext in RAM	IPsec peer authentication key.	Overwritten with zeroes when no longer needed.
IPSec Session Encryption Key	Plaintext in RAM	IPsec session encryption key.	Overwritten with zeroes when no longer needed.

Key/CSP	Storage location and method	Usage	Zeroization
IKE Authentication Key	Plaintext in RAM	IKE peer authentication key.	Overwritten with zeroes when no longer needed.
IKE Key Generation Key	Plaintext in RAM	IKE key generation key.	Overwritten with zeroes when no longer needed.
IKE Session Encryption Key	Plaintext in RAM	IKE session encryption key.	Overwritten with zeroes when no longer needed.
Diffie-Hellman Keys (IKE)	Plaintext in RAM	Secure key exchange for IKE.	Overwritten with zeroes when no longer needed.

The TOE is capable of generating 256 bits *of* entropy using a dedicated hardware noise source and using this to seed the random bit generator in order to provide cryptographic services with up to 256 bits of strength. The TOE is also capable of importing cryptographic keys and certificates from outside the TOE boundary. These keys are zeroized when no longer required.

A detailed design of the cryptographic subsystems and entropy noise sources provided by the TOE has been conducted and was used to design the TOE to ensure strong seeding of the DRBG. This has been found to meet the entropy requirements for collection, strength and the seeding of the DRBG contained within the TOE.

8.4 HTTPS/TLS

The TOE implements HTTPS in accordance with RFC 2818 and TLS in accordance with RFCs 4346 (TLS v1.1) and 5246 (TLS v1.2).

The TOE permits the use of peer certificates. If a certificate presented is determined to be invalid, the TOE will not establish the connection.

The TOE supports the following cipher suites for TLS connections:

- 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_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_SHA as defined in RFC 3268;
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 3268;
- 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_SHA as defined in RFC 4492;
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492;
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289; and
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289.

The TOE will deny connections where the requested protocol is SSL 1.0 through 3.0 and TLS 1.0.

The TOE generates RSA keys of 2048-bit size, uses NIST curve secp256r1 and generates Diffie-Hellman parameters of 2048-bit size for use in key agreement/key exchange messages. The TOE supports the Supported Eliptic Curves Extension by default.

The TOE establishes reference identifiers using the following:

- Host IP address; and/or
- Fully qualified domain name (FQDN).

Wildcards are supported. Certificate pinning is not supported/used.

8.5 SSH

The TOE implements SSH in compliance with RFCs 4251 through 4254, 5647, 5656, 6187 and 6668.

The TOE supports password-based or public key (SSH-RSA) authentication.

The TOE examines the size of each received SSH packet. If the packet is greater than 32768 bytes, it is automatically dropped.

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

The TOE provides data integrity for SSH connections via HMAC-SHA1, HMAC-SHA2-256 and HMAC-SHA2-512.

The TOE supports Diffie-Hellman Group 14 SHA-1 (diffie-hellman-group14-sha1) for SSH key exchanges.

The TOE will re-key SSH connections after 2^28 packets have been transmitted.

8.6 800-56B conformance statements

The TOE can act as both a sender and receiver for RSA-based key establishment schemes.

The TOE fulfils the NIST SP 800-56B requirements listed below without extensions. The TOE does not implement any functionality within the SP 800-56B standard that is listed as “should not” and “shall not”.

Specifically, the TOE claims conformance to:

- Section 5.9 (Key Derivation Functions for Key Establishment Schemes);
- Section 6.3.1 (RSAKPG1 Family: rsakpg1-basic RSA Key Pair Generation with a Fixed Public Exponent);
- Section 6.3.2 (RSAKPG2 Family: rsakpg1-basic RSA Key Pair Generation with a Random PublicExponent);
- Section 6.4 (Assurances of Validity);
  - Section 6.4.1 (Assurance of Key Pair Validity);
  - Section 6.4.2 (Recipient Assurances of Public Key Validity);
- Section 8 (Key Agreement Schemes
- Section 9 (IFC based Key Transport Schemes

8.8 User data protection

The TOE ensures that no information from previously processed information flows is transferred to subsequent information flows. This applies both to information that is input to the TOE from an external source and to information (e.g., padding bits) that might be added by the TOE during processing of the information from the external source. The removal of any previous residual information is done through the zeroization of data when the memory structure is initially created and strict bounds checking on the data prior to it being assigned in memory.

8.9 Identification and authentication

The TOE permits administrators to set a positive integer for failed remote authentication attempts. When this limit is met, the remote user must wait for a defined period of time before further authentication attempts can be made.

The TOE enforces a password policy. Administrative passwords must be at least 15 characters long and may be comprised of any combination of upper and lower case letters, numbers, and the following special characters: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)” and all other standard ASCII characters.

Administrators connecting via a local connection (console) or remote (HTTPS/TLS or SSH) must provide a valid username and password to complete authentication. The TOE provides only obscured feedback (*’s) while authentication is in progress. The logon process is as follows:

- The local administrator connects to the TOE via the console port.
- For remote connections, the remote administrator connects via SSH or the web GUI (TLS/HTTPS). Key exchange and session establishment actions take place;
- The administrator is prompted for their username and password, which they enter.
- If the username and password provided is incorrect, the administrator is presented with an error. See above for the TOE’s behaviour if the number of unsuccessful attempts exceeds the defined threshold; or
- If the username and password provided are correct, the TOE shall end the logon process and give the administrator access to TOE functionality (a successful logon).

8.10 X509 certificates

The TOE utilises X509 certificates in accordance with RFC 5280.

The TOE performs validation of certificates during the handshaking process for both HTTPS and IPsec connections. See below for a description of the certificate path validation algorithm.

The certificate to be used for a given connection is configured by the administrator as part of the applicable policy (TLS, etc.).

If the TOE is unable to use a CRL for determining certificate validity, the TOE will not establish a connection.

The TOE will include the following information in generated CSRs:

- Certification Name;
- Subject Information;
- Organizational Unit;
- Organization;
- Locality (City);

- State/Province;
- Country/Region;
- E-mail;
- Key Type;
- Key Size; and
- Enrolment Method: File Based or Online SCEP.

The TOE provides a single certificate store contained within the TOE file system and no access is provided except via the web GUI or command line for addition/deletion of certificates or CRLs. Certificates must be enabled via the Features list before access to the store is provided.

Once a certificate is loaded into the store, the only editing allowed is to add a comment that is associated with the certificate. No other editing of certificates, keys or CRLs is allowed.

The TOE validates certificates via Certificate Revocation List (CRL, RFC 5759). Certificate validation takes place during the handshake of HTTPS connections. The validation is performed in the following steps:

- The remote client sends its certificate and associated key to the TOE.
- The TOE compares the received certificate and key against the certificate store (CA certificates, remote certificates, etc.) to determine that the certificate is authentic.
- The TOE then compares the certificate against any loaded CRLs.
- If the certificate is determined to be invalid or revoked, the certificate is rejected and the connection is not established
- If the certificate is determined to be valid, the connection process continues.

If the certificate path does not terminate with a trusted CA, the validation will fail.

The TOE will reject CA certificates that lack the basicConstraints section, or contain the section but whose CA flag is not set to TRUE.

When validating certificates provided for a specific purpose (trusted updates and executable code verification, TLS client/server or OCSP signing), the TOE ensures that the extendedKeyUsage field is set to an appropriate value (id-kp 3, 1, 2 or 9 with appropriate OID, respectively).

The TOE is able to generate certificate request messages (per RFC 2986). These requests will include the public key of the TOE and, where specified, device-specific information, the Common Name, Organization, Organizational Unit and Country.

8.11 Security management

The TOE does not permit access to any functions (other than the warning/consent banner and authentication interface) prior to login.

The TOE defines a single role, which is that of the Security Administrator. The Security Administrator is able to perform the following functions:

- Administer the TOE locally and remotely;
- Configure the access banner;
- Configure the session inactivity time before session termination or locking;
- Update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- Configure the cryptographic functionality;
- Modify, delete, generate and/or import cryptographic keys;

- Import X.509v3 certificates;
- Ability to configure firewall rules;

8.12 Protection of the TSF

The TOE prevents the reading of all pre-shared keys, symmetric keys and private keys stored within the TOE boundary.

Pre-shared keys related to administrator passwords and other credentials for the secure operation of the TOE are stored in the TOE's configuration file. Authorized administrators are allowed to enter this information through the communications paths such as the local console or HTTPS GUI. Once the password is entered the TOE encrypts the password using AES-128 and writes the password to the configuration file permanently obscuring the contents. This configuration file with the encrypted password hashes is available through the local console and HTTPS GUI by viewing a full configuration or backup of the configuration. The AES key for the protection of this configuration file and its passwords is generated by the TOE when the TOE is initialized and put into FIPS mode.

The TOE performs the following self-tests upon initialisation:

- **CPU and Memory BIOS self-tests**
  - CPU and memory are initialized by exercising a set of known answer tests and the BIOS is compared against a known checksum of the image. The memory is zeroized and then has a random pattern written and read from the memory.
- **Boot loader image verification**
  - The boot loader will compare the image of the TOE to a known checksum of the image prior to booting.
- **Noise source tests**
  - The noise source is started and pattern analysis is done on the output to ensure that the source is not stuck in a cryptographically weak state. These include both the repetition and adaptive proportion tests
- **FIPS 140-2 Known Answer Tests (KAT)**
  - Comparison of a number of cryptographic functions against an expected set of values

The Noise-source tests and FIPS 140-2 KAT can also be run on demand by the user.

The above tests ensure that the CPU and memory utilised by the TOE are functioning as intended, the BIOS and boot loader image are authentic and stable, the noise source used for entropy generation is functioning at capability and that the cryptographic algorithms used by the TOE are operating correctly. Together, these tests ensure that the TOE is operating at its intended level of capability.

There are several self-tests in which the TOE will enter an error-based blocking state. The first is a failure of any self-tests upon initialization of the TOE. This includes (but is not limited to) BIOS, software/firmware integrity checks and cryptographic self-tests. Upon the detection of one of these test failures, the TOE will halt and no further processing will occur until the TOE is reset.

Additionally the TOE may receive traffic above the capacity of the product it will drop all packets above this capacity. These events are logged to the audit log of the TOE.

The administrator may query the current version of the TOE via the GUI or CLI. The TOE will notify administrators if a new update file is available, but the update process will not commence until requested by the administrator.

Updates to the TOE are applied in accordance with the following process:

- The administrator downloads the upgrade image/package from the Fortinet website.

- Once downloaded, the administrator must transfer the image to the TOE via a trusted path (e.g. the web interface).
- Upon initiating the update process, the TOE will attempt to verify the integrity and authenticity of the update package. This is achieved via the verification of a 2048-bit RSA signature that is applied to the package by the Fortinet development team.
- If the signature cannot be verified, or the integrity of the package cannot be confirmed, the upgrade will fail and an audit log generated accordingly.
- If the signature is verified correctly and the integrity of the package is confirmed, the upgrade will be applied and the TOE restarted.

Additionally, MD5 hashes of each update file are published along with the image on the Fortinet website. Administrators may compare these published hashes against the hashes of the file they have downloaded to ensure that the file is valid.

The TOE maintains its own time source, which is free from outside interference. This timestamp is used for the purposes of generating audit logs and other time-sensitive operations on the TOE including cryptographic key regeneration intervals. The TOE may also connect to an NTP server for synchronisation.

8.13 TOE access

TOE administrators may access the TOE remotely (via the HTTPS/TLS web GUI or SSH) or locally (via the console port).

The TOE permits administrators to define a session lifetime for both local and remote sessions. Once this time limit has been met, the TOE will automatically close the active session (local or remote) and require TOE administrators to re-authenticate before any access to TSF data is permitted. TOE administrators may also manually close their sessions.

Users connecting to the TOE will be presented with a warning and consent banner prior to authentication.

8.14 Trusted path/channels

The TOE provides an Inter-TSF trusted channel between itself and the following entities:

- Between the TOE and a FortiAnalyzer logging platform using TLS; and

These channels can be initiated by either the TOE or the authorised entities.

The TOE provides a trusted path between itself and remote administrative users using the following protocols:

- TLS (Versions 1.1 and 1.2) and HTTPS (in compliance with RFC 2818) for the Web GUI; and
- SSH in compliance with the following RFCs: 4251, 4252, 4253, 4254, 5647, 5656, 6187 and 6668.

These protocols implement cryptographic algorithms to provide data transport security and integrity, preventing unauthorised access to (or modification of) data sent between the TOE and remote administrative users.

8.15 TOE initialisation

The Fortinet family of appliances provides a secure initialization procedure to ensure the integrity of the image and correct cryptographic functioning of the product prior to any information flowing. The product starts from a powered down state and no signals on the wire. The device then powers on and undergoes the following initialization process:

- Bootstrap and Boot Loader

- Verification of the kernel, firmware and software images
- Loading and Initialization of
    - Kernel;
    - Firmware;
    - Cryptographic known answer tests;
    - Entropy gathering and DRBG initialization; and
    - Cryptographic module

Once the kernel, firmware and cryptographic services have been initialized the TOE loads the configured firewall rules. The configuration file is then consulted and are initialized and configured with their network settings as specified and if appropriate transitioned to the link up sate. At this point packets may begin flowing through the various network interfaces. The CLI daemon is then started followed by the Web and the TOE is available for login to accept administrative connections.

8.16 Stateful traffic/packet filtering

The TOE permits the configuration of stateful packet filtering policies. The following protocols and associated attributes are configurable within each policy:

- ICMPv4 (RFC 792)
    - Type; and
    - Code
- ICMPv6 (RFC 4443)
    - Type; and
    - Code
- IPv4 (RFC 791)
    - Source address;
    - Destination Address; and
    - Transport Layer Protocol
- IPv6 (RFC 2460)
    - Source address;
    - Destination Address;
    - Transport Layer Protocol; and
    - The following IPv6 Extension header types:
        - Hop-by-Hop Options;
        - Destination Options;
        - Routing;
        - Fragment;
        - Authentication Header; and
        - No Next Header.
- TCP (RFC 793)
    - Source Port; and
    - Destination Port
- UDP (RFC 768)

  - Source Port; and
  - Destination Port

Rules can be configured to permit or drop traffic (with the generation of audit log entries for either option).

Each rule can be tied to a specific interface (port1, wan1, etc.).

Each packet that arrives on an interface is subject to the enforcement of the stateful traffic filtering. This filtering verifies if the connection is part of an established session or if it is a new connection. If the security attributes of the incoming connection request match those already present for an entry in the state table of the TOE the information flow is automatically allowed. Otherwise this is considered a new connection attempt.

For a new connection attempt a list of administrator-defined security rules are consulted in their sequence order until a match is found for that packet. The packet is then allowed, denied or dropped based on the configuration of this rule.

The session database is consulted to see if an additional session can be created by examining how many currently exist in the database. If this number is below the hardware limit sessions are established by writing the attributes and a TTL into the session database. If the connection is allowed a new session is written into the list of established sessions and can be used to allow subsequent packets for this connection. If logging is enabled for the rule the audit event is sent in real time to the audit server.

Any new session will have the first packet of the exchange inspected according to the firewall table as described above, such as the TCP SYN packet during a typical TCP session negotiation for both the sender and receiver. The TOE will write to the session table the expected source and destination ports for this communication flow based on the observed IP headers.

For FTP the initial handshake communication on port 21 for FTP will be inspected, as well as the server response indicating the expected data and control communication ports. A session will be written to the state table reflecting the expected source and destination ports based on this packet inspection.

For H.323 the TOE will inspect the ARQ request to the gatekeeper device and allow the establishment of this communication via an entry into the state table. The TOE will inspect the response from the gatekeeper to determine the expected UDP port and IP address of the device registered with the gatekeeper and write a session to the session table indicating that this communication is expected and should be allowed.

The TOE utilises a session database to track active sessions for TCP, UDP and ICMP (amongst other protocols). A number of variables (such as source/destination address and ports, sequence numbers, flags and TTL values) are utilised in the management of sessions.

Periodically old sessions exceeding their TTL are removed from the database. Sessions that have been closed are similarly removed from the database.

Each FortiGate™ appliance has a pre-defined number of sessions it can track and is specified on the specifications sheet.

When encountered by the TOE, the following packets will be automatically dropped and an audit log generated for each event:

- Packets which are invalid fragments (see below);
- Fragments that cannot be completely re-assembled;
- Packets where the source address is defined as being on a broadcast network;
- Packets where the source address is defined as being on a multicast network;
- Packets where the source address is defined as being a loopback address;
- Packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address “reserved for future use” (i.e.

240.0.0.0/4) as specified in RFC 5735 for IPv4;

- Packets where the source or destination address of the network packet is defined as an “unspecified address” or an address “reserved for future definition and use” (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;
- Packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified.
- Packets where the source address is equal to the address of the network interface where the network packet was received;
- Packets where the source or destination address of the network packet is a linklocal address; and
- Packets where the source address does not belong to the networks associated with the network interface where the network packet was received, including a description of how the TOE determines whether a source address belongs to a network associated with a given network interface.

The TOE is capable of detecting fragmented packets. When fragmented packets arrive at their destination, they are reassembled and read. If the fragments do not arrive together, they must be held until all of the fragments arrive. Reassembly of a packet requires all of the fragments. The TOE in the evaluated configuration will attempt to reassemble fragmented packets. When these packets arrive at the TOE they will be held by the TOE for reassembly until the TTL expires. Should the TOE detect that there is a missing or invalid fragment during the reassembly the packet will be dropped and logged. This behaviour is capable of being modified or overwritten by the TOE administrator.

Incoming packets are inspected against the session database. Sessions that match all the security attributes and do not exceed the TTL are automatically passed on to their destination. Packets that do not match the attributes in the session database are then compared to the defined firewall rules for that interface identifier based on their unique numerical order. Packets that are permitted are passed to their destination, packets marked for logging are written to the audit log and packets marked for dropping are discarded.

Packet rules are enforced in the order defined by the administrator. If no matching rule is found, the TOE will automatically deny the packets and generate a log entry accordingly.

The TOE maintains half-open TCP sessions in the same manner as full TCP sessions. Once the administrator-defined limit for total sessions is met, sessions (both valid and half-open) are automatically closed based on their timeout value (if not cleared manually by an administrator).

9 APPENDIX A – HARDWARE PLATFORM DETAILS

Model	CPU	ASIC	RAM	Boot	Storage	Entropy
FortiGate-30D	Fortinet SoC2	CP7 Lite	1GB	4GB	N/A	Token
FortiGate-30D-PoE	Fortinet SoC2	CP7 Lite	1GB	4GB	N/A	Token
FortiWiFi-30D	Fortinet SoC2	CP7 Lite	1GB	4GB	N/A	Token
FortiGate-60D	Fortinet SoC2	CP7 Lite	2GB	4GB	N/A	Token
FortiGate-Rugged-60D	Fortinet SoC2	CP7 Lite	2GB	4GB	N/A	Token
FortiWiFi-60D	Fortinet SoC2	CP7 Lite	2GB	8GB	N/A	Token
FortiGate-90D	Fortinet SoC2	CP7 Lite	2GB	8GB	32GB	Token
FortiGate-90D-PoE	Fortinet SoC2	CP7 Lite	2GB	8GB	32GB	Token
FortiWiFi-90D	Fortinet SoC2	CP7 Lite	2GB	8GB	32GB	Token
FortiWiFi-90D-PoE	Fortinet SoC2	CP7 Lite	2GB	8GB	32GB	Token

--- END OF DOCUMENT ---