FortiSandbox 4.4 Security Target

Version: 1.6

October 2025

Prepared For:

Fortinet, Inc.

899 Kifer Rd

Sunnyvale, CA 94086

Prepared By:

UL Verification Services Inc.

Notices:

Table of Contents

1. Security Target (ST) Introduction

The structure of this document is defined by CC v3.1r5 Part 1 Annex A.2, “Mandatory contents of an ST”:

- Section 1 contains the ST Introduction, including the ST reference, Target of Evaluation (TOE) reference, TOE overview, and TOE description.
- Section 2 contains conformance claims to the Common Criteria (CC) version, Protection Profile (PP), PP-module and any security package claims, as well as rationale for these conformance claims.
- Section 3 contains the security problem definition, which includes Threats, Organizational Security Policies (OSP), and Assumptions that must be countered, enforced, and upheld by the TOE and its operational environment.
- Section 4 contains statements of Security Objectives for the TOE, and the TOE Operational Environment (OE) as well as rationale for these security objectives.
- Section 5 contains the definitions of any claimed extended security requirements.
- Section 6 contains the Security Function Requirements (SFR), the Security Assurance Requirements (SAR), as well as the rationale for the claimed SFR and SAR.
- Section 7 contains the TOE summary specification, which includes the detailed specification of the IT security functions

1.1 Security Target Reference

The Security Target reference shall uniquely identify the Security Target.

ST Title: FortiSandbox 4.4 Security Target

ST Version: Version 1.6

ST Author(s): UL Verification Services, Inc

ST Publication Date: October 2025

Keywords: Network Device, SSH

1.2 Target of Evaluation Reference

The Target of Evaluation reference shall identify the Target of Evaluation.

TOE Developer: Fortinet, Inc.
899 Kifer Rd
Sunnyvale, CA 94086

TOE Identification: FortiSandbox 4.4

1.3 Target of Evaluation Overview

1.3.1 TOE Product Type

The TOE is classified as a Virtual or Standalone Network Device.

1.3.2 TOE Usage

FortiSandbox 4.4 is a high-performance security solution that utilizes AI/machine learning technology to identify and isolate advanced threats in real-time.

FortiSandbox inspects files, websites, URLs and network traffic for malicious activity, including zero-day threats, and uses sandboxing technology to analyze suspicious files in a secure virtual environment.

The TOE Evaluated Configuration includes only the functions necessary to enforce the requirements of this protection profile, described below. All other functionality, Fortinet product interactions, or other unique features of the device are outside the scope of the evaluation and were not tested.

1.3.3 TOE Major Security Features Summary

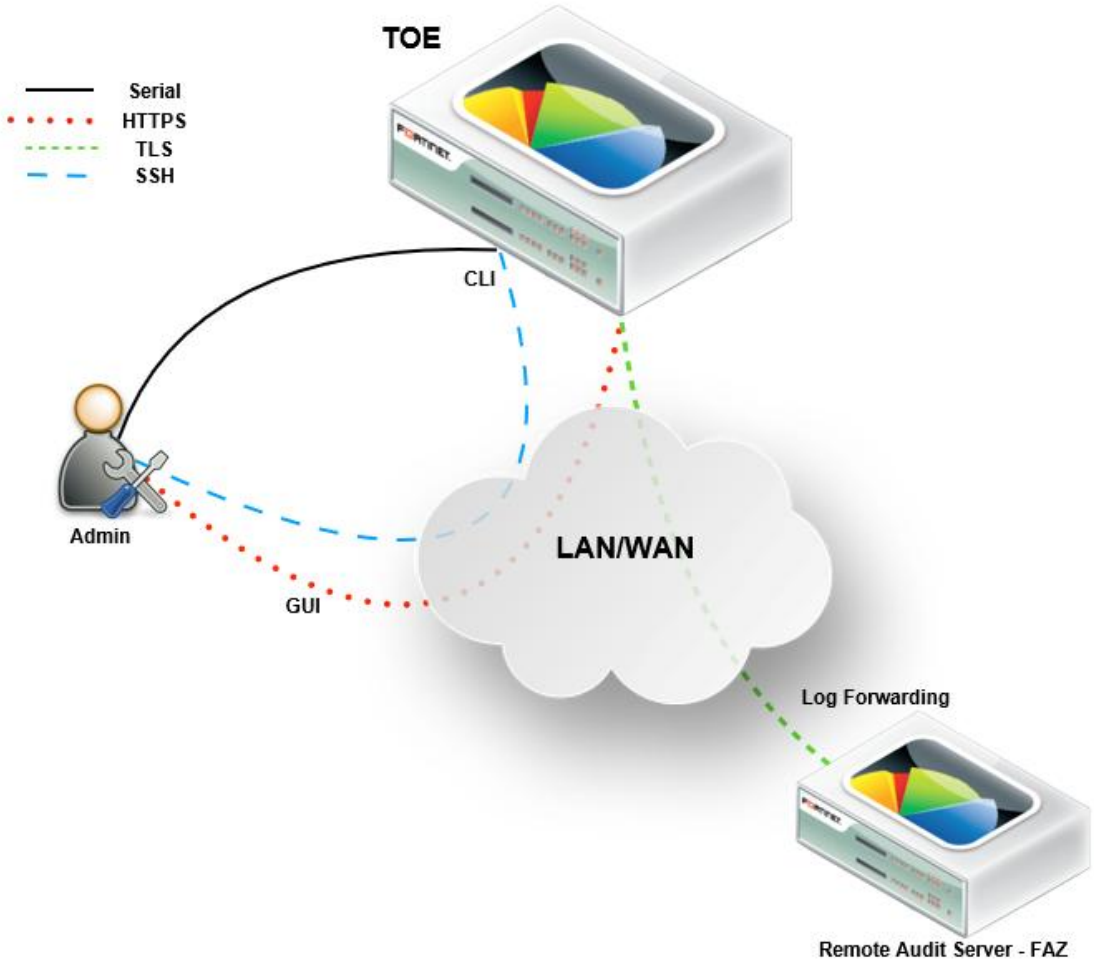
- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted Path/Channels

1.4 Target of Evaluation Description

The TOE interfaces are as follows:

- CLI: Administrative CLI via direct serial connection or SSH.
- GUI: Administrative web GUI via HTTPS.

- Logs: Forwarding of TOE audit events to a remote audit server, which is a Fortinet FortiAnalyzer (FAZ), via TLS.

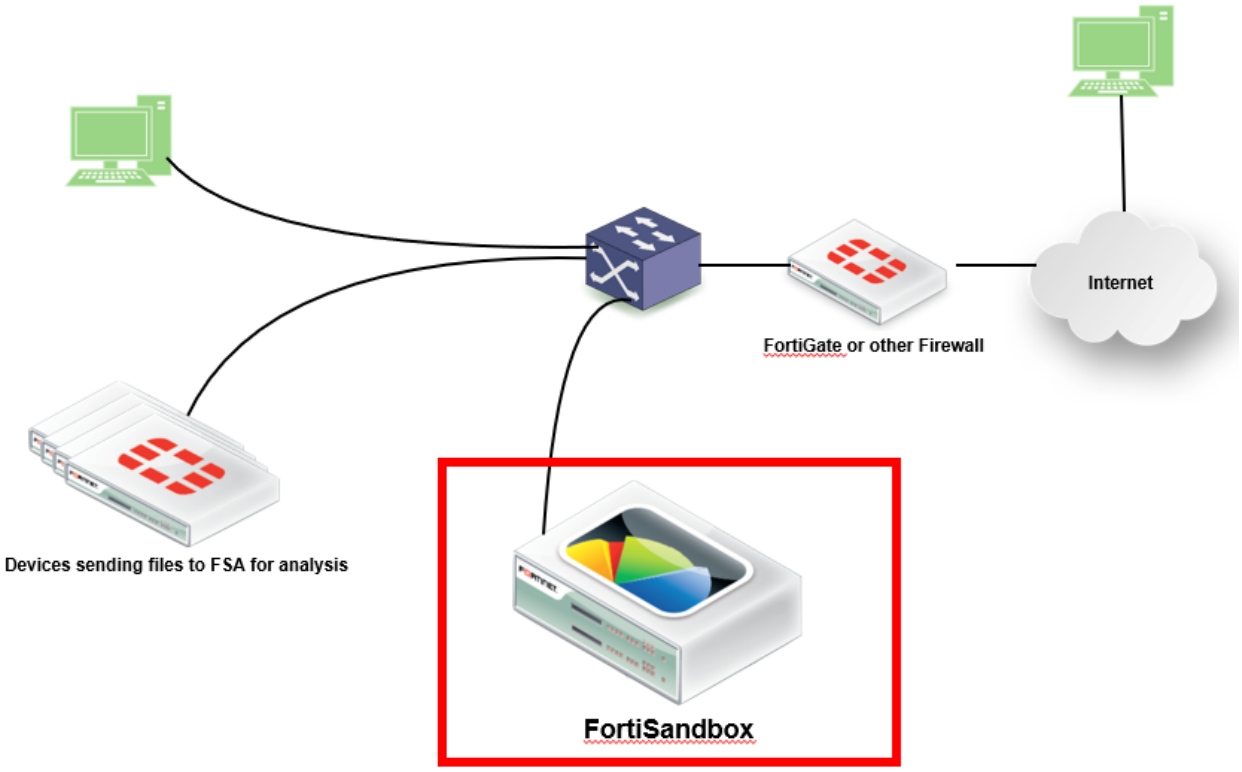


1.4.1 Non-TOE Hardware/Software/Firmware Required By The TOE

- Remote audit server
  - In the evaluated configuration, this must be a Fortinet FAZ appliance
- Platform (hardware and firmware) on which the virtual appliance TOE is hosted. In the tested configuration, this included the following:
  - VMware ESXi 8.0
  - Intel Xeon E5-2630v3, 8 Cores, 2.10GHz processor (Broadwell) processor
- **Hypervisor Environment.** The TOE virtual appliances can be deployed to:
  - Private VM environments (hypervisors) such as VMware ESXi[1]

[1] *Note: The evaluated configuration includes the VMware ESXi 8.0 Hypervisor as the private VM environment on which the TOE virtual appliance was deployed. Other VM appliances have not been tested under this evaluated configuration. A full list of hypervisors and marketplaces to which the TOE's virtual appliances can be deployed can be found on the Deployment Guides section of the FortiSanbox 4.4 documentation: https://docs.fortinet.com/product/fortisandbox/4.4*

- Access to a Certification Authority and corresponding revocation checking mechanism for certificate management.
- A remote management workstation with a supported web browser for remote administrative access:
  - The tested configuration used Google Chrome 117.0.5938.150



The TOE runs in CC mode of operation. Non CC mode of operation is excluded from the evaluation configuration.

1.4.2 TOE Physical Scope

As a standalone network appliance, the TOE consists of the following parts:

Hardware:

- FortiSandbox 1500G, using an AMD EPYC 7313P CPU on the Zen 3 microarchitecture
- FortiSandbox 500G, using an AMD EPYC 3251 CPU on the Zen microarchitecture
- FortiSandbox 3000F, using an AMD EPYC 7402 CPU on the Zen 2 microarchitecture

Software:

- FortiSandbox version 4.4.6-build 4527
  - Included with the product, and available for download from the Fortinet website as a binary file.

As a virtual network appliance running on an ESXi virtualization server, the TOE consists of the following parts:

Software:

- The FortiSandbox 4.4.6-build 4527 VM image for ESXi virtualization servers

Documentation:

- The guidance documentation that is part of the TOE is listed in Section 9 “References” within Table 11: TOE & TOE Guidance Documentation.
- Documentation is available in PDF or HTML format from the vendor website. Administrators can also access documentation by clicking the “?” help button in the TOE GUI to be taken to the online documentation portal on the Fortinet website.

1.4.2.1 Evaluated Configuration

The TOE has 4 evaluated configurations The evaluated configuration consists of the physical scope components listed in Section 1.4.1 above, in conjunction with the administrative configuration as described in the Guidance Documentation.

Additionally, the last evaluated configuration is a virtual network device running as a VM on ESXi virtualization platforms.

When installed as a physical network appliance, the TOE evaluated configuration is a standalone network device.

No other evaluated configurations are expressed or implied.

Details of the evaluated configurations are as follows:

Evaluated Configuration #1:

- FortiSandbox 1500G, using an AMD EPYC 7313P CPU on the Zen 3 microarchitecture
- FortiSandbox version 4.4.6-build 4527
- Non-TOE hardware of Fortinet FortiAnalyzer (FAZ) for remote logging

The FortiSandbox is put in CC mode by issuing the CLI command “cc-mode-conf –e”.

HTTP GUI, Telnet, TFTP are disabled in CC mode by default.

A signed certificate from a trusted CA is loaded to replace the default Fortinet CA signed certificate.

SSH administrative access is enabled via the web based manager. Under system, interfaces, port1, double click on port1 and in the “Access Rights” box check the “SSH” box to enable.

A pre-login banner is enabled via the administrative GUI. Select “Login Disclaimer” and click on the “Show disclaimer on login”.

The FortiAnalyzer (FAZ) remote logging server is set up:

a) FAZ server certificate is generated with a 3rd party tool
b) The root CA/subordinate CA certificates are imported into the FAZ and the root CA certificate is imported into the FortiSandbox
c) The OFTP setting on the FAZ is configured via CLI with “config sys certificate local”, “edit oftp-server”, “set private-key”, “set certificate”, “end”. “configy sys certificate OFTP”, “set mode local”, “set local ‘oftp-server’”

The FortiAnalyzer local log setting is configured via CLI with “config sys locallog fortianalyzer setting”, “set status disable”, “end”.

The FAZ root CA certificate is downloaded via the administrative GUI under System Settings > Certificates > CA Certificates.

The CA certificate for the FAZ is imported to the FortiSandbox via the administrative GUI under Certificates > Create New/Import > CA Certificate

The FAZ is enabled via the FortiSandbox administrative GUI under Log & Report > Log Servers > Create New.

On the FAZ, the FortiSandbox is added to the ADOM via the administrative GUI under Device Manager > Add Device.

Certificate Revocation Lists (CRL) are configured as:

a) Via the administrative GUI under System > Certificates, a .crl file is added by selecting “Create New/Import”.
b) At the CLI the below two commands are issued while substituting for the correct IP address and file names:
    i. cert-crl -nca-crl -ihttp://172.25.176.12/ca.crl.pem -t3
    ii. cert-crl -nint-crl -ihttp://172.25.176.12/intl.crl.pem -t3

Evaluated Configuration #2:

- FortiSandbox 500G, using an AMD EPYC 3251 CPU on the Zen microarchitecture
- FortiSandbox version 4.4.6-build 4527
- Non-TOE hardware of Fortinet FortiAnalyzer (FAZ) for remote logging

*** - All other configuration items identical to “Evaluated Configuration #1” shown above.

Evaluated Configuration #3:

- FortiSandbox 3000F, using an AMD EPYC 7402 CPU on the Zen 2 microarchitecture
- FortiSandbox version 4.4.6-build 4527
- Non-TOE hardware of Fortinet FortiAnalyzer (FAZ) for remote logging

*** - All other configuration items identical to “Evaluated Configuration #1” shown above.

Evaluated Configuration #4:

- FortiSandbox VM virtual appliance
- FortiSandbox version 4.4.6-build 4527 VM image for ESXi virtualization servers
- Virtualized environment of VMware ESXi 8.0 on Intel Xeon E5-2630v3, 8 Cores, 2.10GHz processor (Broadwell) processor
- Non-TOE hardware of Fortinet FortiAnalyzer (FAZ) for remote logging

*** - All other configuration items identical to “Evaluated Configuration #1” shown above.

1.4.3 TOE Logical Scope

The logical boundary of the TOE include those security functions implemented exclusively by the TOE. These security functions are listed in Section 1.3.3 above and are further described in the following subsections. A more detailed description of the implementation of these security functions are provided in Section 7 “TOE Summary Specification”.

1.4.3.1 Audit

- The TOE will audit all events and information defined in Table 4: Auditable Events.
- The TOE will also include the identity of the user that caused the event (if applicable), date and time of the event, type of event, and the outcome of the event.
- The TOE protects storage of audit information from unauthorized deletion
- The TOE prevents unauthorized modifications to the stored audit records.
- The TOE supports transmitting audit data to an external IT entity using TLS protocol.

1.4.3.2 Cryptographic Operations

The TSF performs cryptographic operations as defined in the TSS section 7.2. This includes:

- key generation for TLS and SSH using RSA, ECC, and FFC;
  - RSA with key sizes of 2048 bits
  - ECC over NIST curves P-256, P-384, and/or P-521
  - FFC using 2048-bit or greater keys in accordance with FIPS PUB 186-4
  - FFC using safe-primes according to NIST SP 800-56A r3
- key establishment in SSH and TLS sessions using ECDSA or DH/ECDH;
  - ECC according to NIST SP 800-56A rev3
  - FFC according to FIPS PUB 186-4 and NIST SP 800-56A rev3
  - FFC using Safe Primes, according to NIST SP 800-56A
- bulk encryption using AES in CBC or GCM modes with 128 or 256 bit keys.
- Cryptographic signature generation and verification using RSA or ECDSA
  - RSA with modulus 2048-bit according to FIPS PUB 186-4 section 5.5
  - ECDSA over curves NIST P-256, P-384, and/or P-521, according to FIPS PUB 186-4 section 6 and Appendix D
- Cryptographic hashing operations using SHA-1, SHA-2-256, SHA-2-384, and/or SHA-2-512
  - Keyed Hashing operations using the same underlying hashes in an HMAC function: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512

For details on the cryptographic operations performed by the TOE and the certificates which govern those operations, please see [ST] section 7.2

The TSF zeroizes all plaintext secret and private cryptographic keys and CSPs once they are no longer required.

1.4.3.3 Identification and Authentication

The TSF supports passwords consisting of alphanumeric and special characters. The TSF also allows administrators to set a minimum password length and support passwords with 15 characters or more.

The TSF requires all administrative-users to authenticate before allowing the user to perform any actions other than viewing the warning banner. The TOE may take certain automatic actions prior to authenticating any user:

- generating ephemeral session keys for SSH or TLS sessions, when any user attempts to connect.
- displaying the configured advisory and consent message, in accordance with the "Access Banner" requirement.

These TSF operations occur before identification and authentication.

1.4.3.4 Security Management

The TOE provides management functionality over both local and remote interfaces.

The local interface to a standalone hardware TOE is a dedicated physical port, which proxies direct console interface to the TOE's CLI.

Local access to a virtual TOE is through the ESXi console via URL.

For both standalone hardware TOEs and virtual TOEs, the remote interfaces are an SSH connection to the CLI and a web GUI accessed over TLS/HTTPS.

The TOE supports one default fully privileged user account, Admin, which corresponds to the PP defined Security Administrator. The Admin account has full privileges to all administrative functions.

The TOE supports an additional role: "TOE Users", who may be configured with none, some, or all of the permissions of a "Security Administrator". This enables administrators to delegate some or all of the tasks necessary to manage the TOE to roles with lesser permissions. Collectively, all user accounts with administrative permissions are "administrators".

1.4.3.5 Protection of the TSF

The TSF prevents the reading of all pre-shared keys, symmetric keys, private keys, and plaintext-passwords.

The TOE provides reliable time stamps for itself.

The TOE runs a suite of self-tests during initial start-up (on power on), and when cryptographic operations are performed to demonstrate the correct operation of the TSF.

The TOE provides a means to verify firmware/software updates to the TOE using a digital signature mechanism prior to installing those updates.

1.4.3.6 TOE Access

The TOE, for local interactive sessions, terminates the session after an Authorized Administrator-specified period of session inactivity.

The TOE terminates a remote interactive session after an Authorized Administrator-configurable period of session inactivity.

The TOE allows Administrator-initiated termination of the Administrator's own interactive session.

Before establishing an administrative user session, the TOE displays a Security Administrator-specified advisory notice and consent warning message regarding use of the TOE.

1.4.3.7 Trusted Path/Channels

The TOE is capable of using TLS to provide a trusted communication channel between itself and all authorized IT entities.

The TOE permits the TSF, or the authorized IT entities to initiate communication via the trusted channel.

The TOE is capable of using SSH or TLS/HTTPS to provide a communication path between itself and authorized remote Administrators.

The TOE permits remote administrators to initiate communication via the trusted path.

The TOE requires the use of the trusted path for initial administrator authentication and all remote administration actions.

1.5 Notation, formatting, and conventions

The notation, formatting, and conventions used in this Security Target are defined below; these styles and clarifying information conventions were developed to aid the reader.

Where necessary, the ST author has added application notes to provide the reader with additional details to aid understanding; they are italicized and usually appear following the element needing clarification.

The notation conventions that refer to iterations, assignments, selections, and refinements made in this Security Target are in reference to SARs and SFRs taken directly from CC Part 2 and Part 3 as well as any SFRs and SARs taken from a Protection Profile.

The notation/formatting used by the PP authors to indicate assignments, selections, and refinements of SARs and SFRs from CC Part 2 and Part 3 have not been carried forward in the ST with the exception of refinement (removal) indicated by the strikethrough text. .

The CC permits four component operations (assignment, iteration, refinement, and selection) to be performed on requirement components. These operations are defined in Common Criteria, Part 1; Section 8.1, “Operations” as:

- Iteration: allows a component to be used more than once with varying operations;
- Assignment: allows the specification of parameters;
- Selection: allows the specification of one or more items from a list; and
- Refinement: allows the addition of details.

Iterations performed by the ST author are indicated by a number in parenthesis following the requirement number, e.g., FIA_UAU.1.1(1); the iterated requirement titles are similarly indicated, e.g., FIA_UAU.1(1). Iterations performed by the PP author are indicated by a slash followed by a short description, e.g. FCS_COP.1/Hash.

Assignments are identified with **bold text.**

Selections are identified with <u>underlined text</u>**.**

Refinements that add text use ***bold and italicized text*** to identify the added text***.*** Refinements that remove text is identified with ~~***strikeout, bold, and italicized text***~~ of the removed text.

2. Conformance Claims

2.1 Common Criteria Conformance Claims

This Security Target and TOE are conformant to the Common Criteria Version 3.1 Release 5, CC Part 2 extended [C2], and CC Part 3 conformant [C3].

2.2 Conformance to Protection Profiles

This Security Target claims exact compliance to the collaborative Protection Profile for Network Devices, Version 3.0e, dated December 6, 2023 [cPP]. This Protection Profile will be referred to as cPP or PP for convenience throughout this Security Target.

Table 1 below lists the Technical Decisions published for [cPP] and includes an indication of their applicability to the TOE:

Table 1: [cPP] Technical Decisions			
TD	TD Title	Applies to TOE?	Rationale
TD0836	NIT Technical Decision: Redundant Requirements in FPT_TST_EXT.1	Applies to all TOEs	Modification of a mandatory SFR.
TD0900	NIT Technical Decision: Clarification to Local Administrator Access in FIA_UIA_EXT.1.3	Applies to all TOEs	Modification of a mandatory SFR
TD0921	NIT Technical Decision: Addition of FIPS PUB 186-5 and Correction of Assignment	Applies to all TOEs	Modification of a mandatory SFR
TD0923	NIT Technical Decision: Auditable Event for FAU_STG_EXT.1 in FAU_GEN.1.2	Applies to all TOEs	Modification of application notes and selection criteria for a mandatory SFR

2.3 Conformance to Security Packages

This Security Target claims exact conformance to the following security functional requirements package:

- Functional Package for Secure Shell (SSH), Version: 1.0, 2021-05-13 [SSH]

Table 2 below lists the Technical Decisions published for the Functional Package for SSH Version 1.0 [SSH], and includes an indication of their applicability to the TOE:

Table 2: [SSH] Technical Decisions			
TD	TD Title	TOE Applicability	Rationale
TD0682	Addressing Ambiguity in FCS_SSHS_EXT.1 Tests	Applies to all TOEs	Test modification.
TD0695	Choice of 128 or 256 bit size in AES-CTR in SSH Functional Package.	Applies to all TOEs	Clarification for FCS_COP.1
TD0732	FCS_SSHS_EXT.1.3 Test 2 Update	Applies to all TOEs	Test modification.
TD0777	Clarification to Selections for Auditable Events for FCS_SSH_EXT.1	Applies to all TOEs	Clarification for audit requirements.
TD0909	Updates to FCS_SSH_EXT.1.1 App Note in SSH FP 1.0	Applies to all TOEs	Modification of application notes.

2.4 Conformance Claims Rationale

To demonstrate that exact conformance is met, this rationale shows all threats are addressed, all OSP are satisfied, no additional assumptions are made, all objectives have been addressed, and all applicable SFRs and SARs have been instantiated.

The following address the completeness of the threats, OSP, and objectives, limitations on the assumptions, and instantiation of the SFRs and SARs:

- Threats
  - All threats defined in the cPP;
  - No additional threats have been defined in this ST.
- Organizational Security Policies
  - All OSP defined in the cPP are carried forward to this ST;
  - No additional OSPs have been defined in this ST.
- Assumptions
  - All assumptions defined in the cPP for a standalone or virtual TOE are carried forward to this ST;
  - No additional assumptions for the operational environment have been defined in this ST.
- Objectives
  - All objectives defined in the cPP for a standalone or virtual TOE are carried forward to this ST. Optional and selection based SFRs defined in the cPP are carried forward to this Security Target as required by the cPP.
- All mandatory SFRs and SARs defined in the cPP are carried forward to this Security Target.

Rationale presented in the body of this ST shows all assumptions on the operational environment have been upheld, all the OSP are enforced, all defined objectives have been met and these objectives counter the defined threats.

Additionally, all SFRs and SARs defined in the cPP have been properly instantiated in this Security Target; therefore, this ST demonstrates exact conformance to the cPP and Functional Package for Secure Shell.

3. Security Problem Definition

3.1 Threats

The following section defines the security threats for the TOE, characterized by a threat agent, an asset, and an adverse action of that threat agent on that asset. These threats are taken directly from the PP unchanged.

T.UNAUTHORIZED_ADMINISTRATOR_ACCESS

Threat agents may attempt to gain Administrator access to the Network Device by nefarious means such as masquerading as an Administrator to the device, masquerading as the device to an Administrator, replaying an administrative session (in its entirety, or selected portions), or performing man-in-the-middle attacks, which would provide access to the administrative session, or sessions between Network Devices. Successfully gaining Administrator access allows malicious actions that compromise the security functionality of the device and the network on which it resides.

T.WEAK_CRYPTOGRAPHY

Threat agents may exploit weak cryptographic algorithms or perform a cryptographic exhaust against the key space. Poorly chosen encryption algorithms, modes, and key sizes will allow attackers to compromise the algorithms, or brute force exhaust the key space and give them unauthorized access allowing them to read, manipulate and/or control the traffic with minimal effort.

T.UNTRUSTED_COMMUNICATION_CHANNELS

Threat agents may attempt to target Network Devices that do not use standardized secure tunnelling protocols to protect the critical network traffic. Attackers may take advantage of poorly designed protocols or poor key management to successfully perform man-in-the-middle attacks, replay attacks, etc. Successful attacks will result in loss of confidentiality and integrity of the critical network traffic, and potentially could lead to a compromise of the Network Device itself.

T.WEAK_AUTHENTICATION_ENDPOINTS

Threat agents may take advantage of secure protocols that use weak methods to authenticate the endpoints, e.g. a shared password that is guessable or transported as plaintext. The consequences are the same as a poorly designed protocol, the attacker could masquerade as the Administrator or another device, and the attacker could insert themselves into the network stream and perform a man-in-the-middle attack. The result is the critical network traffic is exposed and there could be a loss of confidentiality and integrity, and potentially the Network Device itself could be compromised.

T.UPDATE_COMPROMISE

Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Nonvalidated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.

T.UNDETECTED_ACTIVITY

Threat agents may attempt to access, change, and/or modify the security functionality of the Network Device without Administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the Administrator would have no knowledge that the device has been compromised.

T.SECURITY_FUNCTIONALITY_COMPROMISE

Threat agents may compromise credentials and device data enabling continued access to the Network Device and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or device credentials for use by the attacker. Threat agents may also be able to take advantage of weak administrative passwords to gain privileged access to the device.

T.SECURITY_FUNCTIONALITY_FAILURE

An external, unauthorized entity could make use of failed or compromised security functionality and might therefore subsequently use or abuse security functions without prior authentication to access, change or modify device data, critical network traffic or security functionality of the device.

3.2 Organizational Security Policies

The following section defines the organizational security policies which are a set of rules, practices, and procedures imposed by an organization to address its security needs. These threats are taken directly from the PP unchanged.

P.ACCESS_BANNER

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

3.3 Assumptions

This section describes the assumptions on the operational environment in which the TOE is intended to be used. It includes information about the physical, personnel, and connectivity aspects of the environment. The operational environment must be managed in accordance with the provided guidance documentation. The following table defines specific conditions that are assumed to exist in an environment where the TOE is deployed. These assumptions are taken directly from the PP unchanged.

A.PHYSICAL_PROTECTION

The Network Device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, the cPP does not include any requirements on physical tamper protection or other physical attack mitigations. The cPP does not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device. For vNDs, this assumption applies to the physical platform on which the VM runs.

A.LIMITED_FUNCTIONALITY

The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the device should not provide a computing platform for general purpose applications (unrelated to networking functionality).

If a virtual TOE evaluated as a pND, following Case 2 vNDs as specified in Section 1.2, the VS is considered part of the TOE with only one vND instance for each physical hardware platform. The exception being where components of a distributed TOE run inside more than one virtual machine (VM) on a single VS. In Case 2 vND, no non-TOE guest VMs are allowed on the platform.

A.NO_THRU_TRAFFIC_PROTECTION

A standard/generic Network Device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the Network Device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the Network Device, destined for another network entity, is not covered by the ND cPP. It is assumed that this protection will be covered by cPPs and PP-Modules for particular types of Network Devices (e.g., firewall).

A.TRUSTED_ADMINISTRATOR

The Security Administrator(s) for the Network Device are assumed to be trusted and to act in the best interest of security for the organization. This includes appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The Network Device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device.

For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are expected to fully validate (e.g. offline verification) any CA certificate (root CA certificate or intermediate CA certificate) loaded into the TOE's trust store (aka 'root store', ' trusted CA Key Store', or similar) as a trust anchor prior to use (e.g. offline verification).

A.REGULAR_UPDATES

The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

A.ADMIN_CREDENTIALS_SECURE

The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.

A.RESIDUAL_INFORMATION

The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

A.VS_TRUSTED_ADMINISTRATOR

The Security Administrators for the VS are assumed to be trusted and to act in the best interest of security for the organization. This includes not interfering with the correct operation of the device. The Network Device is not expected to be capable of defending against a malicious VS Administrator that actively works to bypass or compromise the security of the device.

A.VS_REGULAR_UPDATES

The VS software is assumed to be updated by the VS Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

A.VS_ISOLATON

For vNDs, it is assumed that the VS provides, and is configured to provide sufficient isolation between software running in VMs on the same physical platform. Furthermore, it is assumed that the VS adequately protects itself from software running inside VMs on the same physical platform.

A.VS_CORRECT_CONFIGURATION

For vNDs, it is assumed that the VS and VMs are correctly configured to support ND functionality implemented in VMs.

4. Security Objectives

4.1 Security Objectives for the TOE

None.

4.2 Security Objectives for the Operational Environment

OE.PHYSICAL

Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

OE.NO_GENERAL_PURPOSE

There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE. Note: For vNDs the TOE includes only the contents of the its own VM, and does not include other VMs or the VS.

OE.NO_THRU_TRAFFIC_PROTECTION

The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.

OE.TRUSTED_ADMIN

Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner. For vNDs, this includes the VS Administrator responsible for configuring the VMs that implement ND functionality.

For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE’s trust store in case such certificate can no longer be trusted.

OE.UPDATES

The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.

OE.ADMIN_CREDENTIALS_SECURE

The Administrator’s credentials (private key) used to access the TOE must be protected on any other platform on which they reside.

OE.RESIDUAL_INFORMATION

The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment. For vNDs, this applies when the physical platform on which the VM runs is removed from its operational environment.

OE.VM_CONFIGURATION

For vNDs, the Security Administrator ensures that the VS and VMs are configured to

- reduce the attack surface of VMs as much as possible while supporting ND functionality (e.g., remove unnecessary virtual hardware, turn off unused inter-VM communications mechanisms), and
- correctly implement ND functionality (e.g., ensure virtual networking is properly configured to support network traffic, management channels, and audit reporting).

The VS should be operated in a manner that reduces the likelihood that vND operations are adversely affected by virtualisation features such as cloning, save/restore, suspend/resume, and live migration.

If possible, the VS should be configured to make use of features that leverage the VS's privileged position to provide additional security functionality. Such features could include malware detection through VM introspection, measured VM boot, or VM snapshot for forensic analysis.

5. Extended Components Definition

As stated in Section 2, this Security Target claims exact conformance to the referenced Base-PP and SSH functional package. As such, the extended components definition is contained in the claimed Base-PP and SSH functional package. Both the cPP, and the Functional Package for Secure Shell are Part 3 conformant; therefore, no extended SARs are defined.

6. Security Functional Requirements

This section describes the functional requirements for the TOE. The Security Functional Requirement components in this Security Target are CC Part 2 conformant or CC Part 2 extended as defined in Section 2, Conformance Claims. Operations performed in the ST are denoted according to the formatting conventions in Section 1.5.

Table 3: Security Functional Requirements

SFR	Description
FAU_GEN.1	Audit Data Generation
FAU_GEN.2	User Identity Association
FAU_STG.1	Protected audit trail storage
FAU_STG_EXT.1	Protected Audit Event Storage
FCS_CKM.1	Cryptographic Key Generation (Refinement)
FCS_CKM.2	Cryptographic Key Establishment (Refinement)
FCS_CKM.4	Cryptographic Key Destruction
FCS_COP.1/DataEncryption	Cryptographic Operation (AES Data Encryption/Decryption)
FCS_COP.1/SigGen	Cryptographic Operation (Signature Generation and Verification)
FCS_COP.1/Hash	Cryptographic Operation (Hash Algorithm)
FCS_COP.1/KeyedHash	Cryptographic Operation (Keyed Hash Algorithm)
FCS_HTTPS_EXT.1	HTTPS Protocol
FCS_RBG_EXT.1	Random Bit Generation
FCS_SSH_EXT.1	SSH Protocol
FCS_SSHS_EXT.1	SSH Protocol - Server
FCS_TLSC_EXT.1	TLS Client Protocol Without Mutual Authentication
FCS_TLSS_EXT.1	TLS Server Protocol Without Mutual Authentication
FIA_AFL.1	Authentication Failure Management (Refinement)
FIA_PMG_EXT.1	Password Management
FIA_UIA_EXT.1	User Identification and Authentication
FIA_UAU.7	Protected Authentication Feedback
FIA_X509_EXT.1/Rev	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication
FIA_X509_EXT.3	X.509 Certificate Requests
FMT_MOF.1/Services	Management of security functions behavior
FMT_MOF.1/ManualUpdate	Management of security functions behavior

Table 3: Security Functional Requirements	
SFR	Description
FMT_MTD.1/CoreData	Management of TSF Data
FMT_MTD.1/CryptoKeys	Management of TSF Data
FMT_SMF.1	Specification of Management Functions
FMT_SMR.2	Restrictions on security roles
FPT_APW_EXT.1	Protection of Administrator Passwords
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 (Extended)
FPT_TUD_EXT.1	Trusted Update
FTA_SSL_EXT.1	TSF-initiated Session Locking
FTA_SSL.3	TSF-initiated Termination (Refinement)
FTA_SSL.4	User-initiated Termination (Refinement)
FTA_TAB.1	Default TOE Access Banners (Refinement)
FTP_ITC.1	Inter-TSF Trusted Channel (Refinement)
FTP_TRP.1/Admin	Trusted Path (Refinement)

6.1 Security Audit (FAU)

6.1.1 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 shut-down 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 4.

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 ~~(if applicable),~~ 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 4.

Table 4: Auditable Events		
SFR	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.2	None.	None.
FAU_STG.1	None.	None.
FAU_STG_EXT.1	Configuration of local audit settings.	Identity of account making changes to the audit configuration.
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.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session.	Reason for failure
FCS_RBG_EXT.1	None.	None.
FCS_SSH_EXT.1	[Failure to establish SSH connection]	[Reason for failure and Non-TOE endpoint of attempted connection (IP Address)]
	[Establishment of SSH connection]	[Non-TOE endpoint of attempted connection (IP Address)]
	[Termination of SSH connection session,]	[Non-TOE endpoint of attempted connection (IP Address)]
	[Dropping of packet(s) outside defined size limits]	[Packet size]
FCS_SSHS_EXT.1	No events specified	None
FCS_TLSC_EXT.1	Failure to establish a TLS Session	Reason for failure
FCS_TLSS_EXT.1	Failure to establish a TLS Session	Reason for failure
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_UAU.7	None.	None.

Table 4: Auditable Events		
SFR	Auditable Events	Additional Audit Record Contents
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/Services	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	None.	None.
FMT_MTD.1/CryptoKeys	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMR.2	None.	None.
FPT_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_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).
FTA_SSL_EXT.1	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.	• None

Table 4: Auditable Events		
SFR	Auditable Events	Additional Audit Record Contents
	• Termination of the trusted channel. • Failure of the trusted channel functions.	• None • Reason for failure
FTP_TRP.1/Admin	• Initiation of the trusted path. • Termination of the trusted path. • Failure of the trusted path functions.	• None • None • Reason for failure

6.1.2 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.

6.1.3 FAU_STG.1 Protected Audit Trail Storage

FAU_STG.1.1

The TSF shall protect the stored audit records in the audit trail from unauthorised deletion.

FAU_STG.1.2

The TSF shall be able to prevent unauthorised modifications to the stored audit records in the audit trail.

6.1.4 FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1

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

FAU_STG_EXT.1.2

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

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

FAU_STG_EXT.1.3

The TSF shall maintain a [log file] of audit records in the event that an interruption of communication with the remote audit server occurs.

FAU_STG_EXT.1.4

The TSF shall be able to store [persistent] audit records locally with a minimum storage size of [**1 GB**].

FAU_STG_EXT.1.5

The TSF shall [overwrite previous audit records according to the following rule:[**delete the oldest 100mb log file, then create a new local log file**],[**continue transmission of new log data are**

to the Audit Log Server via trusted channel]] when the local storage space for audit data is full.

FAU_STG_EXT.1.6

The TSF shall provide the following mechanisms for administrative access to locally stored audit records [manual export, ability to view locally].

6.2 Cryptographic Support (FCS)

6.2.1 FCS_CKM.1 Cryptographic Key Generation (Refinement)

FCS_CKM.1.1[2]

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

- RSA schemes using cryptographic key sizes of [**2048 bits**] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 or FIPS PUB 186-5, "Digital Signature Standard (DSS)", A.1;
- 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, or FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2, or ISO/IEC 14888-3, "IT Security techniques - Digital signatures with appendix - Part 3: Discrete logarithm based mechanisms", Section 6.6.;
- FFC Schemes using 'safe-prime' groups that meet the following: "NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and [RFC 3526].

]

6.2.2 FCS_CKM.2 Cryptographic Key Establishment (Refinement)

FCS_CKM.2.1

The TSF shall perform cryptographic key establishment in accordance with a specified cryptographic key establishment method: [

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

]

6.2.3 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, a new value of the key]]

[2] Modified by TD0921, which supersedes TD0918

- 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, a new value of the key]]];

that meets the following: No Standard.

6.2.4 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, GCM] mode and cryptographic key sizes [128 bits, 256 bits] that meet the following: AES as specified in ISO 18033-3, [CBC as specified in ISO 10116, GCM as specified in ISO 19772].

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

FCS_COP.1.1/SigGen[3]

The TSF shall perform cryptographic signature services (generation and verification) in accordance with a specified cryptographic algorithm [

- RSA Digital Signature Algorithm,
- Elliptic Curve Digital Signature Algorithm

]

and cryptographic key sizes [

- For RSA: [**modulus 2048 bits**],
- For ECDSA: [**256 bits, 384 bits, 512 bits**]

]

that meet the following: [

- For RSA schemes: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 5.5, using PKCS #1 v2.1 or FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 5.4 using PKCS #1 v2.2 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3,
- For ECDSA schemes implementing [P-256, P-384, P-521] curves that meet the following: FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Section 6 and Appendix D, Implementing “NIST Recommended" curves” or FIPS PUB 186-5, "Digital Signature Standard (DSS)", Section 6 and NIST SP 800-186 Section 3.2.1, Implementing Weierstrass curves; or ISO/IEC 14888-3, "IT Security techniques - Digital signatures with appendix - Part 3: Discrete logarithm based mechanisms", Section 6.6.

].

6.2.6 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~~

[3] Modified by TD0921 which supersedes TD0918

~~[assignment: cryptographic key sizes]~~ and message digest sizes [160, 256, 384, 512] bits that meet the following: ISO/IEC 10118-3:2004.

6.2.7 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”.

6.2.8 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 ~~protocol~~ using TLS.

6.2.9 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 [[**1**] platform-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.

6.2.10 FCS_SSH_EXT.1 SSH Protocol

FCS_SSH_EXT.1.1

The TOE shall implement SSH acting as a [server] in accordance with that complies with RFCs 4251, 4252, 4253, 4254 [4256, 6668, 8268, 8308, 8332] and [no other standard].

FCS_SSH_EXT.1.2

The TSF shall ensure that the SSH protocol implementation supports the following authentication methods: [

- “keyboard interactive” (RFC 4256)

]

] and no other methods.

FCS_SSH_EXT.1.3 [TD0732[4]]

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

FCS_SSH_EXT.1.4

[4] The Test Evaluation Activity for this SFR was modified by TD0732.

The TSF shall protect data in transit from unauthorised disclosure using the following mechanisms: [

- aes128-cbc (RFC 4253),
- aes256-cbc (RFC 4253),

] and no other mechanisms.

FCS_SSH_EXT.1.5

The TSF shall protect data in transit from modification, deletion, and insertion using: [

- hmac-sha2-256 (RFC 6668),
- hmac-sha2-512 (RFC 6668),

] and no other mechanisms.

FCS_SSH_EXT.1.6

The TSF shall establish a shared secret with its peer using: [

- diffie-hellman-group14-sha256 (RFC 8268),

] and no other mechanisms.

FCS_SSH_EXT.1.7

The TSF shall use SSH KDF as defined in [RFC 4253 (Section 7.2)] to derive the following cryptographic keys from a shared secret: session keys.

FCS_SSH_EXT.1.8

The TSF shall ensure that [

- a rekey of the session keys,

] occurs when any of the following thresholds are met:

- one hour connection time
- no more than one gigabyte of transmitted data,
- or no more than one gigabyte of received data.

6.2.11 FCS_SSHS_EXT.1 SSH Protocol - Server

FCS_SSHS_EXT.1.1 [TD0682[5]]

The TSF shall authenticate itself to its peer (SSH Client) using: [

- rsa-sha2-256 (RFC 8332),

].

6.2.12 FCS_TLSC_EXT.1 TLS Client Protocol Without Mutual Authentication

FCS_TLSC_EXT.1.1 The TSF shall implement [TLS 1.3 (RFC 8446),TLS 1.2 (RFC 5246)] supporting the following ciphersuites:

[

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384

[5] The Test Evaluation Activities for this SFR were modified by TD0682.

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_CCM_SHA256
- TLS_AES_128_CCM_8_SHA256

] *and no other ciphersuites*.

FCS_TLSC_EXT.1.2

The TSF shall verify that the presented identifier matches [the reference identifier per RFC 6125 Section 6, IPv4 address in the SAN, IPv6 address in the SAN].

FCS_TLSC_EXT.1.3

The TSF shall not establish a trusted channel if the server certificate is invalid [

- Without any administrator override mechanism.

].

FCS_TLSC_EXT.1.4

The TSF shall [present the Supported Groups Extension with the following curves/groups: [secp256r1, secp384r1, secp521r1,] and no other curves/groups] in the Client Hello.

FCS_TLSC_EXT.1.5

The TSF shall [

- present the signature_algorithms extension with support for the following algorithms:

[

  - rsa_pkcs1 with sha256(0x0401),
  - rsa_pkcs1with sha384(0x0501),
  - rsa_pkcs1 with sha512(0x0601),
  - ecdsa_secp256r1 with sha256(0x0403),
  - ecdsa_secp384r1 with sha384(0x0503),
  - ecdsa_secp521r1 with sha512(0x0603),
  - rsa_pss_rsae with sha256(0x0804),
  - rsa_pss_rsae with sha384(0x0805),
  - rsa_pss_rsae with sha512(0x0806),
  - rsa_pss_pss with sha256(0x0809),
  - rsa_pss_pss with sha384(0x080a),
  - rsa_pss_pss with sha512(0x080b)

] and no other algorithms;

].

FCS_TLSC_EXT.1.6

The TSF [does not provide] the ability to configure the list of supported ciphersuites as defined in FCS_TLSC_EXT.1.1.

FCS_TLSC_EXT.1.7
The TSF shall prohibit the use of the following extensions:

- Early data extension
- Post-handshake client authentication according to RFC 8446, Section 4.2.6.

FCS_TLSC_EXT.1.8
The TSF shall [not use PSKs, only use PSKs in TLS 1.3 session resumption with forward secrecy].

FCS_TLSC_EXT.1.9 The TSF shall [reject [TLS 1.2, TLS 1.3] renegotiation attempts].

6.2.13 FCS_TLSS_EXT.1 TLS Server Protocol

FCS_TLSS_EXT.1.1

The TSF shall implement [TLS 1.3 (RFC 8446), TLS 1.2 (RFC 5246)] and reject all other TLS and SSL versions. The TLS implementation will support the following ciphersuites:

[

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_CCM_SHA256
- TLS_AES_128_CCM_8_SHA256

] *and no other ciphersuites*.

FCS_TLSS_EXT.1.2

The TSF shall authenticate itself using X.509 certificate(s) using [RSA with key size [2048] bits; ECDSA over NIST curves [secp256r1, secp384r1, secp521r1] and no other curves].

FCS_TLSS_EXT.1.3

The TSF shall perform key exchange using: [

- EC Diffie-Hellman key agreement over NIST curves [secp256r1, secp384r1, secp521r1] and no other curves;
- Diffie-Hellman parameters [of size 2048 bits]

].

FCS_TLSS_EXT.1.4

The TSF shall support [session resumption based on session tickets according to RFC 5077 (TLS 1.2), session resumption according to RFC 8446 (TLS 1.3)].

FCS_TLSS_EXT.1.5 The TSF [does not provide] the ability to configure the list of supported ciphersuites as defined in FCS_TLSS_EXT.1.1.

FCS_TLSS_EXT.1.6 The TSF shall prohibit the use of the following extensions:

- Early data extension

FCS_TLSS_EXT.1.7 The TSF shall [not use PSKs].

FCS_TLSS_EXT.1.8 The TSF shall [reject [TLS 1.2, TLS 1.3] renegotiation attempts].

6.3 Identification and Authentication (FIA)

6.3.1 FIA_AFL.1 Authentication Failure Management (Refinement)

FIA_AFL.1.1

The TSF shall detect when an Administrator configurable positive integer within [**3-20**] 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 a remote session using any authentication method that involves a password until an Administrator defined time period has elapsed].

6.3.2 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1

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

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers and the following special characters: [“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, “>”, “[space]”, “””, “‘”, “+”, “-“, “/”, “:”, “;”, “<”, “=”, “?”, “[“, “\”, “]”, “[underscore]”, “`”, “{“, “|”, “}”, “~”];

b) Minimum password length shall be configurable to between [**6**] and [**64**] characters.

6.3.3 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;
- [automated generation of cryptographic keys, **respond to ICMP Echo requests with ICMP Echo Replies**].

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_UIA_EXT.1.3[6]

The TSF shall provide the following remote authentication mechanisms [Web GUI password, SSH password] and [no other mechanism]. The TSF shall provide the following local authentication mechanisms [password-based].

FIA_UIA_EXT.1.4

The TSF shall authenticate any administrative user's claimed identity according to each authentication mechanism specified in FIA_UIA_EXT.1.3.

6.3.4 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.

6.3.5 FIA_X509_EXT.1/Rev X.509 Certificate Validation

FIA_X509_EXT.1.1/Rev

The TSF shall validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certification path validation supporting a minimum path length of three certificates.
- The certification path must terminate with a trusted CA certificate designated as a trust anchor.
- The TSF shall validate a certification path by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.
- The TSF shall validate the revocation status of the certificate using [a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3,]
- 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 DTLS/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 DTLS/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.

[6] Modified by TD0900

6.3.6 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 [HTTPS, 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 [not accept the certificate].

6.3.7 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, 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.

6.4 Security Management (FMT)

6.4.1 FMT_MOF.1/ManualUpdate Management of Security Functions Behaviour

FMT_MOF.1.1/ManualUpdate

The TSF shall restrict the ability to enable the functions to perform manual updates to Security Administrators.

6.4.2 FMT_MOF.1/Services Management of Security Functions Behaviour

FMT_MOF.1.1/Services

The TSF shall restrict the ability to start and stop ~~the functions~~ services to Security Administrators.

6.4.3 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.

6.4.4 FMT_MTD.1/CryptoKeys Management of TSF data

FMT_MTD.1.1/CryptoKeys

The TSF shall restrict the ability to manage the cryptographic keys to Security Administrators.

6.4.5 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 remotely;
- Ability to configure the access banner;
- Ability to configure the remote session inactivity time before session termination;
- Ability to update the TOE, and to verify the updates using digital signature capability prior to installing those updates;
- [
    - Ability to start and stop services
    - Ability to manage the cryptographic keys;

- Ability to set the time which is used for time-stamps;
- Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;
- Ability to generate Certificate Signing Request (CSR) and process CA certificate response;
- Ability to administer the TOE locally;
- Ability to configure the authentication failure parameters for FIA_AFL.1;

6.4.6 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 remotely

are satisfied.

6.5 Protection of the TSF (FPT)

6.5.1 FPT_APW_EXT.1 Protection of Administrator Passwords

FPT_APW_EXT.1.1

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

FPT_APW_EXT.1.2

The TSF shall prevent the reading of plaintext administrative passwords.

6.5.2 FPT_SKP_EXT.1 Protection of TSF Data (for reading all symmetric keys)

FPT_SKP_EXT.1.1

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

6.5.3 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].

6.5.4 FPT_TST_EXT.1 TSF Testing (Extended)

FPT_TST_EXT.1.1 [TD0836[7]]

The TSF shall run a suite of the following self-tests:

[7] TD0836 implemented.

- During initial start-up (on power on) to verify the integrity of the TOE firmware and software;
- Prior to providing any cryptographic service and [on-demand] to verify correct operation of cryptographic implementation necessary to fulfil the TSF;
- [no other,] self-tests

to demonstrate the correct operation of the TSF.

FPT_TST_EXT.1.2 The TSF shall respond to [all failures, [**integrity failure, cryptographic self-test failure(s)**]] by [[**revert to a 'failure' mode which requires the administrator to manually reboot the device**]].

6.5.5 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] prior to installing those updates.

6.6 TOE Access (FTA)

6.6.1 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.

6.6.2 FTA_SSL.3 TSF-initiated Termination (Refinement)

FTA_SSL.3.1

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

6.6.3 FTA_SSL.4 User-initiated Termination (Refinement)

FTA_SSL.4.1

The TSF shall allow ~~user~~ Administrator-initiated termination of the ~~user's~~ Administrator's own interactive session.

6.6.4 FTA_TAB.1 Default TOE Access Banners (Refinement)

FTA_TAB.1.1

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

6.7 Trusted path/channels (FTP)

6.7.1 FTP_ITC.1 Inter-TSF Trusted Channel (Refinement)

FTP_ITC.1.1

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 ~~modification or~~ disclosure and detection of modification of the channel data.

FTP_ITC.1.2

The TSF shall permit [the TSF, 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 [**remote audit log storage**].

6.7.2 FTP_TRP.1/Admin Trusted Path (Refinement)

FTP_TRP.1.1/Admin

The TSF shall be capable of using [SSH, HTTPS] to provide a communication path between itself and authorized remote Administrators ~~users~~ 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 ~~users~~ to initiate communication via the trusted path.

FTP_TRP.1.3/Admin

The TSF shall require the use of the trusted path for initial Administrator authentication and all remote administration actions.

6.8 Security Assurance Requirements

This Security Target is conformant with the security assurance requirements specified in the cPP.

Table 5: 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)	Labeling of the TOE (ALC_CMC.1)
	TOE CM Coverage (ALC_CMS.1)
Tests (ATE)	Independent Testing – conformance (ATE_IND.1)
Vulnerability assessment (AVA)	Vulnerability Survey (AVA_VAN.1)

6.8.1 Extended Security Assurance Requirements

These requirements are taken directly from the cPP.

6.8.1.1 ASE: Security Target

The ST is evaluated as per ASE activities defined in the CEM. In addition, there may be Evaluation Activities specified within [SD] that call for necessary descriptions to be included in the TSS that are specific to the TOE technology type.

Appendix D provides a description of the information expected to be provided regarding the quality of entropy in the random bit generator.

6.8.1.1.1 ASE_TSS.1.1C Refinement

The TOE summary specification shall describe how the TOE meets each SFR. In the case of entropy analysis, the TSS is used in conjunction with required supplementary information on Entropy.

6.8.1.2 ADV: Development

The design information about the TOE is contained in the guidance documentation available to the end user as well as the TSS portion of the ST, and any required supplementary information required by this cPP that is not to be made public.

6.8.1.2.1 Basic Functional Specification (ADV_FSP.1)

The functional specification describes the TOE Security Functions Interfaces (TSFIs). It is not necessary to have a formal or complete specification of these interfaces. Additionally, because TOEs conforming to this cPP will necessarily have interfaces to the Operational Environment that are not directly invokable by TOE administrators, there is little point specifying that such interfaces be described in and of themselves since only indirect testing of such interfaces may be possible. For this cPP, the Evaluation Activities for this family focus on understanding the interfaces presented in the TSS in response to the functional requirements and the interfaces presented in the AGD documentation. No dedicated “functional specification” documentation is necessary to satisfy the Evaluation Activities specified in [SD].

The Security Target, AGD documentation, supplementary information, or combination of thereof constitutes “functional specification” documentation. This documentation must contain the description of all security-relevant interfaces.

The Evaluation Activities in [SD] are associated with the applicable SFRs; since these are directly associated with the SFRs, the tracing in element ADV_FSP.1.2D is implicitly already done and no additional documentation is necessary.

6.8.1.3 AGD: Guidance Documentation

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

Note that additional Evaluation Activities for the guidance documentation in the case of a distributed TOE are defined in section A.9.1.1.

6.8.1.4 Operational User Guidance (AGD_OPE.1)

The evaluator performs the CEM work units associated with the AGD_OPE.1 SAR. Specific requirements and EAs on the guidance documentation are identified (where relevant) in the individual EAs for each SFR. For the related evaluation activities, the evaluation evidence documents Security Target, AGD documentation (user guidance) and functional specification documentation (if provided) shall be used as input documents. Each input document is subject to ALC_CMS.1-2 requirements.

6.8.1.5 Preparative Procedures (AGD_PRE.1)

The evaluator performs the CEM work units associated with the AGD_PRE.1 SAR. Specific requirements and EAs on the preparative documentation are identified (and where relevant are captured in the Guidance Documentation portions of the EAs) in the individual EAs for each SFR.

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

In addition, the evaluator performs the EAs specified below.

6.8.1.6 ALC: Life-cycle Support

6.8.1.6.1 Labelling of the TOE (ALC_CMC.1)

When evaluating that the TOE has been provided and is labelled with a unique reference, the evaluator performs the work units as presented in the CEM.

6.8.1.6.2 TOE CM Coverage (ALC_CMS.1)

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

6.8.1.6.3 Basic flaw remediation (ALC_FLR.1)

When evaluating the developer's procedures regarding basic flaw remediation, the evaluator performs the work units as presented in the CEM.

6.8.1.6.4 Flaw reporting procedures (ALC_FLR.2)

When evaluating the developer's flaw reporting procedures, the evaluator performs the work units as presented in the CEM.

6.8.1.6.5 Systematic flaw remediation (ALC_FLR.3)

When evaluating the developer's procedures regarding systematic flaw remediation, the evaluator performs the work units as presented in the CEM.

6.8.1.7 Class ATE: Tests

6.8.1.7.1 Independent Testing – Conformance (ATE_IND.1)

The focus of the testing is to confirm that the requirements specified in the SFRs are being met. Additionally, testing is performed to confirm the functionality described in the TSS, as well as the dependencies on the Operational guidance documentation is accurate.

The evaluator performs the CEM work units associated with the ATE_IND.1 SAR. Specific testing requirements and EAs are captured for each SFR in Sections 2, 3 and 4.

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

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

6.8.1.8 Class AVA: Vulnerability Assessment

6.8.1.8.1 Vulnerability Survey (AVA_VAN.1)

[SD, Appendix A] provides a guide to the evaluator in performing a vulnerability analysis.

7. TOE Summary Specification

This section provides evaluators and potential consumers of the TOE with a high-level description of each SFR, thereby enabling them to gain a general understanding of how the TOE is implemented. These descriptions are intentionally not overly detailed, thereby disclosing no proprietary information. These sections refer to SFRs defined in Section 6, Security Functional Requirements.

The TOE consists of the following Security Functions:

- Security Audit
- Cryptographic Support
- Identification and Authentication
- Security Management
- Protection of the TSF
- TOE Access
- Trusted Path/Channels

7.1 Security Audit

7.1.1 Audit Data Generation

As either a virtual or standalone device, the TOE generates audit records residing in its local storage. The TOE generates audit records of security-relevant management functions on both its local and remote interfaces. The audit records that are generated include startup and shutdown of the TOE (the audit functions are not enabled or disabled separately), all administrative actions, and the specific events identified in Table 4 above. For all auditable events that involve a user (e.g. authentication and administration events), the user identity is captured in the audit record. For cryptographic key related events, the associated creation details (date, time) are recorded in the audit record.

When a CSR or CA is imported, the name assigned by the administrator is also logged. When CSRs or CAs are deleted, the name is logged. For SSH, the generation of the key is recorded along with associated identifying information: the date and time of the operation.

All audit records also include date and time of the event, type of event, subject identity, and the outcome of the event where appropriate. When CSRs are created, a new public/private keypair is generated. CSR-related keypairs are uniquely identified by the creation details: date and time, user who initiated the generation.

7.1.2 Audit Storage

The TOE simultaneously logs all generated audit messages both locally and remotely (as long as the trusted channel has been established). Audit records generated by the TOE without an active trusted channel are not queued to be sent, and are only viewable on the TOE itself. Security relevant audit records are stored in the underlaying file system as a persistent database file, and can be accessed from the ***Events*** submenu of the ***Log and Report*** tab of the TOE management interface. These are referred to as System Events or ***kevent*** log types. Only authorized administrators may access the TOE management interface, or use its defined commands to delete log files or view log records. No capability to modify the records is provided.

Access to the audit records is restricted only to properly authenticated and authorized TOE User or Security Administrator accounts with the proper permissions. The TOE does not permit the audit record database to be directly viewed, modified, or deleted via the underlying file system. Viewing, Exporting, or Deleting the audit records from the database is only permitted to properly

identified and authenticated security administrators via the TOE GUI. The audit record database itself is protected in the underlying filesystem with no access permissions.

The default settings for the TOE in CC mode, specify the TOE will log locally and will overwrite the oldest audit logs upon hitting the threshold of memory capacity. The TOE maintains 10 log files of 100mb each. When a log file reaches 100mb, it is “closed” and a new log file is opened. When the TOE needs to close the 10th log file, the oldest (“first”) log file is deleted, and a new file is opened. The TOE maintains a total local audit log file size capacity of 1GB. During this log rotation process, the TSF continues to transmit audit data to the configured audit log server via the trusted channel.

The TOE has configurable options for the remote storage of the audit events. These events are sent to one or more configured audit servers, in real-time, simultaneously with the audit records that are written locally. In the evaluated configuration, the audit server is a FAZ secured through the usage of TLS.

7.2 Cryptographic Support

7.2.1 Cryptographic Key Generation

The TOE uses two libraries: Fortinet FortiSandbox SSL Cryptographic Library v4.4 and Fortinet FortiSandbox FIPS Cryptographic Library v4.4. The SSL library is used to handle trusted channel and trusted path, while the FIPS library is used to handle all other cryptographic operations. When the TOE is executing in a virtualized environment, the alternative libraries (*-VM) are used instead. The TOE obtained the following certificates, which verify the cryptographic operation of the TOE across all cryptographic operations listed below:

The below identifies the algorithm certificates that apply to all cryptographic libraries.

Table 6: Algorithm Certificates

Functions	Standards	Certificates
Asymmetric key generation (FCS_CKM.1)		
RSA Schemes (2048 bits)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3	A7082 (SSL)
ECC Schemes (ECDSA P-256, P-384, P-521 curves)	FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4	A7080 (SSL-VM)
FFC Schemes (‘safe-prime’ groups (2048-bit MODP)	NIST SP 800-56A Revision 3; RFC 3526	
Key Establishment (FCS_CKM.2)		
Elliptic curve-based scheme (ECDSA) P-256, P-384, P-521	NIST Special Publication 800-56A Revision 3	A7082 (SSL)
FFC Schemes using ‘safe-prime’ groups (2048-bit MODP)	NIST Special Publication 800-56A Revision 3, RFC 3526	A7080 (SSL-VM)
Encryption/Decryption (FCS_COP.1/DataEncryption)		
AES in CBC mode (128, 256 bits)	AES as specified in ISO 18033-3 CBC as specified in ISO 10116	A7082 (SSL)
AES in GCM mode (128, 256 bits)	AES as specified in ISO 18033-3	A7080 (SSL-VM)

Functions	Standards	Certificates
	GCM as specified in ISO 19772	A7083 (FIPS) A7081 (FIPS-VM)
Cryptographic signature services (Signature Generation and Verification) (FCS_COP.1/SigGen)		
RSA Digital Signature Algorithm (rDSA) (2048-bit modulus)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3	A7082 (SSL) A7080 (SSL-VM)
ECDSA (P-256, P-384, P-521)	ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves"	
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1 (digest size 160 bits) SHA-256 (digest size 256 bits) SHA-384 (digest size 384 bits) SHA-512 (digest size 512 bits)	ISO/IEC 10118-3:2004	A7082 (SSL) A7080 (SSL-VM) A7083 (FIPS) A7081 (FIPS-VM)
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-1 (key/digest sizes 160 bits) HMAC-SHA2-256 (key/digest sizes 256 bits) HMAC-SHA2-384 (key/digest sizes 384 bits) HMAC-SHA2-512 (key/digest sizes 512 bits)	ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2	A7082 (SSL) A7080 (SSL-VM) A7083 (FIPS) A7081 (FIPS-VM)
Random bit generation (FCS_RBG_EXT.1)		
CTR-DRBG (AES) – 256 bits entropy	ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions	A7083 (FIPS) A7081 (FIPS-VM)

The TOE generates asymmetric keys for TLS, SSH and X.509 certificates as follows:

- TLS Server: 2048 bits RSA, P-256, P-384, P-521 curves ECC
- TLS Client: 2048 bit RSA, P-256, P-384, P-521 curves ECC,

- X.509 Certificate Requests: 2048 bit RSA, and P-256, P-384, P-521 curves ECC key generation
- SSH: 2048 bit RSA, and MODP 2048 bit safe primes per RFC 3526

The TOE performs keys establishment for TLS and SSH as follows:

- TLS Server: ECDSA (P-256, P-384, P-521) and MODP 2048 bit safe primes per RFC 3526
- TLS Client: ECDSA (P-256, P-384, P-521)
- SSH: FFC MODP 2048 bit safe primes per RFC 3526

The TOE uses JitterEntropy 3.4.1 as a raw entropy source collected from CPU execution time jitter. The collected entropy is injected into the Linux kernel /dev/random device using the RNDADDENTROPY ioctl. This noise source provides full entropy to seed the DRBG with 256 bits. The Fortinet <library> contains a CTR_DRBG implemented in accordance with ISO/IEC 18031:2011 Deterministic Random Bit Generator (DRBG) based on the AES 256 block cipher in counter mode (CTR_DRBG (AES-256)). Entropy from the noise source is used to seed the DRBG with 256 bits of full entropy. A failure of the entropy source is a blocking event for the cryptographic system and the entropy source is continually monitored for health; this helps ensure that a catastrophic failure of the noise source will halt the operation of the TOE. The TOE uses its DRBG to generate all keys.

The TOE maintains a number of keys and CSPs related to its secure operation (see Table 6). Administrative passwords are stored in the configuration file on the flash drive of the TOE and are hashed using SHA-256 to ensure their confidentiality. Keys can be zeroized either by erasing the boot device or through a factory reset of the TOE.

The following table describes the origin, storage and zeroization of keys as relevant to FCS_CKM.4 and FPT_SKP_EXT.1 provided by the TOE. “Flash” here means the TOE non-volatile storage. RAM here means the TOE volatile storage. Zeroization is performed using the relevant keystore or filesystem APIs, depending on the storage location.

Table 7: Key/CSP Storage and zeroization

Key or CSP	Storage	Zeroization Method	Origin
Firmware Update Key	Flash	Erase boot device (overwritten with zeroes)	Preconfigured
HTTPS/TLS Pre-Master Secret	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic
SSH Server/Host Key	Flash	Erase boot device (overwritten with zeroes) overwrite with new key (if the hostkey was “regenerated” by the administrator)	Preconfigured and can be regenerated/modified by the administrator using exec ssh-regen-keys
Firmware Integrity Key	Flash	Erase boot device (overwritten with zeroes)	Preconfigured
HTTPS/TLS Pre-	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic

Master/Master Secrets			
HTTPS/TLS Server/Host Key	Flash RAM	Erase boot device (overwritten with zeroes) Power cycle or reboot; session terminated (overwritten with zeroes)	Preconfigured
HTTPS/TLS Session Authentication Key	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic
HTTPS/TLS Session Encryption Key	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic
SSH Session Authentication Key	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic
SSH Session Encryption Key	RAM	Power cycle or reboot; session terminated (overwritten with zeroes)	Automatic
User/Admin Password hashes	Flash	Factory reset (overwritten with zeroes)	Manual

The TOE stores a number of CSPs in volatile memory during normal operation of the cryptographic modules. These CSPs include plaintext ephemeral keys and copies of the persistent keys described above are loaded into memory during normal operation. The TOE maintains these keys in its volatile memory to support the TLS and HTTPS connections to the TOE. These CSPs are cleared when the appliance power cycles or reboots. Ephemeral keys are overwritten with a fixed pattern (zeroes) when they are no longer required. Each of the CSPs are protected from unauthorized access via memory management which disallows any memory reads from other processes within the OS ensuring that the CSPs are only available to the calling application.

Plaintext private keys for the purposes of SSH, HTTPS and TLS are maintained on the flash filesystem and are not viewable through the TOE interfaces. When these keys are no longer required the administrator can remove the keys by erasing the boot device, or at any other time by manually deleting these keys from the keystore using the TOE GUI.

All keys and CSPs can be zeroized by executing the following commands from the CLI: i) factory-reset ii) erase-all-disks . The erase-all-disks command performs a direct overwrite (no intermediate file system APIs are used).

The TOE does not provide any interfaces to view the keys/CSPs.

7.2.2 Cryptographic Operations

The TOE also implements the cryptographic algorithms listed below in support of TLS and SSH as well as for the additional functionality specified.

- AES-CBC, AES-GCM (128-bit, 256-bit)

- RSA signature generation and verification (2048-bit)
- 2048-bit RSA with SHA-256 is also used for TOE update, digital signature verification for firmware integrity, and self-tests
- ECDSA signature generation and verification (P-256, P-384, P-521)
- SHA-1, SHA-256, SHA-384, SHA-512

SHA-256 is also used for administrator password hashing for non-volatile storage, digital signature verification for self-test of configuration integrity and for integrity of firmware update

SHA-512 is only used for SSH, as part of a matching HMAC algorithm, and for storing password hashes in the underlying file system.

The TOE uses keyed hashes with the following parameters:

- HMAC-SHA-1 (160 bit key size with 512 bit block size, 160 bit output length),
- HMAC-SHA-256 (256 bit keysize with 512 bit block size, 256 bit output length),
- HMAC-SHA-384 (256 bit key with a 1024 bit block size, 384 bit output length)
- HMAC-SHA-512 (256 bit key with 1024 bit block, 512 bit output length)

Hash Algorithm	TSF Usage
SHA-1	In TLS ciphersuites
	In matching HMAC
SHA2-512	SSH session message integrity verification
	Creating hashes of user passwords for storage in the underlaying file system.
	In matching HMAC
	In TLS Signature algorithms
SHA2-384	In TLS ciphersuites
	In TLS Signature algorithms
	In matching HMAC
SHA2-256	SSH session message integrity verification
	SSH session key establishment
	SSH peer authentication when the TOE is a server
	In TLS Ciphersuites
	In TLS signature algorithms
	Self-test of configuration integrity
	Firmware update integrity verification
	In matching HMAC

7.2.3 SSH Protocol

The TOE acts only as an SSH server. The TOE's SSH server implements the SSH protocol in accordance with RFCs 4251, 4252, 4253, 4254, 6668, 8268, 8308, and 8332. No optional

characteristics are supported. The TOE supports SSH keyboard-interactive authentication method as described in RFC 4256. Correct authentication requires a valid username and a valid password used as the authentication factor in a separate message from the username and other control data messages.

At all times, packets greater than 256 kilobytes in an SSH transport connection are dropped by terminating the SSH session and not processing the oversized packet.

For its SSH transport implementation, the TOE uses aes128-CBC and aes256-CBC data encryption modes, rejecting all other encryption algorithms. This is hardcoded and not configurable.

The SSH public-key based authentication implementation uses only rsa-sha2-512 as its public key algorithm and rejects all other public key algorithms. The TOE establishes a user identity when an SSH user performs successful password authentication.

The SSH transport implementation uses hmac-sha2-256 and hmac-sha2-512 as its data integrity MAC algorithms and rejects all other MAC algorithms.

The TOE's SSH implementation uses only diffie-hellman-group14-sha256_for its key exchange methods, which implements a SHA2-256 based KDF, as specified in RC 4252 section 7.2.

Within SSH connections the same session keys are used for a threshold of no longer than one hour, and each encryption key is used to protect no more than 1GB of data. Before either of these thresholds are reached, the TOE performs a rekey (approximately 525 mB transmitted, or 525 mB received).

7.2.4 Random Bit Generation

Tthe TOE uses an AES CounterDRBG to provide cryptographically secure random numbers. The entropy source is the JitterRNG daemon, and the DRBG is seeded with a minimum of 256 bits of entropy at all times.

FCS_RBG_EXT.1

7.2.5 TLS & HTTPS Protocols

The TOE implements TLS and HTTPS protocols for external communications. The TOE supports HTTPS to secure the sessions for remote administration over TLSv1.2 and 1.3. The TOE does not use HTTPS in a client capacity. The TOE's HTTPS protocol complies with RFC 2818 (by implementing all "SHALL" and "MUST" statements) and uses the TLS functionality described below. TLS 1.2 or 1.3 is also used for the purposes of protecting the audit logs while in transit to the audit servers. In all cases, an invalid certificate will cause the connection to be aborted.

The TOE implements TLS 1.2 and TLS 1.3 as both a client and a server, rejecting all other TLS/SSL versions. Specifically, enabling CC mode (required in evaluated configuration) forces the TOE to use only TLS versions 1.2 or 1.3. All other versions are automatically disabled by default.

The TOE is a TLS client when communicating with external audit servers. Mutual authentication is not supported. The TOE supports the following cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA

- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_CCM_SHA256
- TLS_AES_128_CCM_8_SHA256

The TOE is a TLS server when communicating with remote administrative users accessing the TOEs GUI. Mutual authentication is not supported. The TOE supports the following cipher suites:

- TLS_DHE_RSA_WITH_AES_128_CBC_SHA
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA
- TLS_DHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_DHE_RSA_WITH_AES_256_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_AES_128_GCM_SHA256
- TLS_AES_256_GCM_SHA384
- TLS_AES_128_CCM_SHA256
- TLS_AES_128_CCM_8_SHA256

The TOE operates in CC mode of operation that restricts the cipher suites and algorithms used by HTTPS/TLS to those identified above. The TOE supports server authentication via x.509v3 certificates using RSA keys of 2048 bits, or ECDSA key sizes of 256 bits, 384 bits, or 512 bits. The administrator does not need to take any specific actions to ensure compliance once the CC mode of operation has been enabled. All required behaviours are performed automatically when CC mode has been enabled, with no further action required by the administrator. In the evaluated configuration, the TOE supports the following parameters for TLS. The TLS client and server uses secp256r1, secp384r1, or secp521r1 when a TLS_ECDHE cipher suite is negotiated. When the TOE is a TLS server, it also permits MODP sizes of 2048-bits when an DHE cipher suite is negotiated.

The TOE terminates the handshake immediately if an unsupported DHE parameter is returned in the Server Key Exchange. Otherwise, the TOE accepts all valid server-generated DHE parameters. This is the default behavior in CC mode and is not configurable.

The TOE supports DNS names and wildcards for DNS names in the SAN and CN. IPv4/IPv6 addresses are supported in the SAN and are configured by the administrative user. The TOE validates any presented server certificate in the manner specified by RFC 6125 section 6. Specifically, if the SAN is present, the client uses the IP address in the SAN as the reference

identifier for the TLS server certificate. When the SAN is not available, the TOE makes use of the FQDN (CN). The CN only supports DNS names. Invalid certificates are rejected. No administrator override exists to re-adjudicate the rejection of an invalid certificate.

The TLS client presents the signature_algorithms extension with support of the listed algorithms in [ST] section 6.2.12. This is the default behaviour in CC mode and is not configurable.

The TLS client presents the supported_groups extension with the following curves/groups: secp256r1, secp384r1, and secp521r1. This is the default behaviour in CC mode and is not configurable.

The TOE does not permit any out-of-band provisioning of pre-shared keys.

The TLS server supports session resumption based on session tickets, in a single context only. Session tickets adhere to the structural format provided in section 4 of RFC 5077 for TLSv1.2 and RFC 8446 for TLSv1.3. Session tickets are encrypted according to the TLS negotiated symmetric encryption algorithm consistent with FCS_COP.1/DataEncryption. Depending on the negotiated ciphersuite, AES in CBC or GCM mode (128, 256 bits) are used to protect session tickets. Because the session resumption mechanism relies on the ability to decrypt and validate the session ticket, the TOE will force a full re-negotiation of the session if the ticket is invalid, expired, or cannot be decrypted.

7.3 Identification and Authentication

The TOE requires all users to be successfully identified and authenticated before allowing any TSF mediated actions to be performed aside from display of the warning banner to the administrator or generate session keys for incoming SSH and TLS connections. Administrative access to the TOE is facilitated by either directly connecting to the appliance console's CLI, remotely to the CLI via SSH, or by remotely connecting to appliance GUI via HTTPS/TLS.

The TOE provides for the ability to respond to ICMP echo requests with ICMP echo replies prior to any user identification or authentication.

Regardless of the interface at which the administrator interacts, the TOE prompts the user for a credential. The administrator credentials are the same across the interfaces (i.e. the interfaces do not have separate credentials). Only after the administrative user presents the correct authentication credentials will they be granted access to the TOE administrative functionality. Only passwords can be used for the SSH connection. If the asserted identity and password cannot be verified, then the login fails and an audit record is generated.

7.3.1 Authentication Failure Management

The TOE provides administrators with the capability to specify a maximum number of authentication attempts between 3 and 20 (default 4) that can be attempted before a user account is locked out from the remote GUI or SSH. The TSF increments the failure counter in non-volatile memory so that the accumulated number of failures is persisted across reboots. The failure counter is per administrator account. In the event the remote administrator is locked out, access is restored by waiting for the lockout period to expire. The lockout period is a configurable value between 1 and 60 minutes, with a default value of 3 minutes. Once the maximum number of attempts has been reached, the account will become locked and inaccessible until an administrator configured period of time has been met. The TOE supports a local interface that is not subject to the authentication failure lockout function and does not lock an administrative user out. Even if an administrator is locked out from a remote interface, they can still login locally. For a virtual TOE, the administrator signs in to ESXi console to launch a local console for each VM. This prevents a situation where no administrator access is available.

7.3.2 Password Management

The TOE provides a local password based authentication mechanism. The TOE supports the local definition of users with corresponding passwords. The passwords can be composed of any combination of upper and lower case letters, numbers, and special characters (that include: "!", "@", "#", "$", "%", "^", "&", "*", "(", ")", ">", "[space]", "“”, "”", "+", "-", "/", ":", ";", "<", "=", "?", "[", "\", "]", "[underscore]", "`", "{", "|", "}", and "~"). The minimum password length is settable by the Authorized Administrator and can range from 6 to 64 characters.

7.3.3 User Identification and Authentication

The process for authentication is the same for administrative access whether administration is occurring via direct connection or remotely. At initial login, the administrative user is prompted to provide a username. After the user provides the username, the user is prompted to provide the administrative password associated with the user account. The TOE then either grants administrative access (if the combination of username and password is correct) or indicates that the login was unsuccessful. The TOE obscures all characters entered when attempting password authentication and does not provide a reason for failure in the cases of a login failure.

7.3.4 X.509 Certificate Operations

The TOE performs X.509 certificate validation in support of TLS and TLS/HTTPS communications. Certificates which are loaded into the trust store for use in an authentication step are evaluated. Since the TOE does not support TLS with mutual authentication, any TLS client certificate presented to the TOE is ignored/not_validated. Specifically, the following validation rules are followed:

The TSF performs RFC 5280 certificate validation and certification path validation supporting a minimum path length of three certificates.

The TSF validates that the certification path terminates with a trusted CA certificate designated as a trust anchor.

The certification path is validated by ensuring that all CA certificates in the certification path contain the basicConstraints extension with the CA flag set to TRUE.

The extendedKeyUsage field is validated according to the following rules:

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.

Note that FIA_X509_EXT.1/Rev also optionally requires the TSF to verify the extendedKeyUsage field for certificates used for trusted updates and executable code integrity verification. The TOE does not use X.509 certificates for these purposes so this portion of the requirement is trivially satisfied by the TSF.

The TOE chooses the certificate to present to external TLS clients based on the server certificate that is loaded into it as part of administrative configuration. In all cases the TOE validates the certificate that is presented to it and validates the chain based on what is included in the certificate.

The TOE supports the use of Certificate Revocation Lists (CRLs) as specified in RFC 5280 for revocation status checking of TLS certificates. In all cases, if the revocation status of a certificate cannot be determined, the TSF treats it as invalid. Both the leaf certificate and any intermediate CA certificates are checked for TLS.

The TOE also includes the ability to generate a certificate signing request as specified by RFC 2986 to be sent to a CA for issuance of TLS server/client certificates. The certificate signing request generates an RSA or elliptic curve key pair and can include the Common Name, Organization, Organizational Unit, and Country fields. When the signed response is loaded into the TOE, the TSF validates the certificate chain from the root CA and will accept the response as the TLS server certificate if the certificate chain is valid. This is subsequently the certificate that the TOE issues to external TLS clients attempting to connect to it.

7.4 Security Management

The TOE provides management functionality over both local and remote interfaces. The local interface is a dedicated physical port, direct console interface to the TOE's CLI. Local access to a virtual TOE is through the ESXi console via URL. The remote interfaces are an SSH connection to the CLI and a web GUI accessed over TLS/HTTPS. The TOE supports one default fully privileged user account, Admin, which corresponds to the PP defined Security Administrator. The Admin account has full privileges to all administrative functions, and is defined as the Global Security Administrator. Other user accounts may be configured, which are defined as Subordinate Security Administrator accounts.

Subordinate Security Administrator accounts may be configured with a variety of permissions and administrative powers, such that the global Admin account may create user accounts to delegate some or all of its powers. In this way, the TOE maintains two roles: the mandatory "Security Administrator" role, and the lesser "TOE User" role. TOE Users may be configured by any security administrator to have some, all, or none of the permissions indicated below.

Table 7 below identifies the management functions that are available on each interface. The local management functions available are those available on the console and the functions available to remote admin interfaces are available via SSH and/or HTTPS. All remote functions can be performed via the CLI and the table notes the specific functions that cannot be performed via the GUI via footnote. All functions are restricted to the Security Administrator.

Table 8: Management Functions by Interface

Function	Local	Remote
Ability to administer the TOE locally	X	
Ability to administer the TOE remotely		X
Ability to configure the access banner	X	X
Ability to configure the session inactivity time before session termination	X	X
Ability to update the TOE and verify the updates using digital signature capability prior to installing those updates	X	X
Ability to start and stop services		X
Ability to configure authentication failure parameters for FIA_AFL.1	X	X[8]
Ability to manage the cryptographic keys	X	X
Ability to set the time which is used for time-stamps	X	X
Ability to manage the TOE's trust store and designate X.509v3 certificates as trust anchors	X	X

[8] This function is only available locally or remotely using SSH to the CLI.

Ability to generate Certificate signing requests (CSRs) and process CA certificate responses	X	X

The Security Administrator has the ability to manage cryptographic keys to manipulate (add, remove) the certificates (and keys) that reside in the TOE's trust store and to generate certificate signing requests, which include key pairs.

The Security Administrator is able to start and stop the following services:

1. HTTPS (Trusted Path)
2. SSH (Trusted Path)
3. TLS Client (Trusted channel to remote audit server)

7.5 Protection of the TSF

7.5.1 Protection of Administrator Passwords & Keys

The TSF stores all Administrative password data in an obfuscated format. Specifically, the data is stored as a SHA-256 hash.

The TSF stores its private key on its local file system when the key pair for the CSR is first generated (the keypair, generated by the TSF, is created in RAM, then written in plaintext to the non-volatile storage via filesystem APIs). When the signed certificate response is received by the TOE, the certificate and private key are stored in plaintext in a secure directory that is not accessible to administrators. This directory is protected by implementation of a limited access interface, and filesystem controls which prevent direct access by any user except through the GUI. The GUI itself provides no mechanism to interact with these keys other than creation and deletion.

The TOE stores its SSH private host key in plaintext in flash and does not provide any interfaces to view the key.

For both credential and non-public key data, no administrative interface exists to read this data.

7.5.2 TSF Testing

The TSF provides a set of self-tests run during initial start-up. During a normal boot-up sequence the TOE administrator can see on the local console the following types of tests:

- Firmware integrity test: The Firmware Integrity Test is run automatically whenever the system images is loaded and confirms through use of RSA 2048-bit and SHA-256 digital signature verification that the image file that's about to be loaded was properly signed and has maintained its integrity since being signed.
- Cryptographic (AES, DHE, SHA, ECDHE, ECDSA, and RSA) known answer tests: For each algorithm, the implementation is fed known plaintext data and a known key (when appropriate). These values are used to generate a value. This value is compared to a known value to verify that the implementation is operating correctly.
- SP 800-90A health tests: For these tests, each of the health tests in defined SP 800-90A are executed.
- RNG known answer test: For this test, known seed values are provided to the DRBG implementation. The DRBG uses these values to generate random bits. These random bits are compared to known random bits to ensure that the DRBG is operating correctly.

The results of the startup self-tests are displayed on the console during the startup process. Indication of successful tests would appear as follows: `Running <test>... passed`; while completion of all self-tests is indicated by: `Self-tests passed.`

The TSF executes the following conditional tests when the related service is invoked:

- Cryptographic (AES, DHE, SHA, KAS, ECDSA sign/verify, and RSA sign/verify) known answer tests: For each algorithm, the implementation is fed known plaintext data and a known key (when appropriate). These values are used to generate a value. This value is compared to a known value to verify that the implementation is operating correctly.
- SP 800-90A health tests: For these tests, each of the health tests in defined SP 800-90A are executed.
- RNG known answer test: For this test, known seed values are provided to the DRBG implementation. The DRBG uses these values to generate random bits. These random bits are compared to known random bits to ensure that the DRBG is operating correctly.

In the event that any of the self-tests or conditional tests fail, the module logs a "error indicator" for the specific test(s) and enters an error state as shown by the following console output:

```
Self-tests failed
Entering error mode...
The system is going down NOW !!
The system is halted.
```

The TOE's Error Mode allows the TOE to enter into a secure state where all data output and cryptographic services are inhibited. The unit shuts down all interfaces including the console and blocks traffic. To resume normal CC mode operation, the administrator must power cycle the TOE. If the self-tests pass after the reboot, the TOE will resume normal CC operation. If a self-test continues to fail after rebooting, there is likely a serious firmware or hardware problem and the TOE should be removed from the network until the problem is solved (the administrator should contact Fortinet Support)

These self-tests are sufficient to ensure that the image and configuration file are operating in a known good state, TSF executable code has appropriately not been modified, and that the cryptographic functionality has not been tampered with or degraded, either through a compromise of the cryptographic functionality itself or through a degradation of any hardware components on which this functionality relies. There is no situation in which an administrator may unwittingly be using a modified TOE or may be using the TOE to transmit sensitive data using a degraded cryptographic implementation.

7.5.3 Trusted Update

The TOE supports firmware updates. Only one version of the TOE firmware is loaded at any one time. The TOE has a manual update mechanism. The TOE protects itself during updates through the use of a cryptographic signature. The update process is performed as follows. The administrator downloads the TOE to their workstation from https://support.fortinet.com. The administrator will then copy the file to the TOE using: HTTPS web interface, SSH, or direct connection to the console port using a RJ-45 to DB-9 serial cable or a nullmodem cable. Once the update is uploaded to the TOE, a 2048 bit RSA signature is verified for any TOE firmware build (builds include patches) to verify the update is valid. The signature is compared to a known key value stored on the TOE and pre-configured into the firmware image. After image file is uploaded and its integrity is checked using digital signature, a reboot occurs and the image will be executed.

The current running version of the firmware can be queried from the CLI using the command: ***status***. The version line of the status display shows the FortiSandbox model number, firmware

version, build number and date. The firmware version can also be queried from the Web UI's menus: ***Dashboard: Status: System Information box***

Before proceeding with a TOE update via the GUI or CLI, the following automated process is followed when in the evaluated mode of operation:

- If a signature is not present, abort the upgrade
- Extract the public key and signature from the firmware
- Validate that the public key is the same as is stored on the TOE. If the public keys do not match abort the upgrade.
- Validate the image signature using the public key from the update. If the image validation using the public key fails, abort the upgrade.

If the firmware load test fails, the error message displayed is "File is not an update file." Otherwise the TOE displays "upgrade successful" and reboots. An administrator may query the current version of the TOE through the CLI or web interface.

7.5.4 Reliable Time Stamps

The TOE is a hardware appliance or a virtual appliance image installed on a hardware appliance that includes a hardware-based real-time clock to ensure that reliable time information is available. The TOE's real-time clock stores the system time and date information. The TOE's embedded OS manages the clock and exposes administrator clock-related functions. The clock is used for audit record time stamps, measuring session activity for termination, timing of lockout in FIA_AFL.1, and for cryptographic operations based on time/date (e.g. validation of X.509 certificates). The Administrator can configure the system time and date of the FortiSandbox unit from the GUI, from the ***Dashboard***: ***Status: System Time*** menus or via the CLI using the ***set time*** command.

7.6 TOE Access

The TOE enforces access restrictions on its local and administrative interfaces. A Security Administrator can configure maximum inactivity times for administrative sessions of 1-480 minutes through the TOE GUI and CLI interfaces. A configured inactivity period will be applied to both local and remote sessions in the same manner. When the interface has been idle for more than the configured period of time, the session will be terminated and will require authentication to establish a new session.

Each interface also includes mechanisms for the Security Administrator on each to voluntarily terminate their own session. On the GUI, this is accomplished using a logout button at the top-righthand side where there is UI to sign admin out, or using the 'exit' command on CLI.

Both local and remote interfaces display a configurable pre-authentication warning banner that can be used to advise Security Administrators of appropriate usage of the system. The custom banner messages are configured using the from the GUI: System > Customization > Custom Message. Configured warning messages are displayed prior to logging in, and for all interfaces: GUI, SSH, and Console.

7.7 Trusted Path/Channels

The TOE supports communications with one or more FortiAnalyzer audit log servers, utilizing a proprietary protocol. This protocol is protected via TLS sessions, where the TOE is a TLS client. This protects the data from modification and disclosure. TLS provides assured identification of the non-TSF endpoint by validating X.509 certificates. The TOE retains a trusted store of certificate authorities which it uses to verify digital signatures on those non-TSF certificates. The TOE is responsible for initiating the trusted channel with the external trusted IT entities.

All remote administrative communications take place over a secure encrypted session. Remote CLI sessions occur over an SSH connection. Remote GUI connections take place over a TLS/HTTPS connection. Sessions are encrypted using AES encryption and uses HMACs to protect integrity. The remote administrators can initiate SSH and TLS communications with the TOE. The TOE is the SSH/TLS server and mutual authentication is not supported.

8. Terms and Definitions

Table 9: TOE Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
AES	Advanced Encryption Standard
ASCII	American Standard Code for Information Interchange
BIOS	Basic Input/Output System
CA	Certificate Authority
CBC	Cipher Block Chaining
CLI	Command Line Interface
CMOS	Complementary Metal–Oxide–Semiconductor
CRL	Certificate Revocation List
CSP	Critical Security Parameter
CSR	Certificate Signing Request
CTR	Counter
DH	Diffie-Hellman
DHE	Diffie-Hellman Ephemeral
DNS	Domain Name System
DRBG	Deterministic Random Bit Generator
DSA	Digital Signature Algorithm
DTLS	Datagram Transport Layer Security
ECDH	Elliptic Curve Diffie-Hellman
ECDHE	Elliptic Curve Diffie-Hellman Ephemeral
ECDSA	Elliptic Curve Digital Signature Algorithm
ESP	Encapsulating Security Payload
GCM	Galois Counter Mode
GUI	Graphical User Interface
HMAC	Hash Message Authentication Code
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPs	Hypertext Transfer Protocol Secure
ICMP	Internet Control Message Protocol
IKE	Internet Key Exchange
IP	Internet Protocol
KAT	Known Answer Test
KDF	Key Derivation Function
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
PCT	Pairwise Consistency Test
PKCS	Public Key Cryptography Standards
RFC	Requests for Comments
RSA	Rivest-Shamir-Adleman
SA	Security Association
SAN	Subject Alternative Name
SFP	Small Form-Factor Pluggable
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol
SPD	Security Policy Database
SSH	Secure Shell
TCP	Transmission Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
UI	User Interface

Table 9: TOE Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
URI	Uniform Resource Identifier
VPN	Virtual Private Network

Table 10: CC Abbreviations and Acronyms	
Abbreviations/ Acronyms	Description
CC	Common Criteria
CCRA	Arrangement on the Recognition of Common Criteria Certificates in the field of IT Security
DOD	Department of Defense
EAL	Evaluation Assurance Level
FIPS	Federal Information Processing Standards
OSP	Organizational Security Policy
PP	Protection Profile
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SFP	Security Function Policy
ST	Security Target
TOE	Target of Evaluation
TSF	TSF = TOE for pND or Case 1 vND according to section 1.2 TSF = TOE + VS for Case 2 vND (vND evaluated as a pND) according to section 1.2
TSFI	TSF Interface
TSS	TOE Summary Specification

9. References

Table 11: TOE & TOE Guidance Documentation			
Reference	Description	Version	Date
[T1]	FortiSandbox 4.4 NDcPP Common Criteria Technote	34-446-1080542-20250905	September 5, 2025
[T2]	Fortisandbox 4.4.6 Administration Guide	7.2.11	June 25, 2025
[T3]	FortiSandbox 4.4.6 Log Message Reference	7.2.11	February 11, 2025
[T4]	FortiSandbox 4.4.6 Getting Started	34-444-1011093-20240524	May 23, 2024
[T5]	FortiSandbox 4.4.0 – 4.4.7 Install Guide for VMware	34-445-888478-20250221	February 21, 2015
[T6]	NDcPP Logging Addendum for FortiSandbox 4.4.6	4.0	July 8, 2025

Table 12: Common Criteria v3.1 References			
Reference	Description	Version	Date
[C1]	Common Criteria for Information Technology Security Evaluation Part 1: Introduction and general model CCMB-2017-04-001	V3.1 R5	April 2017
[C2]	Common Criteria for Information Technology Security Evaluation Part 2: Security functional components CCMB-2017-04-002	V3.1 R5	April 2017
[C3]	Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components CCMB-2017-04-003	V3.1 R5	April 2017
[C4]	Common Criteria for Information Technology Security Evaluation Evaluation Methodology CCMB-2017-04-004	V3.1 R5	April 2017
[C5]	CC and CEM addenda - Exact Conformance, Selection-Based SFRs, Optional SFRs CCDB-013-v2.0	V0.5	Sep 2021

Table 13: Supporting Documentation			
Reference	Description	Version	Date
[cPP]	Collaborative Protection Profile for Network Devices	3.0e	December 6, 2023
[SD]	Supporting Document Mandatory Technical Document Evaluation Activities for Network Device cPP	3.0e	December 6, 2023
[SSH]	Functional Package for Secure Shell (SSH)	1.0	2021-05-13