Security Target

Juniper SRX4700, SRX4120 and SRX2300 for Junos OS 25.2R1

ST Version: 1.0

Date: July 18, 2026

Prepared for:

Juniper Networks

https://www.juniper.net

Prepared by:

Teron Labs

www.teronlabs.com

Revision History

Version	Date	Author(s)	Description of Change
1.0	July 18, 2026	Teron Labs	Final certification version

Contents

List of Tables

1 Security Target Introduction

This section is the Security Target introduction. It describes the Target of Evaluation (TOE) in a narrative way at three levels of abstraction: TOE Reference, TOE Overview and TOE Description. The objective is to assist the reader in understanding the TOE and in determining that the TOE is suitable for the intended use.

The target audience is the users and the potential users of the TOE wishing to gain a precise understanding of the TOE and the security features provided. The readers are assumed to possess a good understanding of the computer networking terms and practices. The readers are also expected to have a good understanding of network and computer security. Finally, the readers are assumed to be proficient in Common Criteria and the terminology thereof. Some familiarity with the networking products of Juniper Networks, Inc. is beneficial.

The Security Target (ST) Introduction commences with the statements of the Security Target Reference and the TOE Reference in Sections 1.1 and 1.2, followed by the TOE Overview in Sect. 1.3. TOE Description is given in Sect. 1.4.

The TOE and the ST claim conformance to Common Criteria and Protection Profiles identified in Table 1.

Table 1 TOE Conformance

CC Version	Common Criteria Version 3.1 Revision 5
Base-PP	collaborative Protection Profile for Network Devices, Version: 3.0e, Date: 06-December-2023 (CPP_ND_V3.0E)
PP-Modules and Packages	– Functional Package for Secure Shell (SSH) Version 1.0, 2021-05-13 (PKG_SSH_V1.0) – PP-Module for Intrusion Prevention Systems (IPS) Version: 1.0, 2021-05-11 (MOD_IPS_v1.0) – PP-Module for Stateful Traffic Filter Firewalls Version 1.4+Errata 20200625, 25-June-2020 (MOD_CPP_FW_V1.4E) – PP-Module for VPN Gateways Version: 1.3, 2023-08-16 (MOD_VPNGW_v1.3)
PP-Configuration	PP-Configuration for Network Devices, Intrusion Prevention Systems, Stateful Traffic Filter Firewalls, and Virtual Private Network Gateways, Version 2.0, 2024-04-25 (CFG_NDcPP-IPS-FW-VPNGW_V2.0)

1.1 Security Target Reference

Security Target Title	Security Target Juniper SRX4700, SRX4120 and SRX2300 for Junos OS 25.2R1
Security Target Version	1.0
Security Target Date	July 18, 2026

1.2 TOE Reference

TOE Identification	Juniper SRX4700, SRX4120 and SRX2300 for Junos OS 25.2R1
TOE Developer	Juniper Networks, Inc.
Evaluation Sponsor	Juniper Networks, Inc.

1.3 TOE Overview

1.3.1 Intended Method of Use

The TOE is a non-virtual and non-distributed network device with complex security functionality. It is a suite of appliances meeting the security requirements stated in Base-PP, the Functional Package, and the PP-Modules. The TOE is a suite of Juniper Networks SRX-series Universal Routing Platforms illustrated in Figure 1. The TOE is an instance of the Juniper Networks portfolio of the software-defined networking (SDN)-enabled routing platforms.

SRX4700

SRX4120

SRX2300

Figure 1 TOE Variants

The TOE is purpose-built to protect network environments and provide Internet Mix (IMIX) firewall with high throughput. The TOE incorporates multiple security services and networking functions for a non-virtual appliance. This allows the TOE to provide highly customizable threat protection, automation, and integration capabilities. Best-in-class advanced security capabilities on the TOE are offered as high throughput Next-Generation Firewall (NGFW), an Intrusion Prevention System (IPS), and a high-speed IPsec Virtual Private Network (VPN) in the data center, enterprise campus, and regional headquarters deployments with IMIX traffic patterns.

The TOE can operate in a single mode or in a Multinode High Availability (HA) Mode. In Multinode HA Mode, a pair of devices, called nodes, executing the same software, is connected and configured to operate like a single device to provide high availability. The two nodes back up each other. The Active node is the primary device, and the other is the backup device. If the primary device fails, the secondary device takes over processing of traffic. The interconnection of the two nodes is protected with IPsec. The configuration of the nodes is with Netconf over SSH.

The TOE supports definition and enforcement of information flow policies among network nodes. Stateful inspection is applied on each packet that traverses the network. The TOE provides a central point of control to manage the network security policy. The network topology enforces that each information flow from one network node and subnetwork to another passes through an instance of the TOE. The TOE then controls the information flows between the nodes and subnetworks based on network node addresses, protocol, type of access requested, and services requested. The TOE ensures that each security-relevant activity is audited and that the TOE functions are protected from potential attacks. The TOE also provides tools to manage all security functions.

The TOE is composed of the chassis and the Junos OS Operating System. In concert, they implement the routing and management plane functions for a complete network appliance. The chassis variants are illustrated in Figure 1.

The TOE implements all security functions required for controlling access to the management functions. The management functions of the TOE are only accessible to legitimate administrators through a Command Line Interface (CLI). No other means but the CLI are available for administering the TOE. The TOE may be managed locally or remotely. Remote management sessions are protected with Secure Shell (SSH).

1.3.2 Major Security Features of the TOE

The TOE implements a set of security functions and security mechanisms required for conformance with the Base-PP enhanced with SSH, Intrusion Prevention System IPS, stateful traffic filtering firewall, and a VPN Gateway. The major security features implemented by the TOE are described in Table 2.

Table 2 Major Security Features of the TOE

Security Feature	Description
Security Audit	The TOE implements an audit function to collect detailed information about the state of the TOE to allow the administrator to troubleshoot the TOE and investigate possible security-related incidents.
Cryptography	The TOE implements a suite of cryptographic algorithms and protocols. All cryptographic functions and protocols are implemented using cryptographic libraries linked to the TOE software. Cryptography does not depend on the platform on which the TOE is executed. Each cryptographic algorithm implemented by the TOE is validated against the Cryptographic Algorithm Validation Program (CAVP). The cryptographic algorithms and protocols are used to implement the critical security functions of the TOE but are also used for implementing the essential network security features and self-testing of the TOE.
Trusted Paths and Channels	The TOE implements trusted paths and trusted channels for secure connectivity between the TOE and a syslog server, between the TOE and a remote management station, and between two instances of the TOE when configured in a Multinode HA Mode. The trusted paths and trusted channels are implemented with SSH and IPsec.
SSH Client and Server	The TOE implements a SSH server and a SSH client to allow secure connection with a syslog server and with a remote management station.
Identification, Authentication, Authorization and Access Control	The TOE ensures that access to the administrative functions is only granted to successfully identified and authenticated users. Illegitimate users are deterred and prevented from gaining access.

Security Management	The TOE implements a Command Line Interface made available to the administrators. The CLI may be accessed locally from console or remotely over a SSH connection.
Protection	The TOE protects itself from tampering by passive and active means to ensure that the TOE always boots into a secure state and remains so when operated.
Intrusion Prevention System (IPS)	The TOE allows the administrator to define profiles and inspects the network traffic against those profiles. Any suspicious network traffic may be dropped.
Packet Filtering Firewall	The TOE implements a network layer firewall and an application gateway for stateless and stateful packet filtering. The administrator may define information flow policies which the TOE enforces to only allow approved information flows from one connected network to another may occur.
VPN Gateway	The TOE implements a VPN gateway with IPsec in tunnel mode for secure connection with an IPsec peer. The IPsec peer is the other instance of the TOE when configured in the Multinode HA mode. The state information between the connected nodes is shared protected with IPsec.

1.3.3 TOE Type

The TOE is a network device implementing the security features required for exact conformance with the Base-PP, the Functional Package, and the PP-Modules identified in Table 1. The TOE is a network appliance which is neither a distributed nor a virtual network device.

1.3.4 Non-TOE Hardware, Software and Firmware

The TOE requires external IT devices to be properly operated. The TOE require the following items for operation:

- Network connectivity,
- Syslog server including a SSHv2 client for connecting to the TOE for the TOE to send audit logs,
- A management station with a SSHv2 client for remote administration of the TOE,
- A Certification Authority to use in the validation of the X.509 public key certificates by the TOE,
- NTP server for synchronizing TOE time when so configured, and
- A management station with a serial connection client for local administration of the TOE.

1.3.5 Disallowed Protocols and Services

The following protocols and services must not be used in association with the TOE:

- Telnet, FTP and SNMP are not considered secure. They violate the trusted path and trusted channel requirements and must not be used.
- SSL and TLS are not included in the certification and must not be used.
- The TOE must not be managed via J-Web, JUNOScript and JUNOScope. Neither is included in the certification. All administration of the TOE must be through the CLI.
- No user must be assigned super-user or FreeBSD root account privileges.

1.4 TOE Description

TOE Description consists of a description of the physical scope of the TOE and of the logical scope of the TOE.

1.4.1 Physical Scope of the TOE

The parts of the TOE included in the physical scope of the TOE are identified and described in Table 3.

Table 3 Parts Included in the Physical Scope of the TOE

Part of the TOE	Identification	Description
TOE Hardware	SRX4700, SRX4120 and SRX2300	The hardware platform and the casing of the TOE. Includes the processor, the memories, the physical ports, network cards, and the persistent storage.
TOE Software	Juniper Junos OS 25.2R1	The Junos OS included in the TOE is Juniper Junos OS 25.2R1. TOE software is distributed as an installation package. The following installation packages are used for different variants of the TOE: – SRX4700: junos-vmhost-install-srxhe2-x86-64-25.2R1.9.tgz – SRX4120: junos-vmhost-install-srxmr2-x86-64-25.2R1.9.tgz – SRX2300: junos-vmhost-install-srxmr2-x86-64-25.2R1.9.tgz
Security Guidance	Juniper SRX4700, SRX4120, SRX2300 for Junos OS 25.2R1 Common Criteria Guidance Supplement	The Common Criteria Guidance supplement for the TOE. The security guidance is distributed as a document in PDF format.

The variants of the TOE are a suite of network devices with an architecture illustrated in Figure 2. The TOE includes the chassis of the TOE. The Chassis implements the casing, the physical ports, and the hardware foundation for all those functions of the TOE which require hardware.

The KVM Hypervisor virtualizes the hardware for access by the software parts of the TOE. The software implements the routing engine and the packet forwarding engine of the TOE. Together, the two implement all routing plane and management plane functions of the TOE. Jointly, all software constitutes the Juniper Junos operating system.

The TOE is connected to the management console and to a syslog server. The management console may be local or remote. The TOE is also connected to the networks which it interconnects. Only the routing plane functions are

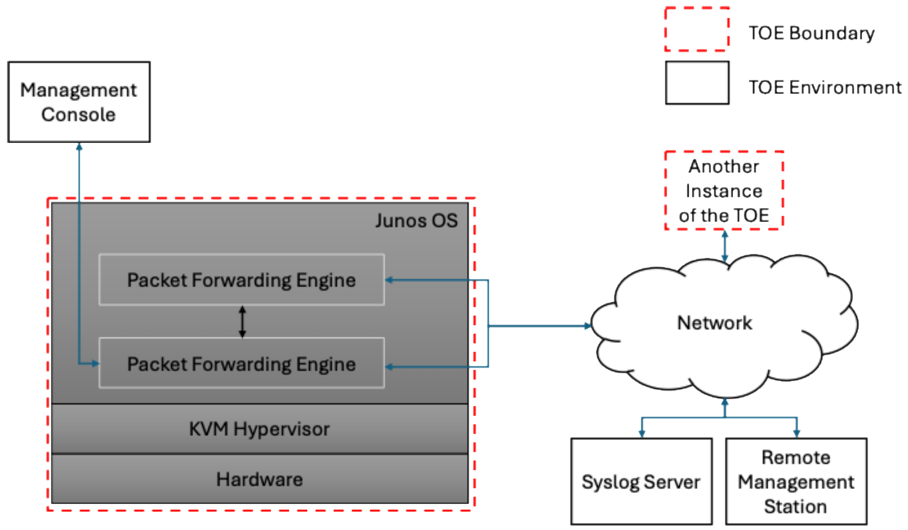

Figure 2 TOE Architecture

implemented on the network traffic to and from the interconnected networks. All management plane functions are implemented on the devices connected to the dedicated management ports of the TOE.

The TOE implements the following distinct sets of interfaces:

1. The operationally required interfaces, including power management, and the mechanical interfaces used for the cooling and ventilation of the TOE as well as the LEDs informing the user of the status of the TOE.
2. Network interfaces used for connecting the TOE to the interconnected networks. Network interfaces are the interfaces for the ingress and egress network traffic. They are physically separate from all other network interfaces. The TOE implements the networking functionality for the network traffic to traverse through it.
3. High Availability interfaces for the Multinode HA Mode.
4. Management interfaces are used by the administrators to manage the TOE. Management interface is through dedicated network ports and may be accessed locally from console or remotely over SSH. The management interface implements a CLI which is the only means of administering the TOE.

While offering identical functionality, the variants of the TOE offer different capabilities for different applications. The capabilities of different variants of the TOE are fully detailed in their datasheets given in Table 4.

Table 4 Hardware Capabilities of the Variants of the TOE

TOE Variant	Datasheet URL
SRX4700	https://www.juniper.net/documentation/us/en/hardware/srx4700/index.html
SRX4120	https://www.juniper.net/documentation/us/en/hardware/srx4120/index.html
SRX2300	https://www.juniper.net/documentation/us/en/hardware/srx2300/index.html

1.4.2 Multinode HA Mode

The Administrator of the TOE can set up the Multinode HA Mode for High Availability. Multinode HA Mode is common for both variants of the TOE. It supports two instances of the TOE presenting themselves as independent nodes to the rest of the network. The nodes are connected to adjacent infrastructure belonging to the same or different networks, all depending on the deployment mode. These nodes can either be collocated or separated across geographies. Participating nodes back up each other to ensure a fast synchronized failover in case of system or hardware failure.

Several models for network deployment of Multinode HA Mode are supported[1]. Two instances of the TOE are connected through the vNICs mapped to the physical NICs of the platform. The connection is through an Interchassis Link (ICL) which connects the nodes over a routed part (instead of a dedicated Layer 2 network seen in some high availability implementations). The nodes are configured by the administrator of the TOE with Netconf over SSH. Once the nodes are configured, the ICL communication is protected with IPsec. An example configuration of the TOE in a Multinode HA Mode over the ICL is given in Figure 3.

1.4.3 Logical Scope of the TOE

The TOE implements the security functionality required by a network appliance meeting the Base-PP, the Functional Package, and the PP-Modules. The major security features of the TOE are summarized in Table 5.

[1] Details are available at https://www.juniper.net/documentation/us/en/software/junos/high-availability/topics/topic-map/mnha-introduction.html

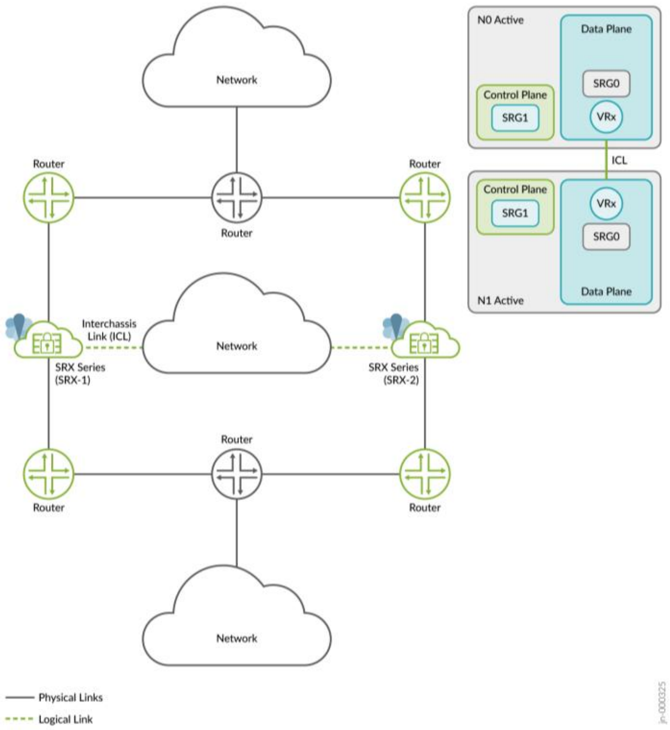

Figure 3 Example HA Configuration of the TOE

Table 5 Logical Scope of the TOE

Security Feature	Description
Security Audit	The TOE implements an audit function. A rich set of audit data is collected and stored as audit records. Each audit record includes a time stamp stating the exact time at which the audit record was generated. Each audit record also includes sufficient information to allow administrators of the TOE to examine the events and investigate possible security violations and attempts thereof. Audit records are stored in log files within the TOE. The administrator may also configure the TOE to forward the audit records to an external syslog server. The syslog server is not part of the TOE. Forwarding the audit records to a syslog server takes place over a trusted channel.
Cryptography	The TOE implements cryptography on software. The underlying cryptography for the trusted paths, trusted channels, and self-testing is implemented in software. All cryptographic libraries are linked to the TOE software. The TOE does not depend in the platform for the implementation of the cryptographic services. Each cryptographic algorithm implemented by the TOE is CAVP-validated. This fulfills the requirements of the NIAP Policy Letter #5: Applicability and Relationship of NIST Cryptographic Algorithm Validation Program (CAVP) and Cryptographic Module Validation Program (CMVP) to NIAP's Common Criteria Evaluation and Validation Scheme (CCEVS).

Identification, Authentication, Authorization and Access	The TOE does not implement general purpose computing facilities. Access is only granted to legitimate administrators. The TOE implements an authentication window for each attempted connection and displays an access banner on that window. The administrator may configure the content of the banner to inform unauthorized users of the restricted nature of access and the consequences of attempted unauthorized access. Each user is identified with a username and authenticated with a password. Only upon successful identification and authentication is the user granted access to the TOE. The TOE implements protective measures against attempted password guessing. Each user is assigned a retry counter which keeps track of the number of consecutive failed authentication attempts on that user account. If the number exceeds the administrator-configurable number of consecutive failed authentication attempts, the account is locked for a period of time. Each user may terminate their own session, and the TOE also implements an inactivity timer for each account. If the inactivity timer reaches the maximum allowed time of inactivity, the TOE terminates the session, and the user is required to re-authenticate to re-establish access.
Security Management	The TOE implements a CLI accessible to the successfully authenticated administrators. The CLI may be accessed locally from console or remotely over SSH. the CLI implements the entire human user interface of the TOE. There are no alternative methods of administering the TOE. Administrators may use the CLI for all security management tasks on the TOE.
Protection	The TOE protects itself by passive and active means. Passive protection is achieved through the construction of the TOE. The TOE is a dedicated software with restricted interfaces. It does not provide general computing capabilities. Access is restricted to authorized administrators and all administrator accesses are through a CLI. Administrators have no root access to the OS kernel. The network interfaces are mapped to interfaces which are physically separate from the management ports. Network interfaces may not be used for administering the TOE except when used for connecting to the TOE from a remote management station. The TOE implements a set of security measures for protecting the functions it implements and the configuration parameters. The TOE also maintains a clock which is used for generating time stamps and implementing various times used in the enforcement of security functions. The TOE implements self-tests at the start-up and takes protective measures in case of a failure of self-tests. Further, the TOE also allows upgrading of the software in case of vulnerabilities being discovered in the implementation.
High Availability	The logical scope of the TOE includes both the single mode and the Multinode HA Mode. The security functionality of the TOE includes the security functionality of a TOE which is configured to the Multinode HA Mode, and the security functionality of a TOE which is configured to a single. In the multinode HA Mode, the TOE which acts as an Active node is connected to the Backup node, and the TOE which acts as a backup node is connected to the Active node. The nodes are configured with Netconf over SSH. State sharing between the Active node and the Backup node is over an ICL protected with IPsec.

SSH Client and SSH Server	The TOE implements a SSH Server Client and a SSH Server. SSH allows the administrator to access the CLI from a remote management station, and a connection to the TOE from a syslog server to which audit records are forwarded. SSH is also used for protecting the Netconf communication use for the configuration of the nodes in the Multinode HA Mode. SSH ensures that each remote accesses are secure. The SSH implementation of the TOE allows both password-based and public key-based authentication and implements a suite of cryptographic algorithms allowed by the PP.
Firewall	The TOE implements the means to control information flows. Information flow control is based on the TOEs core function of forwarding network packets from source network entities to destination network entities. Traffic filters can be set to control information flows in accordance with the presumed identity of the source, the identity of the destination, the transport layer protocol, the source service identifier, and the destination service identifier (TCP or UDP port number). The TOE also implements a stateful network traffic filtering. Stateful traffic filtering is implemented through the examination of network packets and the application of information flow rules to each packet. Each packet is examined individually, and based on the packet information and the information flow rules, the TOE determines whether the packet is forwarded or dropped.
VPN Gateway	The TOE implements a VPN gateway with IPsec. IPsec is used for secure inteconnection of two instances of the TOE. The TOE implements IKEv2 to exchange keys between IPsec peers. X509-based certificates may be used for authenticating the peers in the IKE key exchange.
Intrusion Prevention	The administrator may configure the TOE to analyze IP-based network traffic forwarded to the TOE’s interfaces and detect violations of administrator defined IPS policies. The TOE is capable of a proactive response to terminate/interrupt an active potential threat and of a response in real time to interrupt a suspicious traffic flow. When deployed on an approved platform, the TOE constitutes a standalone network device. The entire functionality of the IPS is contained within the standalone TOE. This is referred to as Use Case 1 in the PP-Module.
Trusted Paths and Channels	The TOE implements trusted path and trusted channels for granting administrators secure accesses for managing the TOE remotely and for connecting the TOE to external IT systems. The administrators may connect to the TOE from a remote management station using SSH. The CLI is made accessible over the secure connection to successfully identified and authenticated administrators. The TOE may additionally be connected to from a syslog server over a trusted channel implemented using SSH. The SSH connection is used to protect all audit data relayed from the TOE to the syslog server. When configured in the Multinode HA Mode, the TOE implements a trusted channel to protect the state information shared between the Active node and the Backup node. The trusted channel is implemented using IPsec

2 Conformance Claims

This section states the Conformance Claims for the ST and the TOE, including a statement of the Conformance Claims, a statement of the Conformance Claim Rationale, and the Identification of the Technical Decisions applicable to the TOE.

2.1 Statement of Conformance Claims

The ST and the TOE claim conformance to Common Criteria Version 3.1 Revision 5, Part 1 through to Part 3 identified in the following:

- Common Criteria for Information Technology Security Evaluation, Part 1: Introduction and general model, April 2017, Version 3.1 Revision 5, CCMB-2017-04-001.
- Common Criteria for Information Technology Security Evaluation, Part 2: Security functional components, April 2017, Version 3.1 Revision 5, CCMB-2017-04-002.
- Common Criteria for Information Technology Security Evaluation, Part 3: Security assurance components, April 2017, Version 3.1 Revision 5, CCMB-2017-04-003.

The ST claims CC Part 2 conformance as CC Part 2 Extended.

The ST claims CC Part 3 conformance as CC Part 3 Conformant.

The ST claims conformance to the Base-PP which is the following Protection Profile:

- collaborative Protection Profile for Network Devices, Version: 3.0e, Date: 06-December-2023 (CPP_ND_V3.0E).

The ST claims conformance to the following PP-Modules:

- PP-Module for VPN Gateways, Version: 1.3, 2023-08-16 (MOD_VPNGW_v1.3),
- PP-Module for Stateful Traffic Filter Firewalls, Version 1.4 + Errata 20200625, 25-June-2020 (MOD_CPP_FW_V1.4E), and
- PP-Module for Intrusion Prevention Systems (IPS), Version: 1.0, 2021-05-11 (MOD_IPS_V1.0).

The ST claims conformance to the following PP-Configuration:

- PP-Configuration for Network Devices, Intrusion Prevention Systems, Stateful Traffic Filter Firewalls, and Virtual Private Network Gateways, Version 2.0, 2024-04-25 (CFG_NDcPP-IPS-FW-VPNGW_V2.0)

The ST claims conformance to the following Functional Package:

- Functional Package for SSH Version 1.0 (PKG_SSH_V1.0)

The ST claims no conformance to any Evaluation Assurance Level or any other security assurance requirement package. Security assurance requirements applicable to the TOE are those drawn from the Base-PP as required by the PP-Configuration.

The ST claims conformance to the Base-PP as PP-conformant.

The ST claims conformance to the PP-Configuration as PP-configuration-conformant.

The ST claims exact conformance to Base-PP, exact conformance to each PP-Module, exact conformance to the PP-Configuration, and exact conformance to the Functional Package. Exact conformance is defined in CC and CEM addenda for Exact Conformance, Selection-Based SFRs, and Optional SFRs, CCDB-013-v2.0 Final, 2021-Sep-30.

2.2 Conformance Claim Rationale

2.2.1 TOE Type Consistency Rationale

The TOE is a non-virtual and non-distributed network appliance. It implements a set of security features required for exact conformance with the Base-PP and with the PP-Modules.

The PP and the PP-Modules are used in accordance with the PP-Configuration. These are exactly the PP, the PP-Module, and the PP-configuration claimed in Sect. 2.1. The PP and the PP-Modules are exactly as identified in Sect. 1.3 of the PP-Configuration. This ensures that the TOE Type is consistent with the TOE Type in the Base-PP, PP-Modules, and PP-Configuration.

2.2.2 Security Problem Definition Consistency

The statement of the Security Problem Definition in this ST is reproduced exactly from the Base-PP and from the claimed PP-Modules. The resulting Security Problem Definition is a union of the Security Problem Definition of the Base-PP and those of the PP-Modules. There are no additional Security Problem Definition elements included in the statement of the Security Problem Definition. This ensures that the statement of the Security Problem Definition is consistent with the PP-Configuration.

2.2.3 Security Objective Consistency

The statement of the Security Objectives in this ST is reproduced exactly from the Base-PP and the PP-Modules. The resulting Security Objectives statement is a union of the Security Objectives of the Base-PP and those stated in the PP-Modules. There are no additional Security Objectives included in the statement of the Security Objectives. This ensures that the statement of the Security Objectives is consistent with the PP-Configuration.

2.2.4 Security Requirements Consistency

The security functional requirements are drawn exactly from the Base-PP and the PP-Modules. The statement of the security functional requirements includes all mandatory security requirements and those selection-based and optional security functional requirements applicable to the TOE. The developer does not include additional components in the statement of the security functional requirements. As such, the security functional requirements are consistently drawn from the Base-PP and the PP-Modules, and the ST ensures the consistency of the security functional requirements.

The security assurance requirements are drawn from the Base-PP only. This is consistent with Sect. 2.2 of the PP-Configuration. This ensures the consistency of the security assurance requirements.

2.3 Technical Decisions

Technical Decisions applicable to the Base-PP and each PP-Module are given separately. Each is given in a dedicated section below.

2.3.1 Technical Decisions Applicable to the Base-PP

The Technical Decisions (TD) applicable to the Base-PP are given in Table 6. For each TD which is not applicable, a brief justification for the exclusion is given.

Table 6 Technical Decisions Applicable to the Base-PP

TD	Description	Applicable	Exclusion Rationale (if applicable)

TD 1033	Sunset Dates for NDcPP Configurations	Yes	
TD 0990	NIT Technical Decision: CTR_DRBG in FCS_RBG_EXT.1.2	Yes	
TD 0973	NIT Technical Decision: FCS_(D)TLSS_EXT.1.3 Test 2 DHE Ciphersuite Conditionality	No	The ST does not claim TLS
TD 0923	NIT Technical Decision: Auditable event for FAU_STG_EXT.1 in FAU_GEN.1.2	Yes	
TD 0921	NIT Technical Decision: Addition of FIPS PUB 186-5 and Correction of Assignment	Yes	
TD 0900	NIT Technical Decision: Clarification to Local Administrator Access in FIA_UIA_EXT.1.3	Yes	
TD 0899	NIT Technical Decision: Correction of Renegotiation Test for TLS 1.2	No	The ST does not claim TLS.
TD 0886	Clarification to FAU_STG_EXT.1 Test 6	Yes	
TD 0880	NIT Decision: Removal of Duplicate Selection in FMT_SMF.1.1	Yes	
TD 0879	NIT Decision: Correction of Chapter Headings in CPP_ND_V3.0E	Yes	
TD 0868	NIT Technical Decision: Clarification of time frames in FCS_IPSEC_EXT.1.7 and FCS_IPSEC_EXT.1.8	No	IPsec is claimed through MOD_VPNGW_v1.3, not the Base-PP. Therefore, the IPsec-specific TD to the Base-PP are not applicable.
TD 0836	NIT Technical Decision: Redundant Requirements in FPT_TST_EXT.1	Yes	

2.3.2 Technical Decisions Applicable to the Functional Package

The Technical Decisions applicable to the PKG_SSH_V1.0 and their applicability to the TOE is given in Table 7.

Table 7 Technical Decisions Applicable to the Functional Package

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 1023	Clarifications to FCS_SSH_EXT.1.5 When aes256-gcm@openssh.com Is Selected	Yes	
TD 1015	Addition of ECD to PKG_SSH_V1.0	Yes	
TD 0967	Allowance of Kex-strict in PKG_SSH_V1.0	Yes	
TD 0909	Updates to FCS_SSH_EXT.1.1 App Note in SSH FP 1.0	Yes	

TD 0777	Clarification to Selections for Auditable Events for FCS_SSH_EXT.1	Yes	
TD 0732	FCS_SSHS_EXT.1.3 Test 2 Update	Yes	
TD 0695	Choice of 128 or 256 bit size in AES-CTR in SSH Functional Package.	Yes	
TD 0682	Addressing Ambiguity in FCS_SSHS_EXT.1 Tests	Yes	

2.3.3 Technical Decisions Applicable to MOD_IPS_V1.0

Technical decisions applicable to MOD_IPS_V1.0 and their applicability to the TOE is given in Table 8.

Table 8 Technical Decisions Applicable to MOD_IPS_V1.0

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 0940	Updated Conformance Claims for MOD_IPS	Yes	
TD 0902	Updating RFC 2460 to 8200 in MOD_IPS_V1.0	Yes	
TD 0828	Aligning MOD_IPS_V1.0 with CPP_ND_V3.0E	Yes	
TD 0722	IPS_SBD_EXT.1.1 EA Correction	Yes	
TD 0595	Administrative corrections to IPS PP-Module	Yes	

2.3.4 Technical Decisions Applicable to MOD_CPP_FW_V1.4E

Technical decisions applicable to MOD_CPP_FW_V1.3E and their applicability to the TOE is given in Table 9

Table 9 Technical Decisions Applicable to MOD_CPP_FW_V1.4E

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 0924	NIT Technical Decision: FFW_RUL_EXT.1.2 Expected Rule Granularity Level	Yes	
TD 0827	Aligning MOD_CPP_FW_v1.4E with CPP_ND_V3.0E	Yes	
TD 0551	NIT Technical Decision for Incomplete Mappings of OEs in FW Module v1.4+Errata	Yes	

TD 0545	NIT Technical Decision for Conflicting FW rules cannot be configured (extension of RfI#201837)	Yes	

2.3.5 Technical Decisions Applicable to MOD_VPNGW_v1.3

The technical decisions applicable to MOD_VPNGW_v1.3 and their applicability to the TOE given in Table 10.

Table 10 Technical Decisions Applicable to MOD_VPNGW_v1.3

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 1034	Clarification on RFC 8784 in FIA_PSK_EXT.1	Yes	
TD 0986	Alignment of Updated IPsec Changes to MOD_VPNGW_V1.3	Yes	
TD 0944	Adding FIPS 186-5 in MOD_VPNGW_V1.3	Yes	
TD 0941	Updated Conformance Claims for MOD_VPNGW 1.3	Yes	
TD 0838	PPK Configurability in FIA_PSK_EXT.1.1	No	The statement of TD 1034 claims that it supersedes TD 0838 which is now archived. Yet, TD 0838 is included at the NIAP site for MOD_VPNGW_V1.3 as an applicable Technical Decisions. Therefore, it is listed here but, being superseded by TD 1034, considered not applicable.
TD 0824	Aligning MOD_VPNGW 1.3 with NDcPP 3.0E	Yes	
TD 0811	Correction to Referenced SFR in FIA_PSK_EXT.3 Test	No	The ST does not claim FIA_PSK_EXT.3
TD 0781	Correction to FIA_PSK_EXT.3 EA for MOD_VPNGW_v1.3	No	The ST does not claim FIA_PSK_EXT.3

3 Security Problem Definition

The Security Problem Definition includes a statement of the Threats, Assumptions and OSPs applicable to the TOE. Each is stated in this section. The Security Problem Definition is fully drawn from the Base-PP and the PP-Modules. There are no additional Security Problem Definition elements defined in the Functional Package.

3.1 Threats

The threats applicable to the TOE are drawn from the Base-PP and the PP-Modules. There are no additional threats defined in the Functional Package. There are no additions or omissions, and the wording of each threat statement is taken verbatim.

3.1.1 Threats Drawn from the Base-PP

The statement of threats drawn from the Base-PP is given in Table 11.

Table 11 Threats Drawn from the Base-PP

Threat ID	Threat Statement
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.

3.1.2 Threats Drawn from MOD_IPS_V1.0

The threats drawn from MOD_IPS_V1.0 are stated in Table 12.

Table 12 Threats Drawn from MOD_IPS_V1.0

Threat ID	Threat Statement
T.NETWORK_ACCESS	Unauthorized access may be achieved to services on a protected network from outside that network, or alternately services outside a protected network from inside the protected network. If malicious external devices are able to communicate with devices on the protected network via a backdoor then those devices may be susceptible to the unauthorized disclosure of information
T.NETWORK_DISCLOSURE	Sensitive information on a protected network might be disclosed resulting from ingress- or egress-based actions.
T.NETWORK_DOS	Attacks against services inside a protected network, or indirectly by virtue of access to malicious agents from within a protected network, might lead to denial of services otherwise available within a protected network.
T.NETWORK_MISUSE	Access to services made available by a protected network might be used counter to operational environment policies. Devices located outside the protected network may attempt to conduct inappropriate activities while communicating with allowed public services (e.g. manipulation of resident tools,

	SQL injection, phishing, forced resets, malicious zip files, disguised executables, privilege escalation tools, and botnets).

3.1.3 Threats Drawn from MOD_CPP_FW_V1.4E

The threats drawn from MOD_CPP_FW_V1.4E are stated in Table 13.

Table 13 Threats Drawn from MOD_CPP_FW_V1.4E

Threat ID	Threat Statement
T.NETWORK_DISCLOSURE	An attacker may attempt to "map" a subnet to determine the machines that reside on the network, and obtaining the IP addresses of machines, as well as the services (ports) those machines are offering. This information could be used to mount attacks to those machines via the services that are exported.
T.NETWORK_ACCESS	With knowledge of the services that are exported by machines on a subnet, an attacker may attempt to exploit those services by mounting attacks against those services.
T.NETWORK_MISUSE	An attacker may attempt to use services that are exported by machines in a way that is unintended by a site's security policies. For example, an attacker might be able to use a service to "anonymize" the attacker's machine as they mount attacks against others.
T.MALICIOUS_TRAFFIC	An attacker may attempt to send malformed packets to a machine in hopes of causing the network stack or services listening on UDP/TCP ports of the target machine to crash.

3.1.4 Threats Drawn from MOD_VPNGW_v1.3

The threats drawn from MOD_VPNGW_v1.3 are stated in Table 14.

Table 14 Threats Drawn from MOD_VPNGW_v1.3

Threat ID	Threat Statement
T.DATA_INTEGRITY	Devices on a protected network may be exposed to threats presented by devices located outside the protected network that may attempt to modify the data without authorization. If known malicious external devices are able to communicate with devices on the protected network or if devices on the protected network can communicate with those external devices then the data contained in the communications may be susceptible to a loss of integrity.
T.NETWORK_ACCESS	Devices located outside the protected network may seek to exercise services located on the protected network that are intended to only be accessed from inside the protected network or only accessed by entities using an authenticated path into the protected network. Devices located outside the protected network may, likewise, offer services that are inappropriate for access from within the protected network.

	From an ingress perspective, VPN gateways can be configured so that only those network servers intended for external consumption by entities operating on a trusted network (e.g., machines operating on a network where the peer VPN gateways are supporting the connection) are accessible and only via the intended ports. This serves to mitigate the potential for network entities outside a protected network to access network servers or services intended only for consumption or access inside a protected network. From an egress perspective, VPN gateways can be configured so that only specific external services (e.g., based on destination port) can be accessed from within a protected network, or moreover are accessed via an encrypted channel. For example, access to external mail services can be blocked to enforce corporate policies against accessing uncontrolled email servers, or, that access to the mail server must be done over an encrypted link.
T.NETWORK_DISCLOSURE	Devices on a protected network may be exposed to threats presented by devices located outside the protected network, which may attempt to conduct unauthorized activities. If known malicious external devices are able to communicate with devices on the protected network, or if devices on the protected network can establish communications with those external devices (e.g., as a result of a phishing episode or by inadvertent responses to email messages), then those internal devices may be susceptible to the unauthorized disclosure of information. From an infiltration perspective, VPN gateways serve not only to limit access to only specific destination network addresses and ports within a protected network, but whether network traffic will be encrypted or transmitted in plaintext. With these limits, general network port scanning can be prevented from reaching protected networks or machines, and access to information on a protected network can be limited to that obtainable from specifically configured ports on identified network nodes (e.g., web pages from a designated corporate web server). Additionally, access can be limited to only specific source addresses and ports so that specific networks or network nodes can be blocked from accessing a protected network thereby further limiting the potential disclosure of information. From an exfiltration perspective, VPN gateways serve to limit how network nodes operating on a protected network can connect to and communicate with other networks limiting how and where they can disseminate information. Specific external networks can be blocked altogether or egress could be limited to specific addresses or ports. Alternately, egress options available to network nodes on a protected network can be carefully managed in order to, for example, ensure that outgoing connections are encrypted to further mitigate inappropriate disclosure of data through packet sniffing.
T.NETWORK_MISUSE	Devices located outside the protected network, while permitted to access particular public services offered inside the protected network, may attempt to conduct inappropriate activities while communicating with those allowed public services. Certain services offered from within a protected network may also represent a risk when accessed from outside the protected network. From an ingress perspective, it is generally assumed that entities operating on external networks are not bound by the use policies for a given protected network. Nonetheless, VPN gateways can log policy violations that might indicate violation of publicized usage statements for publicly available services. From an

	egress perspective, VPN gateways can be configured to help enforce and monitor protected network use policies. As explained in the other threats, a VPN gateway can serve to limit dissemination of data, access to external servers, and even disruption of services – all of these could be related to the use policies of a protected network and as such are subject in some regards to enforcement. Additionally, VPN gateways can be configured to log network usages that cross between protected and external networks and as a result can serve to identify potential usage policy violations.
T.REPLAY_ATTACK	If an unauthorized individual successfully gains access to the system, the adversary may have the opportunity to conduct a "replay" attack. This method of attack allows the individual to capture packets traversing throughout the network and send the packets at a later time, possibly unknown by the intended receiver. Traffic is subject to replay if it meets the following conditions: • Cleartext: an attacker with the ability to view unencrypted traffic can identify an appropriate segment of the communications to replay as well in order to cause the desired outcome • No integrity: alongside cleartext traffic, an attacker can make arbitrary modifications to captured traffic and replay it to cause the desired outcome if the recipient has no means to detect these

3.2 Assumptions

The assumptions applicable to the TOE are drawn from the Base-PP and the PP-Modules. There are no additions or omissions, and the wording of each assumption statement is taken verbatim.

3.2.1 Assumptions Drawn from the Base-PP

The assumptions drawn from the Base-PP are stated in Table 15.

Table 15 Assumptions Drawn from the Base-PP

Assumption ID	Assumption Statement
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.

3.2.2 Assumptions Drawn from MOD_IPS_V1.0

The assumptions drawn from MOD_IPS_V1.0 are given in Table 16. Assumption A.NO_THRU_TRAFFIC_PROTECTION drawn from the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not to those defined in the PP-Module.

Table 16 Assumptions Drawn from MOD_IPS_V1.0

Assumption ID	Assumption Statement
A.CONNECTIONS	It is assumed that the TOE is connected to distinct networks in a manner that ensures that the TOE's security policies will be enforced on all applicable network traffic flowing among the attached networks.

3.2.3 Assumptions Drawn from MOD_CPP_FW_V1.4E

There are no additional assumptions defined in MOD_CPP_FW_V1.4E. Assumption A.NO_THRU_TRAFFIC_PROTECTION drawn from the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not to those defined in the PP-Module.

3.2.4 Assumptions Drawn from MOD_VPNGW_v1.3

The assumptions drawn from MOD_VPNGW_v1.3 are given in Table 17. Assumption A.NO_THRU_TRAFFIC_PROTECTION drawn from the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not to those defined in the PP-Module.

Table 17 Assumptions Drawn from MOD_VPNGW_v1.3

Assumption ID	Assumption Statement
A.CONNECTIONS	It is assumed that the TOE is connected to distinct networks in a manner that ensures that the TOE's security policies will be enforced on all applicable network traffic flowing among the attached networks.

3.3 Organizational Security Policies

The Organizational Security Policies (OSP) applicable to the TOE are drawn from the Base-PP and the PP-Modules. There are no additions or omissions, and the wording of each OSP statement is taken verbatim.

3.3.1 OSPs Drawn from the Base-PP

The OSPs drawn from the Base-PP are given in Table 18.

Table 18 OSPs Drawn from the Base-PP

OSP ID	OSP Statement
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.

3.3.2 OSPs Drawn from MOD_IPS_V1.0

The OSPs drawn from MOD_IPS_V1.0 are given in Table 19.

Table 19 OSPs Drawn from MOD_IPS_V1.0

OSP ID	OSP Statement
P.ANALYZE	Analytical processes and information to derive conclusions about potential intrusions must be applied to IPS data and appropriate response actions taken.

3.3.3 OSPs Drawn from MOD_CPP_FW_V1.4E

There are no additional OSPs defined in [MOD_CPP_FW_V1.4E].

3.3.4 OSPs Drawn from MOD_VPNGW_v1.3

There are no additional OSPs defined in [MOD_VPNGW_v1.3].

4 Security Objectives

The security objectives are stated for the TOE Sect. 4.1 and for the operational environment of the TOE in Sect. 4.2. The security objectives rationale is given in Sect. 4.3. The security objectives are drawn from the Base-PP and the PP-Modules. There are no additional Security Objectives defined in the Functional Package.

4.1 Security Objectives for the TOE

The security objectives for the TOE are drawn from the Base-PP and the PP-Modules. Each are stated below.

4.1.1 Security Objectives for the TOE Drawn from the Base-PP

There are no security objectives for the TOE explicitly stated in the Base-PP.

4.1.2 Security Objectives for the TOE Drawn from MOD_IPS_V1.0

Security objectives for the TOE drawn from MOD_IPS_V1.0 are given in Table 20.

Table 20 Security Objectives for the TOE Drawn from MOD_IPS_V1.0

Security Objective ID	Security Objective Statement
O.IPS_ANALYZE	Entities that reside on or communicate across monitored networks must have network activity effectively analyzed for potential violations of approved network usage. The TOE must be able to effectively analyze data collected from monitored networks to reduce the risk of unauthorized disclosure of information, inappropriate access to services, and misuse of network resources.
O.IPS_REACT	The TOE must be able to react in real-time as configured by the Security Administrator to terminate and block traffic flows that have been determined to violate administrator-defined IPS policies.
O.SYSTEM_MONITORING	To be able to analyze and react to potential network policy violations, the IPS must be able to collect and store essential data elements of network traffic on monitored networks.
O.TOE_ADMINISTRATION	To address the threat of unauthorized administrator access that is defined in the Base-PP, conformant TOEs will provide the functions necessary for an administrator to configure the IPS capabilities of the TOE.

4.1.3 Security Objectives for the TOE Drawn from MOD_CPP_FW_V1.4E

The security objectives for the TOE drawn from MOD_CPP_FW_V1.4E are given in Table 21.

Table 21 Security Objectives for the TOE Drawn from MOD_CPP_FW_V1.4E

Security Objective ID	Security Objective Statement
O.,RESIDUAL_ INFORMATION	The TOE shall implement measures to ensure that any previous information content of network packets sent through the TOE is made unavailable either upon

	deallocation of the memory area containing the network packet or upon allocation of a memory area for a newly arriving network packet or both.
O.STATEFUL_TRAFFIC_FILTERING	The TOE shall perform stateful traffic filtering on network packets that it processess. For this the TOE shall support the definition of stateful traffic filtering rules that allow to permit or drop network packets. The TOE shall support assignment of the stateful traffic filtering rules to each distinct network interface. The TOE shall support the processing of the applicable stateful traffic filtering rules in an administratively defined order. The TOE shall deny the flow of network packets if no matching stateful traffic filtering rule is identified. Depending on the implementation, the TOE might support the stateful traffic filtering of Dynamic Protocols (optional).

4.1.4 Security Objectives for the TOE Drawn from MOD_VPNGW_v1.3

The security objectives for the TOE drawn from MOD_VPNGW_v1.3 are given in Table 22.

Table 22 Security Objectives for the TOE Drawn from MOD_VPNGW_v1.3

Security Objective ID	Security Objective Statement
O.,ADDRESS_FILTERING	To address the issues associated with unauthorized disclosure of information, inappropriate access to services, misuse of services, disruption or denial of services, and network-based reconnaissance, compliant TOE's will implement packet filtering capability. That capability will restrict the flow of network traffic between protected networks and other attached networks based on network addresses of the network nodes originating (source) or receiving (destination) applicable network traffic as well as on established connection information.
O.AUTHENTICATION	To further address the issues associated with unauthorized disclosure of information, a compliant TOE's authentication ability (IPSec) will allow a VPN peer to establish VPN connectivity with another VPN peer and ensure that any such connection attempt is both authenticated and authorized. VPN endpoints authenticate each other to ensure they are communicating with an authorized external IT entity.
O.CRYPTOGRAPHIC_FUNCTIONS	To address the issues associated with unauthorized disclosure of information, inappropriate access to services, misuse of services, disruption of services, and network-based reconnaissance, compliant TOE's will implement cryptographic capabilities. These capabilities are intended to maintain confidentiality and allow for detection and modification of data that is transmitted outside of the TOE.
O.FAIL_SECURE	There may be instances where the TOE's hardware malfunctions or the integrity of the TOE's software is compromised, the latter being due to malicious or non-malicious intent. To address the concern of the TOE operating outside of its hardware or software specification, the TOE will shut down upon discovery of a problem reported via the self-test mechanism and provide signature-based validation of updates to the TSF.
O.PORT_FILTERING	To further address the issues associated with unauthorized disclosure of information, etc., a compliant TOE's port filtering capability will restrict the flow of network traffic between protected networks and other attached networks based on the originating

	(source) or receiving (destination) port (or service) identified in the network traffic as well as on established connection information.
O.SYSTEM_MONITORING	To address the issues of administrators being able to monitor the operations of the VPN gateway, it is necessary to provide a capability to monitor system activity. Compliant TOEs will implement the ability to log the flow of network traffic. Specifically, the TOE will provide the means for administrators to configure packet filtering rules to 'log' when network traffic is found to match the configured rule. As a result, matching a rule configured to 'log' will result in informative event logs whenever a match occurs. In addition, the establishment of security associations (SAs) is auditable, not only between peer VPN gateways, but also with certification authorities (CAs).
O.TOE_ADMINISTRATION	TOEs will provide the functions necessary for an administrator to configure the packet filtering rules, as well as the cryptographic aspects of the IPsec protocol that are enforced by the TOE.

4.2 Security Objectives for the Operational Environment

The security objectives for the operational environment are drawn from the Base-PP and the PP-Modules.

4.2.1 Security Objectives for the Operational Environment Drawn from the Base-PP

The security objectives for the operational environment are drawn in verbatim and are stated in Table 23.

Table 23 Security Objectives for the Operational Environment Drawn from the Base-PP

Security Objective ID	Security Objective Statement
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_ PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE. 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.

4.2.2 Security Objectives for the Operational Environment Drawn from MOD_IPS_V1.0

Security objectives for the operational environment drawn from MOD_IPS_V1.0 are given in Table 24. Security objective OE.NO_THRU_TRAFFIC_PROTECTION defined in the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not those defined in the PP-Module.

Table 24 Security Objectives for the Operational Environment Drawn from MOD_IPS_V1.0

Security Objective ID	Security Objective Statement
OE.CONNECTIONS	TOE administrators will ensure that the TOE is installed in a manner that will allow the TOE to effectively enforce its policies on network traffic of monitored networks.

4.2.3 Security Objectives for the Operational Environment Drawn from MOD_CPP_FW_V1.4E

There are no additional security objectives for the operational environment defined in MOD_CPP_FW_V1.4E. Security objective OE.NO_THRU_TRAFFIC_PROTECTION defined in the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not those defined in the PP-Module.

4.2.4 Security Objectives for the Operational Environment Drawn from MOD_VPNGW_v1.3

The security objectives for the operational environment drawn from MOD_VPNGW_v1.3 are given in Table 25. Security objective OE.NO_THRU_TRAFFIC_PROTECTION defined in the Base-PP only applies to the interfaces in the TOE that are defined by the Base-PP, not those defined in the PP-Module.

Table 25 Security Objectives for the Operational Environment Drawn from MOD_VPNGW_v1.3

Security Objective ID	Security Objective Statement
OE.CONNECTIONS	The TOE is connected to distinct networks in a manner that ensures that the TOE security policies will be enforced on all applicable network traffic flowing among the attached networks.

4.3 Security Objectives Rationale

The statement of the security problem definition and the statements of the security objectives are drawn verbatim from the Base-PP and the PP-Modules. The security objectives rationale in each is directly applicable to the ST. They are not repeated.

5 Security Requirements

This section states the security requirements applicable to the TOE. The statement commences with the extended components definition in Sect. 5.1. The statement of the extended components is followed by the statement of the notations and conventions used in the expression of the security requirements. The security functional requirements are summarized in Sect. 5.3 and stated on a per functional class basis for the Base-PP, the Functional Package where applicable, and each PP-Module. The mandatory security functional requirements are stated first, followed by the statement of the selection-based and optional requirements. The security assurance requirements are given in Sect. 5.5. The security requirements rationale is given in Sect. 5.20.

5.1 Extended Components Definition

The ST references several extended components. Each one is taken verbatim from the following:

- Appendix C of the Base-PP,
- Sect. 3.2 of the Functional Package,
- Appendix C of MOD_IPS_V1.0,
- Appendix C of MOD_CPP_FW_V1.4E, and
- Appendix C of MOD_VPNGW_v1.3.

The statement of the extended components is not altered and is applicable without any modifications. it is not repeated here.

5.2 Notation and Conventions

This ST follows primarily the conventions of the Base-PP in the expression of the Security Functional Requirements:

- Unaltered Security Functional Requirements are stated using the notation given in CC Part 2 or in the applicable extended component definition.
- For each refinement, the added text is indicated with a **bold font.** Removal of text is indicated with a ~~strikethrough~~.
- For each selection, the selected values are indicated with <u>underlined text</u>.
    - For example, a selection "[selection: disclosure, modification, loss of use]" in a Security Functional Requirement might become "[<u>disclosure</u>]" when the selection is performed in the ST.
- Each assignment is indicated with *italicized font*.
- Each assignment within a selection is indicated with *<u>italicized and underlined font</u>*.
    - For example, an assignment within a selection "[selection: change_default, query, modify, delete, [assignment: other operations]]" in a Security Functional Requirement might become "[<u>change_default</u>,[*<u>select_tag</u>*]]" when the selection and assignment are performed in the ST.
- Iteration is indicated by adding a descriptive string starting with "/" (e.g. "FCS_COP.1/Hash").
- Each extended Security Functional Requirement is indicated with a label "_EXT" in the end of the requirement name (e.g. FCS_RBG_EXT) following the notation in the Extended Components Definition.

When the Base-PP, an applicable Functional Package, or a PP-Module uses an alternative notation or expression in the statement of a Security Functional Requirements, that notation or expression is followed in the ST even if deviating from the above conventions. For example, the capitalization of the component names is followed in verbatim even if sometimes inconsistent.

The notation for expressing the Security Assurance Requirements is taken verbatim from the Base-PP for compliance with the PP-Configuration.

5.3 Security Functional Requirements Summary

The Security Functional Requirements applicable to the TOE may be mandatory, selection-based, or optional. Each is summarized in Table 26.

Table 26 SFR Summary

Mandatory Security Functional Requirements Drawn from the Base-PP	
Security Functional Class	**Security Functional Components**
FAU: Security Audit	FAU_GEN.1 Audit Data Generation FAU_GEN.2 User Identity Association FAU_STG_EXT.1 Protected Audit Event Storage
FCS: Cryptographic Support	FCS_CKM.1 Cryptographic Key Generation FCS_CKM.2 Cryptographic Key Establishment FCS_CKM.4 Cryptographic Key Destruction FCS_COP.1/DataEncryption Cryptographic Operation (AES Data Encryption/Decryption) FCS_COP.1/SigGen Cryptographic Operation (Signature Generation and Verification) FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm) FCS_COP.1/KeyedHash Cryptographic Operation (Keyed Hash Algorithm) FCS_IPSEC_EXT.1 IPsec Protocol FCS_RBG_EXT.1 Random Bit Generation
FIA: Identification and Authentication	FIA_UIA_EXT.1 User Identification and Authentication FIA_X509_EXT.1/Rev X.509 Certificate Validation FIA_X509_EXT.2 X.509 Certificate Authentication FIA_X509_EXT.3 X.509 Certificate Requests
FMT: Security Management	FMT_MOF.1/ManualUpdate Management of Security Functions Behaviour 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: Protection of the TSF	FPT_SKP_EXT.1 Protection of TSF Data (for reading of all symmetric keys) FPT_STM_EXT.1 Reliable Time Stamps FPT_TST_EXT.1 TSF Testing FPT_TUD_EXT.1 Trusted Update

FTA: TOE Access	FTA_SSL.3 TSF-Initiated Termination FTA_SSL.4 User-Initiated Termination FTA_TAB.1 Default TOE Access Banners
FTP: Trusted Path/Channels	FTP_ITC.1 Inter-TSF Trusted Channel FTP_TRP.1/Admin Trusted Path
Selection-Based Security Functional Requirements Drawn from the Base-PP	
Security Functional Class	**Security Functional Components**
FCS: Cryptographic Support	FCS_NTP_EXT.1 NTP Protocol
FIA: Identification and Authentication	FIA_AFL.1 Authentication Failure Handling FIA_PMG_EXT.1 Password Management FIA_UAU.7 Protected Authentication Feedback
FMT: Security Management	FMT_MOF.1/Services Management of Security Functions Behaviour FMT_MOF.1/Functions Management of Security Functions Behaviour
FPT: Protection of the TST	FPT_APW_EXT.1 Protection of Administrator Passwords
FTA: TOE Access	FTA_SSL_EXT.1 TSF-initiated Session Locking
Optional Security Functional Requirements Drawn from the Base-PP	
Security Functional Class	**Security Functional Components**
None.	
Mandatory Security Functional Requirements Drawn from the Functional Package	
FCS: Cryptographic Support	FCS_SSH_EXT.1 SSH Protocol
Selection-Based Security Functional Requirements Drawn from the Functional Package	
FCS: Cryptographic Support	FCS_SSHC_EXT.1 SSH Protocol -- Client FCS_SSHS_EXT.1 SSH Protocol -- Server
Optional Security Functional Requirements Drawn from the Functional Package	
None.	
Mandatory Security Functional Requirements Drawn from MOD_IPS_V1.0	
FAU: Security Audit	FAU_GEN.1/IPS Audit Data Generation (IPS)
FMT: Security Management	FMT_SMF.1/IPS Specification of Management Functions (IPS)
IPS: Intrusion Prevention	IPS_ABD_EXT.1 Anomaly-Based IPS IPS_IPB_EXT.1 IP Blocking IPS_NTA_EXT.1 Network Traffic Analysis IPS_SBD_EXT.1 Signature-Based IPS Functionality
Selection-Based Security Functional Requirements Drawn from MOD_IPS_V1.0	
None.	

Optional Security Functional Requirements Drawn from MOD_IPS_V1.0	
None.	
Mandatory Security Functional Requirements Drawn from MOD_CPP_FW_V1.4E	
FDP: User Data Protection	FDP_RIP.2 Full Residual Information Protection
FFW: Firewall	FFW_RUL_EXT.1 Stateful Traffic Filtering
FMT: Security Management	FMT_SMF.1/FFW Specification of Management Functions
Selection-Based Security Functional Requirements Drawn from MOD_CPP_FW_V1.4	
None.	
Optional Security Functional Requirements Drawn from MOD_CPP_FW_V1.4	
None.	
Mandatory Security Functional Requirements Drawn from MOD_VPNGW_v1.3	
FAU: Security Audit	FAU_GEN.1/VPN Audit Data Generation (VPN Gateway)
FCS: Cryptographic Support	FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication)
FMT: Security Management	FMT_SMF.1/VPN Specification of Management Functions
FPF: Packet Filtering	FPF_RUL_EXT.1 Packet Filtering Rules
FPT: Protection of the TSF	FPT_FLS.1/SelfTest Failure with Preservation of Secure State (Self-Test Failures) FPT_TST_EXT.3 Self-Test with Defined Methods
FTP: Trusted Path/Channels	FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications)
Selection-Based Security Functional Requirements Drawn from MOD_VPNGW_v1.3	
FIA: Identification and Authentication	FIA_PSK_EXT.1 Pre-Shared Key Composition FIA_PSK_EXT.2 Generated Pre-Shared Keys
Optional Security Functional Requirements Drawn from MOD_VPNGW_v1.3	
None.	

5.4 Mandatory Security Functional Requirements Drawn from the Base-PP

5.4.1 Security Audit (FAU)

5.4.1.1 Security Audit Data Generation (FAU_GEN)

FAU_GEN.1 Audit data generation (Refinement)

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 <u>not specified</u> 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 Administrator account shall be logged)];*

d) Specifically defined auditable events listed in Table 27.

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

e) For each audit event type, based on the auditable event definitions of the functional components included in the **c**PP/ST, *information specified in column three of* Table 27.

Table 27 Security Functional Requirements and Auditable Events

Mandatory Requirements		
Requirement	**Auditable Events**	**Additional Audit Record Contents**
FAU_GEN.1	None.	None.
FAU_GEN.2	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_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure
FCS_RBG_EXT.1	None.	None.
FDP_RIP.2[2]	None.	None.
FFW_RUL_EXT.1[3]	Application of rules configured with the "log" operation	Source and destination addresses Source and destination ports Transport Layer Protocol TOE Interface

[2] Drawn from MOD_CPP_FW_V1.4E
[3] Drawn from MOD_CPP_FW_V1.4E

FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address).
FIA_X509_EXT.1/Rev	• Unsuccessful attempt to validate a certificate • Any addition, replacement or removal of trust anchors in the TOE's trust store	• Reason for failure of certificate validation • identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.2	None	None
FIA_X509_EXT.3	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.
FMT_MTD.1/CoreData	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMF.1/FFW[4]	All management activities of TSF data (including creation, modification and deletion of firewall rules).	None.
FMT_SMR.2	None.	None.
FPT_SKP_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None.
FTA_SSL.4	The termination of an interactive session.	None.
FTA_TAB.1	None.	None.
FTP_ITC.1	• Initiation of the trusted channel. • Termination of the trusted channel.	• None • None • Reason for failure

[4] Drawn from MOD_CPP_FW_V1.4E

	• Failure of the trusted channel functions.	
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
Selection-Based Requirements Drawn from the Base-PP		
Requirement	**Auditable Events**	**Additional Audit Record Contents**
FCS_NTP_EXT.1	• Configuration of a new time server • Removal of a configured time server	Identity if new/removed time server
FCS_SSH_EXT.1[5,6]	[Failure to establish SSH connection]	[Reason for failure and Non-TOE endpoint of attempted connection (IP Address), *None*] Application Note: The "Reason for Failure and Non-TOE endpoint of attempted connection (IP Address)" applies when the TOE is operating as a SSH Server. The "None" applies with the TOE is operating as a SSH Client.
FCS_SSH_EXT.1[7]	[Establishment of SSH connection]	[Non-TOE endpoint of connection (IP Address), *None*] Application Note: The "Non-TOE endpoint of connection (IP Address)" applies when the TOE is operating as a SSH Server. The "None" applies with the TOE is operating as a SSH Client.
FCS_SSH_EXT.1[8]	[Termination of SSH connection session]	[Non-TOE endpoint of connection (IP Address), *None*] Application Note: The "Non-TOE endpoint of connection (IP Address)" applies when the TOE is operating as a SSH Server. The

[5] Drawn from the Functional Package
[6] As per TD 0777
[7] Drawn from the Functional Package
[8] Drawn from the Functional Package

		"None" applies with the TOE is operating as a SSH Client.
FCS_SSH_EXT.1[9]	[<u>Dropping of packet(s) outside defined size limits</u>]	[<u>Packet size</u>]
FCS_SSHC_EXT.1[10]	No events specified	
FCS_SSHS_EXT.1[11]	No events specified	
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT1	None.	None.
FIA_UAU.7	None.	None.
FMT_MOF.1/Services	None.	None.
FMT_MTD.1/CryptoKeys	None.	None.
FMT_MOF.1/Functions	None.	None.
FPT_APW_EXT.1	None.	None.
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session lock	None.
Optional Requirements		
Requirement	**Auditable Events**	**Additional Audit Record Contents**
None.		

FAU_GEN.2 User identity association

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

5.4.1.2 Security audit event storage (Extended - FAU_STG_EXT)

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*.

]

[9] Drawn from the Functional Package
[10] Drawn from the Functional Package
[11] Drawn from the Functional Package

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 [*an Administrator-configurable number of up to 1000 log files, the size of each being Administrator-configurable within the constraints of the memory available in the TOE*].

FAU_STG_EXT.1.5 The TSF shall [*overwrite previous audit records according to the following rule:* [*oldest log file is overwritten*]] 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 [ability to view locally].

5.4.2 Cryptographic Support (FCS)

5.4.2.1 Cryptographic Key Management (FCS_CKM)

FCS_CKM.1 Cryptographic Key Generation (Refinement)

FCS_CKM.1.1[12] 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, 3072 bits, 4096 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].*

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

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].*

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

[12] As per TD0921

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

that meets the following: *No Standard*.

5.4.2.2 Cryptographic Operation (FCS_COP)

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

FCS_COP.1.1/DataEncryption[13] The TSF shall perform encryption/decryption in accordance with a specified cryptographic algorithm AES used in [***CBC, GCM] and [CTR]*** mode and cryptographic key sizes ***[128 bits, 256 bits] and [192 bits]*** that meet the following: AES as specified in ISO 18033-3, ***[CBC as specified in ISO 10116, GCM as specified in ISO 19772], and [CTR as specified in ISO 10116]***.

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

FCS_COP.1.1/SigGen[14] 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: [2048 bits, 3072 bits, 4096 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.*

].

[13] As per MOD_VPNGW_v1.3

[14] As per TD0921

FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)

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

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

FCS_COP.1.1/KeyedHash The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm [*HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512, implicit*] and cryptographic key sizes [*160, 256 and 512 bits*] **and message digest sizes [*160, 256, 384 bits, 512*] bits** that meet the following: *ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*.

5.4.2.3 FCS_IPSEC_EXT.1 IPsec Protocol

FCS_IPSEC_EXT.1 IPsec Protocol[15]

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

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

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

FCS_IPSEC_EXT.1.4 The TSF shall implement the IPsec protocol ESP as defined by RFC 4303 using the cryptographic algorithms [***AES-CBC-128, AES-CBC-256 (specified in RFC 3602), AES-GCM-128, AES-GCM-256 (specified in RFC 4106)] and [AES-CBC-192 (specified in RFC 3602)***] together with a Secure Hash Algorithm (SHA)-based HMAC [*HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512*].

FCS_IPSEC_EXT.1.5[16] The TSF shall implement the protocol: [

- *IKEv2 as defined in RFC 7296 [with mandatory support for NAT traversal as specified in RFC 7296, Section 2.23], and [RFC 4868 for hash functions]*

].

FCS_IPSEC_EXT.1.6 The TSF shall ensure the encrypted payload in the [*IKEv2*] protocol uses the cryptographic algorithms [*AES-CBC-128, AES-CBC-192, AES-CBC-256 (specified in RFC 3602), AES-GCM-128, AES-GCM-256 (specified in RFC 5282)*].

FCS_IPSEC_EXT.1.7[17] The TSF shall ensure that [

- *IKEv2 SA lifetimes can be configured by a Security Administrator based on [*
    - *length of time, where the time values can be configured between [0.2 hours] and [24 hours]*

  ]

].

FCS_IPSEC_EXT.1.8[18] The TSF shall ensure that [

- *IKEv2 Child SA lifetimes can be configured by a Security Administrator based on [*
    - *number of bytes*

[15] As per MOD_VPNGW_v1.3
[16] As per TD 0824
[17] As per TD 0986
[18] As per TD 0986

- *length of time, where the time values can be configured between [0.2 hours] and [24 hours]*

]

].

FCS_IPSEC_EXT.1.9 The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange ("x" in g^x mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least [*224 (for DH Group 14), 256 (for DH Group 15), 256 (for DH Group 16), 256 (for DH Group 19), 384 (for DH Group 20), and 521 (for DH Group 21)*] bits.

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

- *at least 128 bits in size and at least half the output size of the negotiated pseudorandom function (PRF) hash*

].

FCS_IPSEC_EXT.1.11[19] The TSF shall ensure that IKE protocols implement DH Groups

- **20 (384-bit Random ECP) according to RFC 5114 and**

[

- *[14 (2048-bit MODP), 15 (3072-bit MODP), 16 (4096-bit MODP)] according to RFC 3526*
- *[19 (256-bit Random ECP), 21 (521-bit Random ECP)] according to RFC 5114.*

].

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

FCS_IPSEC_EXT.1.13[20] The TSF shall ensure that **[*IKEv2*]** protocols perform peer authentication using [*RSA, ECDSA*] that use X.509v3 certificates that conform to RFC 4945 and [Pre-shared Keys that conform to RFC 8784].

FCS_IPSEC_EXT.1.14 The TSF shall only establish a trusted channel if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following fields and types: **Distinguished Name (DN),** [*SAN: IP address, SAN: Fully Qualified Domain Name (FQDN), SAN: user FQDN*].

5.4.2.4 Random Bit Generation (Extended - FCS_RBG_EXT)

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 [*HMAC_DRBG [SHA-512]]*.

FCS_RBG_EXT.1.2[21] The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [*[1] software-based noise source, [1] platform-based noise source*] *with a minimum of* [*256 bits*] of entropy at least equal to [

[19] As per TD 0986
[20] As per TD 0824
[21] As per TD 0990

- *the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 "Security Strength Table for Hash Functions"*

].

5.4.3 Identification and Authentication (FIA)

5.4.3.1 User Identification and Authentication (Extended - FIA_UIA_EXT)

FIA_UIA_EXT.1 User Identification and Authentication

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

- Display the warning banner in accordance with FTA_TAB.1;
- [*ICMP Echo*].

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[22] The TSF shall provide the following remote authentication mechanisms [*SSH password, SSH public key, X.509 certificate*] 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.

5.4.3.2 Authentication using X.509 certificates (Extended – FIA_X509_EXT)

5.4.3.3 FIA_X509_EXT.1 X.509 Certificate Validation

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.*

[22] As per TD 0900

- o *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.

5.4.3.4 FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1[23] The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for **IPsec and [*no other protocols*]** and [*no additional uses*].

FIA_X509_EXT.2.2[24] When the TSF cannot establish a connection to determine the validity of a certificate, the TSF shall [*allow the Administrator to choose whether to accept the certificate in these cases*].

5.4.3.5 FIA_X509_EXT.3 X.509 Certificate Requests

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.

5.4.4 Security Management (FMT)

5.4.4.1 Management of functions in TSF (FMT_MOF)

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*.

5.4.4.2 Management of TSF Data (FMT_MTD)

FMT_MTD.1/CoreData Management of TSF Data

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

FMT_MTD.1/CryptoKeys Management of TSF Data

FMT_MTD.1.1/CryptoKeys[25] The TSF shall restrict the ability to [[*manage*]] the [cryptographic keys **and certificates used for VPN operation**] to [*Security Administrators*].

5.4.5 Specification of Management Functions (FMT_SMF)

FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1[26] The TSF shall be capable of performing the following management functions:

- *Ability to administer the TOE remotely;*

[23] As per MOD_VPNGW_v1.3
[24] As per MOD_VPNGW_v1.3
[25] As per MOD_VPNGW_v1.3
[26] As per TD0880

- *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 configure local audit behaviour (e.g. changes to storage location for audit; changes to behaviour when local audit storage space is full; changes to local audit storage size);*
  - *Ability to modify the behaviour of the transmission of audit data to an external IT entity;*
  - *Ability to manage cryptographic keys;*
  - *Ability to configure the cryptographic functionality;*
  - *Ability to configure thresholds for SSH rekeying;*
  - *Ability to configure the lifetime for IPsec SAs;*
  - *Ability to re-enable an Administrator account;*
  - *Ability to set the time which is used for time-stamps;*
  - *Ability to configure NTP;*
  - *Ability to configure the reference identifier for the peer;*
  - *Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;*
  - *Ability to administer the TOE locally;*
  - *Ability to configure the local session inactivity time before session termination or locking;*
  - *Ability to configure the authentication failure parameters for FIA_AFL.1;*
  - *Ability to manage the trusted public keys database].*

5.4.5.1 Security management roles (FMT_SMR)

5.4.5.2 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.

5.4.6 Protection of the TSF (FPT)

5.4.6.1 Protection of the TSF Data (Extended - FPT_SKP_EXT)

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

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

5.4.6.2 Time stamps (Extended - FPT_STM_EXT)

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, synchronise time with an NTP server*].

5.4.6.3 TSF Testing (Extended - FPT_TST_EXT)

FPT_TST_EXT.1 TSF Testing (Extended) [27]

FPT_TST_EXT.1.1[28] The TSF shall run a suite of the following self-tests:

- *During initial start-up (on power on) to verify the integrity of the TOE firmware and software;*
- *Prior to providing any cryptographic service and [at no other time] to verify correct operation of cryptographic implementation necessary to fulfil the TSF;*
- *[start-up] self-tests [Crypto integrity test]. to demonstrate the correct operation of the TSF:* ***noise source health tests.***

FPT_TST_EXT.1.2 The TSF shall respond to [all failures] *by [[rebooting the TOE]]*.

Trusted Update (FPT_TUD_EXT)

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[29,30] The TSF shall provide means to authenticate firmware/software updates to the TOE using a digital signature mechanism and [*no other mechanism*] prior to installing those updates.

5.4.7 TOE Access (FTA)

5.4.7.1 Session Locking and Termination (FTA_SSL)

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*.

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.

5.4.7.2 TOE Access Banners (FTA_TAB)

FTA_TAB.1 Default TOE Access Banners (Refinement)

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

[27] As per TD 0824
[28] As per MOD_VPNGW_v1.3
[29] As per MOD_VPNGW_v1.3
[30] As per TD 0824

5.4.8 Trusted Path/Channels (FTP)

5.4.8.1 Trusted Channel (FTP_ITC)

FTP_ITC.1 Inter-TSF Trusted Channel (Refinement)

FTP_ITC.1.1 The TSF shall **be capable of using [IPsec, SSH] to** provide a **trusted** communication channel between itself and ~~another trusted IT product~~ **authorized IT entities supporting the following capabilities: audit server, [[*node configured in Multinode HA Mode*]]** 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 [Forwarding of audit records to an external audit server, communication between another host configured in Multinode HA Mode].

5.4.8.2 Trusted Path (FTP_TRP)

FTP_TRP.1/Admin Trusted Path (Refinement)

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

5.5 Selection-Based Requirements Drawn from the Base-PP

5.5.1 Cryptographic Support (FCS)

5.5.1.1 Cryptographic Protocols (Extended – FCS_NTP_EXT)

FCS_NTP_EXT.1 NTP Protocol

FCS_NTP_EXT.1.1 The TSF shall use only the following NTP version(s) [*NTP v3 (RFC 1305), NTP v4 (RFC 5905)*].

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

- *Authentication using [SHA256] as the message digest algorithm(s);*

].

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

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

5.5.2 Identification and Authentication (FIA)

5.5.2.1 Authentication Failure Handling (FIA_AFL)

FIA_AFL.1 Authentication Failure Handling (Refinement)

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

FIA_AFL.1.2 When the defined number of unsuccessful authentication attempts has been met, the TSF shall [*prevent the offending Administrator from successfully establishing a remote session using any authentication method that involves a password until [unlocking the account from console] is taken by an Administrator; 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*].

5.5.2.2 Protected Authentication Feedback (FIA_UAU)

FIA_UAU.7 Protected Authentication Feedback (Refinement)

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**.

5.5.2.3 Password Management (Extended – FIA_PMG_EXT)

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: [*"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [all other standard ASCII, extended ASCII and Unicode characters]*];
b. Minimum password length shall be configurable to between [10] and [20] characters.

5.5.3 Protection of the TSF (FPT)

5.5.3.1 Protection of Administrator Passwords (Extended – FPT_APW_EXT)

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.

5.5.4 Security Management (FMT)

5.5.4.1 Management of functions in TSF (FMT_MOF)

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*.

FMT_MOF.1/Functions Management of Security Functions Behaviour

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

5.5.5 TOE Access (FTA)

5.5.5.1 TSF-initiated Session Locking (Extended – FTA_SSL_EXT)

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.

5.6 Optional Requirements Drawn from the Base-PP

The TOE does not implement any optional security functional requirements drawn from the Base-PP. Therefore, none is claimed in the ST.

5.7 Mandatory Security Functional Requirements Drawn from the Functional Package

FCS_SSH_EXT.1 SSH Protocol

FCS_SSH_EXT.1.1 The TOE shall implement SSH acting as a [*client, server*] in accordance with that complies with RFCs 4251, 4252, 4253, 4254, [5647, *4344, 5656, 6668, 8308, 8268, 8332*] and [*no other standard*].

FCS_SSH_EXT.1.2 The TSF shall ensure that the SSH protocol implementation supports the following authentication methods: [

- *"password" (RFC 4252),*
- *"publickey" (RFC 4252): [*
    - *rsa-sha2-256 (RFC 8332),*
    - *rsa-sha2-512 (RFC 8332),*
    - *ecdsa-sha2-nistp256 (RFC 5656),*
    - *ecdsa-sha2-nistp384 (RFC 5656),*
    - *ecdsa-sha2-nistp521 (RFC 5656)*

  ]

] and no other methods.

FCS_SSH_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [262144 bytes] in an SSH transport connection are dropped.

FCS_SSH_EXT.1.4 The TSF shall protect data in transit from unauthorised disclosure using the following mechanisms: [

- *aes128-ctr (RFC 4344),*
- *aes256-ctr (RFC 4344),*

- *aes128-cbc (RFC 4253),*
- *aes256-cbc (RFC 4253,)*
- *aes128-gcm@openssh.com (RFC 5647),*
- *aes256-gcm@openssh.com (RFC 5647)*

] 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-group16-sha512 (RFC 8268),*
- *diffie-hellman-group18-sha512 (RFC 8268),*
- *ecdh-sha2-nistp256 (RFC 5656),*
- *ecdh-sha2-nistp384 (RFC 5656),*
- *ecdh-sha2-nistp521 (RFC 5656)*

] and no other mechanisms.

FCS_SSH_EXT.1.7 The TSF shall use SSH KDF as defined in [

- *RFC 4253 (Section 7.2),*
- *RFC 5656 (Section 4)*

] 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.

5.8 Selection-Based Security Functional Requirements Drawn from the Functional Package

FCS_SSHC_EXT.1 SSH Protocol - Client

FCS_SSHC_EXT.1.1 The TSF shall authenticate its peer (SSH server) using: [

- *using a local database by associating each host name with a public key corresponding to the following list: [*
    - *rsa-sha2-256 (RFC 8332),*
    - *rsa-sha2-512 (RFC 8332),*

   - *ecdsa-sha2-nistp256 (RFC 5656),*
   - *ecdsa-sha2-nistp384 (RFC 5656),*
   - *ecdsa-sha2-nistp521 (RFC 5656)*

  ]

] as described in RFC 4251 section 4.1.

FCS_SSHS_EXT.1 SSH Protocol - Server

FCS_SSHS_EXT.1.1 The TSF shall authenticate itself to its peer (SSH Client) using: [

- *rsa-sha2-256 (RFC 8332),*
- *rsa-sha2-512 (RFC 8332),*
- *ecdsa-sha2-nistp256 (RFC 5656),*
- *ecdsa-sha2-nistp384 (RFC 5656),*
- *ecdsa-sha2-nistp521 (RFC 5656)*

].

5.9 Optional Security Functional Requirements Drawn from the Functional Package

The Functional Package does not define any optional requirements. Therefore, none are included in the ST.

5.10 Mandatory Security Functional Requirements Drawn from MOD_IPS_V1.0

5.10.1 Security Audit (FAU)

5.10.1.1 FAU_GEN.1 Audit data generation (Refinement)

FAU_GEN.1/IPS Audit Data Generation (IPS)

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

a) Start-up and shut-down of the **IPS** functions;
b) All **IPS** auditable events for the [*not specified*] level of audit; and
c) [*All dissimilar IPS events;*
d) *All dissimilar IPS reactions;*
e) *Totals of similar events occurring within a specified time period;*
f) *Totals of similar reactions occurring within a specified time period;*
g) *The events in the IPS Events table.*
h) [*no other auditable events*]][31].

FAU_GEN.1.2/IPS The TSF shall record within each **IPS auditable event** record at least the following information:

a) Date and time of the event, type of event **and/or reaction**, ~~subject identity, and the outcome (success or failure) of the event;~~ and
b) For each **IPS** audit**able** event type, based on the auditable event definitions of the functional components included in the PP~~/ST~~, *[information specified in column three of the IPS Events table*].

[31] As per TD 0595

Table 28 IPS Events

Requirement	Auditable Events	Additional Audit Record Contents
Mandatory Requirements		
FMT_SMF.1/IPS	Modification of an IPS policy element.	Identifier or name of the modified IPS policy element (e.g. which signature, baseline, or known-good/known-bad list was modified).
IPS_ABD_EXT.1	Inspected traffic matches an anomaly-based IPS policy	Source and destination IP addresses.
		The content of the header fields that were determined to match the policy.
		TOE interface that received the packet.
		Aspect of the anomaly-based IPS policy rule that triggered the event (e.g. throughout, time of day, frequency, etc.)
		Network-based action by the TOE (e.g. allowed, blocked, sent reset to source IP, sent blocking notification to firewall).
IPS_IPB_EXT.1	Inspected traffic matches a list of known-good or known-bad addresses applied to an IPS policy.	Source and destination IP addresses (and, if applicable, indication of whether the source and/or destination address matched the list).
		TOE Interface that received the packet.
		Network-based action by the TOE (e.g. allowed, blocked, sent reset).
IPS_NTA_EXT.1	Modification of which IPS policies are active on a TOE interface. Enabling/disabling a TOE interface with IPS policies applied. Modification of which mode(s) is/are active on a TOE interface.	Identification of the TOE Interface
		The IPS policy and the interface mode (if applicable).
IPS_SBD_EXT.1	Inspected traffic matches a signature-based IPS rule with logging enabled.	Name or identifier of the matched signature.
		Source and destination IP addresses.
		The content of the header fields that were determined to match the signature.
		TOE interface that received the packet.
		Network-based action by the TOE (e.g. allowed, blocked, sent reset).
Selection-Based Requirements		
None.		
Optional Requirements		
None.		

5.10.2 Security Management (FMT)

5.10.2.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1/IPS Specification of Management Functions (IPS)

FMT_SMF.1.1/IPS The TSF shall be capable of performing the following management functions [

- *Enable, disable signatures applied to sensor interfaces, and determine the behavior of IPS functionality*
- *Modify these parameters that define the network traffic to be collected and analyzed:*
    - *Source IP addresses (host address and network address)*
    - *Destination IP addresses (host address and network address)*
    - *Source port (TCP and UDP)*
    - *Destination port (TCP and UDP)*
    - *Protocol (Ipv4 and Ipv6)*
    - *ICMP type and code*
- *Update (import) signatures*
- *Create custom signatures*
- *Configure anomaly detection*
- *Enable and disable actions to be taken when signature or anomaly matches are detected*
- *Modify thresholds that trigger IPS reactions*
- *Modify the duration of traffic blocking actions*
- *Modify the known-good and known-bad lists (of IP addresses or address ranges)*
- *Configure the known-good and known-bad lists to override signature-based IPS policies]*

5.10.3 Intrusion Prevention (IPS)

5.10.3.1 IPS_ABD_EXT.1 Anomaly-Based IPS Functionality

IPS_ABD_EXT.1 Anomaly-Based IPS

IPS_ABD_EXT.1.1 The TSF shall support the definition of [*<u>anomaly ('unexpected') traffic patterns</u>*] including the specification of [

- *throughput ([<u>bits per second</u>]);*
- *time of day;*
- *frequency;*
- *thresholds;*

]

and the following network protocol fields:

- *[[<u>Ipv4: source address; destination address, Ipv6: source address; destination address, TCP: source port; destination port, UDP: source port; destination port</u>]]*

IPS_ABD_EXT.1.2 The TSF shall support the definition of anomaly activity through [*<u>manual configuration by administrators</u>*].

IPS_ABD_EXT.1.3 The TSF shall allow the following operations to be associated with anomaly-based IPS policies:

- In any mode, for any sensor interface: [
    - *<u>allow the traffic flow</u>*
    - *<u>send a TCP reset to the source address of the offending traffic</u>*
    - *<u>send a TCP reset to the destination address of the offending traffic</u>]*

- In inline mode:
  - *[allow the traffic flow*
  - *block/drop the traffic flow*
  - *and [no other actions]]*.

5.10.3.2 IPS_IPB_EXT.1 IP Blocking

IPS_IPB_EXT.1 IP Blocking

IPS_IPB_EXT.1.1 The TSF shall support configuration and implementation of known-good and known-bad lists of [*source, destination*] IP addresses and [*no additional address types*].

IPS_IPB_EXT.1.2 The TSF shall allow [Security Administrators] to configure the following IPS policy elements: [*known-good list rules, known-bad list rules, IP addresses*].

5.10.3.3 IPS_NTA_EXT.1 Network Traffic Analysis

IPS_NTA_EXT.1 Network Traffic Analysis

IPS_NTA_EXT.1.1 The TSF shall perform analysis of IP-based network traffic forwarded to the TOE's sensor interfaces, and detect violations of administratively-defined IPS policies.

IPS_NTA_EXT.1.2[32] The TSF shall process (be capable of inspecting) the following network traffic protocols: [

- *Internet Protocol version 4 (IPv4), RFC 791*
- *Internet Protocol version 6 (IPv6), RFC 8200*
- *Internet control message protocol version 4 (ICMPv4), RFC 792*
- *Internet control message protocol version 6 (ICMPv6), RFC 2463*
- *Transmission Control Protocol (TCP), RFC 793*
- *User Data Protocol (UDP), RFC 768].*

IPS_NTA_EXT.1.3 The TSF shall allow the signatures to be assigned to sensor interfaces configured for promiscuous mode, and to interfaces configured for inline mode, and support designation of one or more interfaces as 'management' for communication between the TOE and external entities without simultaneously being sensor interfaces.

- *Promiscuous (listen-only) mode: [none];*
- *Inline (data pass-through) mode: [Ethernet interfaces];*
- *Management mode: [Ethernet interfaces, out-of-band management Ethernet interfaces];*
- *[*
  - *[Session-reset-capable interfaces: [Ethernet interfaces]].*

]

5.10.3.4 IPS_SBD_EXT.1 Signature-Based IPS Functionality

IPS_SBD_EXT.1 Signature-Based IPS Functionality

IPS_SBD_EXT.1.1 The TSF shall support inspection of packet header contents and be able to inspect at least the following header fields: [

- *IPv4: version; header length; packet length; ID; IP flags; fragment offset; time to live (TTL); protocol; header checksum; source address; destination address; IP options; and [no other field]*
- *IPv6: version; payload length; next header; hop limit; source address; destination address; routing header; and [traffic class, flow label].*
- *ICMP: type; code; header checksum; and [[rest of header (varies based on the ICMP type and code]].*

[32] As per TD 0902

- *ICMPv6: type; code; and header checksum.*
- *TCP: source port; destination port; sequence number; acknowledgement number; offset; reserved; TCP flags; window; checksum; urgent pointer; and TCP options.*
- *UDP: source port; destination port; length; and UDP checksum].*

IPS_SBD_EXT.1.2 The TSF shall support inspection of packet payload data and be able to inspect at least the following data elements to perform string-based pattern-matching:

- *ICMPv4 data: characters beyond the first 4 bytes of the ICMP header.*
- *ICMPv6 data: characters beyond the first 4 bytes of the ICMP header.*
- *TCP data (characters beyond the 20 byte TCP header), with support for detection of:*
    - *i) FTP (file transfer) commands: help, noop, stat, syst, user, abort, acct, allo, appe, cdup, cwd, dele, list, mkd, mode, nlst, pass, pasv, port, pass, quit, rein, rest, retr, rmd, rnfr, rnto, site, smnt, stor, stou, stru, and type.*
    - *ii) HTTP (web) commands and content: commands including GET and POST, and administrator-defined strings to match URLs/URIs, and web page content.*
    - *iii) SMTP (email) states: start state, SMTP commands state, mail header state, mail body state, abort state.*
    - *iv) [<u>no other types of TCP payload inspection</u>];*
- *UDP data: characters beyond the first 8 bytes of the UDP header;*
- *[<u>no other types of TCP payload inspection</u>]*].

IPS_SBD_EXT.1.3 The TSF shall be able to detect the following header-based signatures (using fields identified in IPS_SBD_EXT.1.1) at IPS sensor interfaces: [

a) IP Attacks
   i. IP Fragments Overlap (Teardrop attack, Bonk attack, or Boink attack)
   ii. IP source address equal to the IP destination (Land attack)
b) ICMP Attacks
   i. Fragmented ICMP Traffic (e.g. Nuke attack)
   ii. Large ICMP Traffic (Ping of Death attack)
c) TCP Attacks
   i. TCP NULL flags
   ii. TCP SYN+FIN flags
   iii. TCP FIN only flags
   iv. TCP SYN+RST flags
d) UDP Attacks
   i. UDP Bomb Attack
   ii. UDP Chargen DoS Attack].

IPS_SBD_EXT.1.4 The TSF shall be able to detect all the following traffic-pattern detection signatures, and to have these signatures applied to IPS sensor interfaces: [

a) Flooding a host (DoS attack)
   i. ICMP flooding (Smurf attack, and ping flood)
   ii. TCP flooding (e.g. SYN flood)
b) Flooding a network (DoS attack)
c) Protocol and port scanning
   i. IP protocol scanning
   ii. TCP port scanning
   iii. UDP port scanning
   iv. ICMP scanning].

IPS_SBD_EXT.1.5 The TSF shall allow the following operations to be associated with signature-based IPS policies:

- *In any mode, for any sensor interface: [*
    - *allow the traffic flow;*
    - *send a TCP reset to the source address of the offending traffic;*
    - *send a TCP reset to the destination address of the offending traffic;]*
- *In inline mode:*
    - *block/drop the traffic flow;*
    - *and [*
        - *allow all traffic flow;].*

IPS_SBD_EXT.1.6 The TSF shall support stream reassembly or equivalent to detect malicious payload even if it is split across multiple non-fragmented packets.

5.11 Selection-Based Security Functional Requirements Drawn from MOD_IPS_V1.0

There are no selection-based security functional requirements defined in MOD_IPS_V1.0. Therefore, none are included in the ST.

5.12 Optional Security Functional Requirements Drawn from MOD_IPS_V1.0

The TOE does not implement any optional security functional requirements drawn from MOD_IPS_V1.0. Therefore, none is claimed in the ST.

5.13 Mandatory Security Functional Requirements Drawn from MOD_CPP_FW_V1.4E

5.13.1 User Data Protection (FDP)

5.13.1.1 FDP_RIP.2 Full Residual Information Protection

FDP_RIP.2 Full Residual Information Protection

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

5.13.2 Firewall (FFW)

5.13.2.1 FFW_RUL_EXT.1 Stateful Traffic Filtering

FFW_RUL_EXT.1 Stateful Traffic Filtering

FFW_RUL_EXT.1.1 The TSF shall perform stateful traffic filtering on network packets processed by the TOE.

FFW_RUL_EXT.1.2 The TSF shall allow the definition of stateful traffic filtering rules using the following network protocol fields:

- *ICMPv4*
    - *Type*
    - *Code*
- *ICMPv6*
    - *Type*
    - *Code*
- *IPv4*
    - *Source address*
    - *Destination Address*

  - *Transport Layer Protocol*
- *IPv6*
  - *Source address*
  - *Destination Address*
  - *Transport Layer Protocol*
  - *[no other field]*
- *TCP*
  - *Source Port*
  - *Destination Port*
- *UDP*
  - *Source Port*
  - *Destination Port*

and distinct interface.

FFW_RUL_EXT.1.3 The TSF shall allow the following operations to be associated with stateful traffic filtering rules: permit or drop with the capability to log the operation.

FFW_RUL_EXT.1.4 The TSF shall allow the stateful traffic filtering rules to be assigned to each distinct network interface.

FFW_RUL_EXT.1.5 The TSF shall:

a) accept a network packet without further processing of stateful traffic filtering rules if it matches an allowed established session for the following protocols: TCP, UDP, [*ICMP*] based on the following *network packet attributes:*
   1. *TCP: source and destination addresses, source and destination ports, sequence number, Flags;*
   2. *UDP: source and destination addresses, source and destination ports;*
   3. *['ICMP: source and destination addresses, type, [code]'].*

b) Remove existing traffic flows from the set of established traffic flows based on the following: [*session inactivity timeout, completion of the expected information flow*].

FFW_RUL_EXT.1.6 The TSF shall enforce the following default stateful traffic filtering rules on all network traffic:

a) The TSF shall drop and be capable of [logging] packets which are invalid fragments;
b) The TSF shall drop and be capable of [logging] fragmented packets which cannot be re-assembled completely;
c) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a broadcast network;
d) The TSF shall drop and be capable of logging packets where the source address of the network packet is defined as being on a multicast network;
e) The TSF shall drop and be capable of logging network packets where the source address of the network packet is defined as being a loopback address;
f) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address "reserved for future use" (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4;
g) The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is defined as an "unspecified address" or an address "reserved for future definition and use" (i.e. unicast addresses not in this address range: 2000::/3) as specified in RFC 3513 for IPv6;

h) *The TSF shall drop and be capable of logging network packets with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified; and*

i) *[no other rules].*

FFW_RUL_EXT.1.7 The TSF shall be capable of dropping and logging according to the following rules:

a) *The TSF shall drop and be capable of logging network packets where the source address of the network packet is equal to the address of the network interface where the network packet was received;*

b) *The TSF shall drop and be capable of logging network packets where the source or destination address of the network packet is a link-local address;*

c) *The TSF shall drop and be capable of logging network packets where the source address of the network packet does not belong to the networks associated with the network interface where the network packet was received.*

FFW_RUL_EXT.1.8 The TSF shall process the applicable stateful traffic filtering rules in an administratively defined order.

FFW_RUL_EXT.1.9 The TSF shall deny packet flow if a matching rule is not identified.

FFW_RUL_EXT.1.10 The TSF shall be capable of limiting an administratively defined number of half-open TCP connections. *In the event that the configured limit is reached, new connection attempts shall be dropped and the drop event shall be [logged].*

5.13.3 Security Management (FMT)

5.13.3.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1/FFW Specification of Management Functions

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

- *Ability to configure firewall rules;*

5.14 Selection-Based Security Functional Requirements Drawn from MOD_CPP_FW_V1.4E

There are no selection-based Security Functional Requirements defined in MOD_CPP_FW_V1.4. Therefore, none are included in the ST.

5.15 Optional Security Functional Requirements Drawn from MOD_CPP_FW_V1.4

The TOE does not implement any optional security functional requirements drawn from MOD_CPP_FW_V1.4. Therefore, none is claimed in the ST.

5.16 Mandatory Security Functional Requirements Drawn from MOD_VPNGW_v1.3

5.16.1 Security Audit (FAU)

5.16.1.1 FAU_GEN.1 Audit data generation (Refinement)

FAU_GEN.1/VPN Audit Data Generation (VPN Gateway)

FAU_GEN.1.1/VPN 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. Indication that TSF self-test was completed
c. Failure of self-test
d. All auditable events at [not specified] level of audit; and
e. [*auditable events defined int he Auditable Events for Mandatory Requirements table*].

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

Date and time of the event, type of event, subject identify (if applicable), and the outcome (success or failure) of the event; and
For each audit event type, based on the auditable event definitions of the functional components included in the PP/ST, *[additional information defined in the Auditable Events for Mandatory Requirements table for each auditable event, where applicable*].

Table 29 Auditable Events for Mandatory Requirements

Requirement	Auditable Events	Additional Audit Record Contents
Mandatory Requirements		
FAU_GEN.1/VPN	No events specified	N/A
FCS_CKM.1/IKE	No events specified	N/A
FMT_SMF.1/VPN	All administrative actions	No additional information.
FPF_RUL_EXT.1	Application of rules configured with "log" operation	• Source and destination addresses • Source and destination ports • Transport layer protocol
FPT_FLS.1/SelfTest	No events specified	N/A
FPT_TST_EXT.3	No events specified	N/A
FTP_ITC.1/VPN	Initiation of the trusted channel	No additional information.
	Termination of the trusted channel	No additional information.
	Failure of the trusted channel functions	Identification of the initiator and target of failed trusted channel establishment attempt
Selection-Based Requirements		
FIA_PSK_EXT.1	No events specified	N/A
FIA_PSK_EXT.2	No events specified	N/A
Optional Requirements		
None.		

5.16.2 Cryptographic Support (FCS)

5.16.2.1 FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1/IKE Cryptographic Key Generation (for IKE Peer Authentication)

FCS_CKM.1.1/IKE[33] The TSF shall generate **asymmetric** cryptographic keys **used for IKE peer authentication** in accordance with a specified cryptographic key generation algorithm: **[**

- ***FIPS PUB 186-5, "Digital Signature Standard (DSS)," Appendix A.1 for RSA schemes***
- ***FCS_FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2 for ECDSA schemes, and implementing "NIST curves" P-384 and [P-256, P-521]***

] and [

- ***FFC Schemes using "safe-prime" groups that meet the following: NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and [RFC 3526]***

] and specified cryptographic key sizes [*equivalent to, or greater than, a symmetric key strength of 112 bits*].

5.16.3 Security Management (FMT)

5.16.3.1 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1/VPN Specification of Management Functions

FMT_SMF1.1/VPN The TSF shall be capable of performing the following management functions [

- *Definition of packet filtering rules*
- *Association of packet filtering rules to network interfaces*
- *Ordering of packet filtering rules by priority*

[

- *No other capabilities*

]].

5.16.4 Packet Filtering (FPF)

5.16.4.1 FPF_RUL_EXT.1 Packet Filtering Rules

FPF_RUL_EXT.1 Packet Filtering Rules

FPF_RUL_EXT.1.1 The TSF shall perform packet filtering on network packets processed by the TOE.

FPF_RUL_EXT.1.2 The TSF shall allow the definition of packet filtering rules using the following network protocols and protocol fields: [

- *IPv4 (RFC 791)*
  - *source address*
  - *destination address*
  - *protocol*
- *IPv6 (RFC 8200)*
  - *source address*

[33] Asa per TD 0944

  - *destination address*
  - *next header (protocol)*
- *TCP (RFC 793)*
  - *source port*
  - *destination port*
- *UDP (RFC 768)*
  - *source port*
  - *destination port*

].

FPF_RUL_EXT.1.3 The TSF shall allow the following operations to be associated with packet filtering rules: permit and drop with capability to log the operation.

FPF_RUL_EXT.1.4 The TSF shall allow the packet filtering rules to be assigned to each distinct network interface.

FPF_RUL_EXT.1.5 The TSF shall process the applicable packet filtering rules (as determined in accordance with FPF_RUL_EXT.1.4) in the following order: [*Administrator-defined*]

FPF_RUL_EXT.1.6 The TSF shall drop traffic if a matching rule is not identified.

5.16.5 Protection of the TSF (FPT)

5.16.5.1 FPT_FLS.1 Failure with Preservation of Secure State

FPT_FLS.1/SelfTest Failure with Preservation of Secure State (Self-Test Failures)

FPT_FLS.1.1/SelfTest The TSF shall **shut down** when the following types of failures occur: [*failure of the power-on self-tests, failure of integrity check of the TSF executable image, failure of noise source health tests*].

5.16.5.2 FPT_TST_EXT.3 Self-Test with Defined Methods

FPT_TST_EXT.3 Self-Test with Defined Methods

FPT_TST_EXT.3.1 The TSF shall run a suite of the following self-tests [*[when loaded for execution]*] to demonstrate the correct operation of the TSF: [*integrity verification of stored executable code*].

FPT_TST_EXT.3.2 The TSF shall execute the self-testing through [*a TSF-provided cryptographic service specified in FCS_COP.1/**SigGen***].

5.16.6 Trusted Path/Channels (FTP)

5.16.6.1 FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1/VPN Inter-TSF Trusted Channel (VPN Communications)

FTP_ITC1.1/VPN The TSF shall **be capable of using IPsec to** provide a communication channel between itself and **authorized IT entities supporting VPN communications** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from **disclosure and detection of modification of the channel data**.

FTP_ITC1.2/VPN The TSF shall permit [*the authorized IT entities*] to initiate communication via the trusted channel.

FTP_ITC1.3/VPN The TSF shall initiate communication via the trusted channel for [*remote VPN gateways or peer*].

5.17 Selection-Based Security Functional Requirements Drawn from MOD_VPNGW_v1.3

5.17.1 Identification and Authentication (FIA)

5.17.1.1 FIA_PSK_EXT.1 Pre-Shared Key Composition

FIA_PSK_EXT.1 Pre-Shared Key Composition

FIA_PSK_EXT.1.1 The TSF shall be able to use pre-shared keys for IPsec and [*IKEv2*].

FIA_PSK_EXT.1.2 The TSF shall be able to accept the following as pre-shared keys: [*generated bit-based*] keys.

5.17.1.2 FIA_PSK_EXT.2 Generated Pre-Shared Keys

FIA_PSK_EXT.2 Generated Pre-Shared Keys

FIA_PSK_EXT.2.1 The TSF shall be able to [

- accept externally generated pre-shared keys.

]

5.18 Optional Security Functional Requirements Drawn from MOD_VPNGW_v1.3

The TOE does not implement any optional security functional requirements drawn from MOD_VPNGW_v1.3. Therefore, none is claimed in the ST.

5.19 Security Assurance Requirements

This section states the Security Assurance Requirements. The applicable Security Assurance Requirements are stated in Table 30.

Table 30 Security Assurance Requirements

Security Assurance Class	Security Assurance Components
Security Target (ASE)	Conformance claims (ASE_CCL.1) Extended components definition (ASE_ECD.1) ST Introduction (ASE_INT.1) Security objectives for the operational environment (ASE_OBJ.1) Stated security requirements (ASE_REQ.1) Security Problem Definition (ASE_SPD.1) TOE summary specification (ASE_TSS.1 - Refined)
Development (ADV)	Basic functional specification (ADV_FSP.1)
Guidance Documents (AGD)	Operational user guidance (AGD_OPE.1)

	Preparative procedures (AGD_PRE.1)
Life Cycle Support (ALC)	Labelling of the TOE (ALC_CMC.1) TOE CM Coverage (ALC_CMS.1) (Optional) Systematic Flaw Remediation (ALC_FLR.3)
Tests (ATE)	Independent testing - conformance (ATE_IND.1)
Vulnerability Assessment (AVA)	Vulnerability survey (AVA_VAN.1)

The refinement in ASE_TSS.1 as defined in CPP_ND_V3.0E is as follows:

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.**

5.20 Security Requirements Rationale

The Security Functional Requirements are drawn from the Base-PP, and the Functional Package and PP-Modules to which the ST claims exact conformance. The Security Functional Requirements include each mandatory requirement and each applicable optional and selection-based requirement. Only the operations allowed by the Base-PP, Functional Package and PP-Nodules are implemented. Therefore, the Security Functional Rationale of each is directly applicable to the ST and is not repeated.

The Security Assurance Requirements are only drawn from the Base-PP as required by the PP-Configuration. None. are added or removed. The optional component ALC_FLR.1 (Basic Flaw Remediation) is included as allowed by the Base-PP. Therefore, the Security Assurance Requirements Rationale of the Base-PP is directly applicable to the ST and is not repeated.

6 TOE Summary Specification

The TOE Summary Specification includes the description how the TOE fulfills the security functional requirements, and how the developer and the evaluator fulfill the security assurance requirements. Each is described in this section. Additional details on the cryptographic algorithms and protocols implemented in the TOE are also given.

6.1 Security Functional Requirements Drawn from the Base-PP

The fulfillment of the mandatory security functional requirements is given in Table 31.

Table 31 Fulfillment of the Mandatory Security Functional Requirements Drawn from the Base-PP

Security Functional Component	Fulfillment

FAU_GEN.1
 	
FAU_GEN.2
	The TOE generates and stores audit records for several events. The list of audit events per Security Functional Component is given in Table 27. Auditing is implemented using syslog.
	
	The detail of what events are to be recorded by syslog are determined by the logging level specified the level argument of the set system syslog CLI command. The audit knobs detailed in the security guidance must be configured.
	
	In the minimum, the TOE records the following information with each log entry:
	– Date and time of the event and/or reaction,
	– Type of event and/or reaction,
	– Subject identity (where applicable), and
	– The outcome (success or failure) of the event (if applicable).
	
	The subject identity is the username of the human user of the TOE or the IP Address of the peer entity attempting to connect to the TOE. Cryptographic keys are identified by the following detail when generated, imported, changed, or deleted:
	– SSH session keys: Key reference provided by process id, including the following keys:
	– SSH keys generated for outbound trusted channel to external syslog server.
	– SSH keys imported for outbound trusted channel to external syslog server.
	– SSH key configured for SSH public key authentication: The hash of the public key used for authentication.
	
	For SSH (ephemeral) session keys the PID is used as the key reference to relate the key generation and key destruction. The key destruction event is recorded as a session disconnect event. For example, key generation and key destruction events for a single SSH session key would be reflected by records such as the following:
	
	`Sep 27 15:09:36 yeti sshd[6529]: Accepted publickey for root from 10.163.18.165 port 45336 ssh2: RSA SHA256:l1vri77TPQ4VaupE2NMYiUXPnGkqBWIgD5vW0OuglGI`
	
	…
	
	`Sep 27 15:09:40 yeti sshd[6529]: Received disconnect from 10.163.18.165 port 45336:11: disconnected by user`
	
	`Sep 27 15:09:40 yeti sshd[6529]: Disconnected from 10.163.18.165 port 45336`

	SSH keys generated for outbound trusted channels are identified in the audit record by the public key filename and fingerprint. For example: `Sep 27 23:36:49 yeti ssh-keygen [67873]: Generated SSH key file /root/.ssh/id_rsa.pub with fingerprint SHA256:g+7lsR7x4lQb1JT8Q3scfb2sOl8lyccojGdmkmw4dwM` SSH keys imported for use in establishing outbound trusted channels are identified in the audit record by the hash of the key imported and the username of the user importing the key. The key is bound to the username. SSH keys used for trusted channels are not deleted by the management daemon when SSH is de-configured. SSH keys used for trusted channels are only deleted is when a `request vmhost zeroize` command is issued by the administrator. That commences zeroization of the entire appliance of which it is not possible to store an audit record.
FAU_STG_EXT.1	Syslog included in the TOE can be configured to store the audit logs locally or to send them to one or more syslog log servers. Log entries are sent in real time via Netconf over SSH. Local audit logs are stored in /var/log/ in the filesystem of the TOE. Only a Security Administrator can read, delete, or archive log files. Managing the log files is through the CLI interface or through direct access to the filesystem. The log files are automatically deleted locally if the administrator-configurable limit on the storage volume is reached. The default maximum storage size is 1Gb but the administrator can modify the allocated storage size using the size argument on the set system syslog CLI command. The TOE uses an active log file supported by archive files. The default number of archive files is 10 but the administrator may configure the number to be between 1 and 1000. When the active log file reaches the maximum size, the TOE closes the file, compresses it, and names the compressed archive file 'logfile.0.gz'. The TOE then opens and writes to a new active log file. When the new active log file reaches the maximum size, 'logfile.0.gz' is renamed 'logfile.1.gz', and the active log file is closed, compressed, and renamed 'logfile.0.gz'. This is repeated so that the latest compressed logfile is always named 'logfile.0.gz'. When the maximum number of archive files is reached or when the size of the active file reaches the configured maximum size, the contents of the oldest archived file are deleted so the current active file can be archived. A 1Gb syslog file takes approximately 0.25Gb of storage when archived. Syslog files total size may reach the complete storage capacity allocated to the /var filesystem. The complete storage capacity is platform specific. When the filesystem size reaches 92% of the storage capacity, an event is generated but the event daemon process (being a privileged process) still can continue using the reserved storage blocks. This allows the syslog to continue storing events while the administrator frees the storage. If the administrator does not free the storage in time, the /var filesystem becomes exhausted. If the file system is exhausted, a final log entry "No space left on device" is generated and the logging is terminated. Other functions of the TOE shall continue when the audit log storage space is exhausted.

FCS_CKM.1	The TOE generates cryptographic keys using RSA Schemes, ECC Schemes, and FFC Schemes: 1. RSA schemes are used to generate cryptographic key sizes of 2048-bit or greater. The keys are generated in accordance with FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.1. 2. ECC schemes are used to generate cryptographic keys for use with NIST curves P-256, P-384, and P-521. The ECC Schemes are used in accordance with FIPS PUB 186-5, "Digital Signature Standard (DSS)", Appendix A.2. 3. FFC Schemes use 'safe-prime' groups are used for generating SSH session keys. The FFC Schemes are used in accordance with NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and RFC 3526. The TOE implements each "shall" and each "should" requirement from FIPS PUB 186-5 Appendix A.1 and A.2. It does not implement any of the "shall not" and "should not" requirements. The key generation methods used by the TOE are detailed and the CAVP references given in Sect. 6.7.2.
FCS_CKM.2	The TOE implements key establishment using Elliptic curve-based key establishment schemes and FFC-based key establishment schemes. 1. Elliptic curve-based key establishment schemes are used in accordance with NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography". 2. FFC Schemes using "safe-prime" groups are used in accordance with 'NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and using groups listed in RFC 3526. The key establishment, authentication, encryption and data integrity algorithms and methods used by the TOE are detailed and the CAVP references given in Sect. 6.7.2.
FCS_CKM.4	The TOE implements functions for secure erasure of cryptographic keys and Critical Security Parameters (CSK). The keys and CSPs stored in the volatile memory are typically erased by the TOE software at the termination of a session. Keys and Critical security parameters stored in the non-volatile memory are erased when the administrator decommissions the TOE. The details are given in Sect. 6.7.1.
FCS_COP.1/DataEncryption	The TOE implements the AES key sizes and modes of operation used for symmetric encryption and decryption as required by the PP. Each cryptographic algorithm used by the TOE is CAVP validated. The CAVP certificate references together with the details of the key sizes and modes of operation are given in Sect. 6.7.2.
FCS_COP.1/SigGen	The TOE implements asymmetric cryptography for digital signature computation and verification. Each cryptographic algorithm used by the TOE is CAVP validated. The asymmetric cryptographic details together with the CAVP certificate references are given in Sect. 6.7.2.
FCS_COP.1/Hash	The TOE implements cryptographic hash functions. Each cryptographic algorithm used by the TOE is CAVP validated. The CAVP certificate references are given in Sect. 6.7.2.

<table>
<tr><td></td><td>The hash functions are used by the TOE for the following purposes:
<table>
<tr><th></th><th>SHA-1</th><th>SHA-256</th><th>SHA-384</th><th>SHA-512</th></tr>
<tr><td>SSH Hashing</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>SSH HMAC</td><td></td><td>X</td><td></td><td>X</td></tr>
<tr><td>SSH RSA Key Agreement</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>SSH ECC Key Agreement</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>IPsec ESP HMAC</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>IKEv2 HMAC</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>RSA Signature generation and verification</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>ECDSA Signature generation and verification</td><td></td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>Password hashing</td><td></td><td>X</td><td></td><td>X</td></tr>
<tr><td>File system integrity self-tests</td><td>X</td><td>X</td><td></td><td></td></tr>
<tr><td>Firmware integrity self-test</td><td></td><td>X</td><td></td><td></td></tr>
</table></td></tr>
<tr><td>FCS_COP.1/KeyedHash</td><td>The HMAC algorithms used by the TOE are detailed in the following:
<table>
<tr><th></th><th>HMAC-SHA-256</th><th>HMAC-SHA-384</th><th>HMAC-SHA-512</th></tr>
<tr><td>Key length</td><td>256 bits</td><td>512 bits</td><td>512 bits</td></tr>
<tr><td>Hash function</td><td>SHA-256</td><td>SHA-384</td><td>SHA-512</td></tr>
<tr><td>Block size</td><td>512 bits</td><td>512 bits</td><td>1024 bits</td></tr>
<tr><td>Output size</td><td>256 bits</td><td>384 bits</td><td>512b its</td></tr>
</table></td></tr>
<tr><td>FCS_IPSEC_EXT.1</td><td> The TOE is conformant to RFC 4301 and implements IPsec in tunnel mode. IPsec is used to protect state sharing of the instances of the TOE in the Multinode HA Mode.
There is a single IKE daemon, which is used to negotiate all IPsec tunnels. There is also a single implementation of IPSec used for custom VPN communications implemented in the data plane. The cryptographic methods the TOE implements for IPsec are given in Table 40.
IKEv2 is implemented as defined in RFCs 5996 (with mandatory support for NAT traversal) and RFC 4868 for hash functions.
The TOE permits configuration of the IPsec lifetime exchanges for custom VPN tunnels in terms of length of time (180 seconds to 8 hours). In addition, the TOE permits configuration of the IPsec lifetime based on configuration of the IPsec lifetime in terms of number of (kilo)bytes (64 to 4294967294 kilo bytes).
For IKE, the TOE permits configuration of the lifetime exchanges in terms of length of time (180 seconds to 24 hours). IKEv2 SA lifetime can be configured by the administrator to a value between 0.2 and 24 hours. </td></tr>
</table>

	IKEv2 Child SA lifetimes can also be configured by a Security Administrator. The lifetime may be based on a number of bytes or a length of time. The length of time may be a value between 0.2 and 24 hours.
	
	The following CLI commands configure a lifetime of either seconds or kilobytes:
	
	`set security ipsec proposal <name> lifetime-seconds <seconds>`
	
	`set security ipsec proposal <name> lifetime-kilobytes <kb>`
	
	`set security ike proposal <name> lifetime-seconds <seconds>`
	
	The TOE supports Diffie-Hellman Groups 14, 15, 16, 19, 20 and 21. When the TOE receives an IKE proposal, it will select the first DH group that matches the acceptable DH groups configured in the TOE. The negotiation will fail if there is no match. Similarly, when the peer initiates the IKE protocol, the TOE will select the first match from the IKE proposal sent by the peer and the negotiation fails if no acceptable match is found.
	
	The TOE uses HMAC DRBG with SHA-512 for the generation of DH exponents and nonces in the IKE key exchange protocol of length 224 bits (for DH Group 14), 256 bits (for DH Groups 19FCS_CKM.) and 384 bits (for DH Group 20).
	
	The TOE supports both RSA and ECDSA for use with X.509v3 certificates that conform to RFC 4945 and pre-shared Keys for IPsec support.
	
	The TOE ensures that the strength of the symmetric algorithm (128, 192 or 256 bits) negotiated to protect the IKEv2 IKE_SA connection is greater than or equal to the strength of the symmetric algorithm negotiated to protect the IKEv2 CHILD_SA connection.

FCS_RBG_EXT.1	All random numbers used by the TOE are generated in accordance with the NIST Special Publication 800-90. The TOE uses the HMAC_DRBG with SHA2-512 implemented in the kernel libraries. The CAVP References of the DRBG are given in Table 40. The selected HMAC_DRBG algorithm is seeded from a software-based entropy source which contains in the minimum 256 bits of entropy. An Entropy Assessment Report for the RBG is produced in a separate document.
FIA_UIA_EXT.1	The TOE requires users to be successfully identified and authenticated prior to granting them access to the controlled functions. For remote access, the TOE requires a SSH connection which is established using password based, public key or X.509 certificate based authentication. For local access, the TOE requires password-based authentication. The only functions the TOE allows on behalf of users prior to successful identification and authentication are the negotiation of the cryptographic protocols required for a trusted path for user authentication, displaying of the access banner in the authentication window, and responding to ICMP Echo.
FIA_X509_EXT.1 FIA_X509_EXT.2 FIA_X509_EXT.3	The TOE uses X.509 certificates as defined in RFC 5280 and RFC 8603. To generate a Certificate Request, the administrator uses the CLI command request security pki generate-certificate-request and supplies the following values: – Certificate-id – The internal identifier string for this certificate

- Domain-name
- Email address
- IP address
- Subject (DC=<Domain component>, CN=<Common-Name>, OU=<Organizational-Unit-name>, O=<Organization-name>, SN=<Serial-Number>, L=<Locality>, ST=<state>, C=<Country>)
- Filename – The local file in which to store the certificate signing request

The following extendedKeyUsage rules for X.509 certificates are not supported by the TOE:

- Certificates used for trusted updates and executable code integrity verification with 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 with 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 with 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 with the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.

To validate certificates, the TOE extracts the subject, issuer, subjects public key, signature, basicConstraints and validity period fields. If any of those fields is not present, the validation fails. The issuer is looked up in the PKI database. If the issuer is not present, or if the issuer certificate does not have the CA:true flag in the basicConstraints section, the validation fails. The TOE verifies the validity of the signature. If the signature is not valid, the validation fails. It then confirms that the current date and time is within the valid time period specified in the certificate.

The TOE may be configured to perform a revocation check using CRL in accordance with RFC 5280 (Sect. 6.3) and RFC 8603 (Sect. 7). If the CRL fails to download, there are two possible outcomes: If the TOE is configured with the option to skip CRL checking on download failure enabled, then the certificate shall be considered as having passed the validation. If the TOE is configured with the option to skip CRL checking on download failure disabled, then the certificate is considered to have failed validation.

The TOE validates a certificate path by building a chain of (at least 3) certificates based upon issuer and subject linkage, validating each according the certificate validation procedure described above. If any certificate in the chain fails validation, the validation fails as a whole. A self-signed certificate is not required to be at the root of the certificate chain.

The TOE determines if a certificate is a CA certificate by requiring the CA:true flag to be present in the basicConstraints section.

The TOE requires that the configured IKE identity of the local and remote endpoints to match the contents of the certificate associated with a SA endpoint. The TOE permits the identity to be expressed as distinguished name, fully qualified domain name (FQDN), user FQDN or IP address. If either certificate does not validate, or the contents do not match the configured identity, then the SA will not be established.

The PKI daemon validates all X509 certificates received from VPN peers during the IKE negotiation. If the TSF cannot establish a connection to determine the validity of a certificate, the SA will not be established unless the administrator of the TOE has

	explicitly configured the TOE to disable the CRL check in case the connection cannot be established. For public key-based authentication of IPsec connections, the TOE validates the X.509 certificates by extracting the subject, issuer, signature, basicConstraints and validity period fields. If any of those fields is not present, the validation fails. The issuer is looked up in the PKI database. If the issuer CA is not present, or if the issuer certificate does not have the CA:true flag in the basicConstraints section, the validation fails. The TOE verifies the validity of the signature. If the signature is not valid, the validation fails. It then confirms that the current date and time is within the valid time period specified in the certificate. The TOE generates Certificate Request Messages as specified in RFC 2986 when validating certificates for IPsec connections. Common Name, Organization, Organizational Unit, Country and public key details are provided in the CSR. Junos OS validates the chain of certificates from the Root CA when the CA Certificate Response is received.
FMT_MOF.1/ManualUpdate	Administrators of the TOE may query the current version of the TOE firmware using the CLI command show version. if a new version of the TOE firmware is available, the administrators may initiate an update of the TOE firmware. The TOE does not allow partial updates. The administrator must upgrade to the entire new release. Updates are downloaded and applied manually. The TOE does not implement automatic updates.
FMT_MTD.1/CoreData	The TOE implements human user authentication using passwords. Each user is identified with a username and authenticated with a password. Access to the management functions of the TOE is only granted to successfully identified and authenticated users assigned to the role Security Administrator. The authentication function of the TOE ensures that inactive sessions are terminated, authentication failures are handled in a manner that prevents password guessing, passwords may not be accessed in the file system, and the passwords are not echoed on the terminal when a user is being authenticated. Any remote management session is protected with SSHv2. These measures jointly ensure that the access to the management functions is only granted to legitimate administrators, and that the non-administrative users are effectively prevented from accessing the management functions.
FMT_MTD.1/CryptoKeys	The TOE allows successfully authenticated administrators to perform the following management functions on the cryptographic keys: – Manage the threshold for SSH rekeying, – Generation of SSH keys, – All key management functions on IKE and IPsec, and – All management functions on the X.509 cerificates. The cryptographic keys used by the TOE and the mechanisms available for the administrator to erase them is detailed in Sect. 6.7.1.
FMT_SMF.1	The TOE implements a Command Line Interface (CLI) which allows the administrators to manage the TOE. The CLI may be accessed locally from console or from a remote management station over a SSHv2 of IPsec connection. The entire CLI is accessible to all administrator, whether accessing the TOE locally or remotely. The TOE prevents access to the CLI by unauthenticated and unauthorized users.

	The TOE implements the following management functions fully detailed in the security guidance: – Configuring the access banner, – Configuring the session inactivity time before session termination, – Updating the TOE software, and verifying the update using digital signature capability prior to installation, – Starting and stopping services – Configuring the local audit behaviour, – Managing the cryptographic keys, – Configuring the thresholds for SSH rekeying, – Re-enabling a locked Administrator account, – Setting the time used for time-stamps, – Configure the reference identifier for the peer, – Configuring NTP, – Managing the trust store of the TOE, including designating X.509v3 certificates as trust anchors, and importing X.509 certificates to the trust store, – Configuring the authentication failure parameters, and – Managing the SSH key databases.
FMT_SMR.2	The only role maintained by the TOE is a Security Administrator. Only users accessing the TOE from user accounts assigned to the Security Administrator role are granted the right to administer the TOE. Each user account has attributes user identity (username), authentication data (password) and role (privilege) assigned to it. The role Security Administrator is associated with a login class “security-admin”. The security-admin logic class is assigned the necessary privileges which permit the users to perform all management functions of the TOE. Security Administrators may administer the TOE locally from system console or remotely over a SSHv2 trusted path using the SSHv2 protocol.
FPT_SKP_EXT.1	The CLI implemented by the TOE does not include commands for viewing the cryptographic keys. The TOE enforces kernel-level file access rights to the key containers. The access rights granted by the TOE limit access to the contents of cryptographic key containers only to the processes with cryptographic rights and to the shell users with root permission. As security administrators do not have root permission to the Junos OS, the measures restrict access to the contents of the key containers to authorized processes only.
FPT_STM_EXT.1	The TOE implements a clock based maintained by the OS Kernel. The time may be set by the administrator. The TOE also supports synchronization of the time with a NTP Server. The clock may be used for generating real-time time stamps and counters indicating the time from a specific event. The clock is used by the TOE to produce a time stamp for each audit record generated by the TOE, to implement inactivity timers for the administrative sessions, to implement the periods on which a user may not attempt re-authentication after a failed authentication attempt, and to implement protocol timers required for triggering re-keying or termination of a protocol session.
FPT_TST_EXT.1	The TOE runs the following set of self-tests when powered on to verify the correct operation of the TOE software:

	1. Power on test to determine that the boot-device responds and performs a memory size check to confirm the amount of available memory. 2. File integrity test to verify each mounted signed package and to assert that system files have not been tampered with. To test the integrity of the firmware, the SHA-1 fingerprints of the executables and other immutable files are regenerated and validated against the reference fingerprints contains in the manifest file. 3. Crypto integrity test to check the integrity of cryptographic keys and major CSPs. 4. Authentication error to verify that veriexec is enabled and operates as expected using /opt/sbin/kats/cannot-exec.real. 5. Kernel, libmd, OpenSSL, QuickSec, SSH tests to verify the output from known answer tests for the cryptographic algorithms. The TOE only executes legitimate binaries. Within the package containing the TOE software, each firmware image includes fingerprints of the executables and other immutable files. The TOE will not execute any binary without validating the registered fingerprint. This protects the TOE from unauthorized firmware which might compromise the integrity of the TOE. The self-tests ensure that only authorized executables are allowed to run and ensure the correct operation of the TOE. In case of a corrupt state or a failure in a self-test, the TOE will panic. The event will be logged, the TOE will cease processing network traffic and CLI commands, and restart. When the TOE restarts, the boot process shall not succeed without passing each self-test. This constitutes the automatic recovery and self-test behavior of the TOE
FPT_TUD_EXT.1	TOE does not allow partial updates. The administrator must upgrade to the entire new release. Updates are downloaded and applied manually. The TOE does not implement automatic updates. The installable firmware package containing an update to the TOE software has a digital signature attached. The digital signature is computed using ECDSA (P-256) with SHA-256 in the development environment of the TOE. The TOE checks the digital signature and only proceeds with the installation if the verification succeeds. The TOE maintains a set of fingerprints (i.e. SHA-1 digests) for executable files and other files which should be immutable. The manifest file is digitally signed using the package signing key in the development environment The signature is verified by the TOE. The fingerprint loader will only process a manifest for which it can verify the signature. Without a valid digital signature an executable will not be executed. When the command is issued to install an update, the manifest file for the update is verified and stored, and each executable/immutable file is verified before being executed. If any of the fingerprints in an update fails the verification, the upgrade will fail, and the TOE will use the last known verified image instead.
FTA_SSL.3	The administrator may configure the TOE to terminate each local and remote user session after a period of inactivity. The TOE implements a clock and generates an instance of a counter for each user to track the clock cycles since last activity. The count is reset each time the TOE detects activity on the user session. When the

	instance of a counter reaches the number of clock cycles equating to the configured period of inactivity, the user session is locked out. the Administrator may configure the inactivity period. The default value is 30 seconds. When a user is locked out, the TOE overwrites the display device and makes the current contents unreadable. The session is also terminated to prevent any further interaction with the TOE until a successful re-authentication.
FTA_SSL.4	Each user sessions, whether local or remote, can be terminated by the user. The user may log out of an existing local or remote session by issuing a logout command. When the command is issued, the user exits the session, and the TOE makes the current contents unreadable. No user activity can take place until a successful re-authentication.
FTA_TAB.1	The administrator may configure an access banner to be displayed at each local and remote authentication exchange. The banner may provide warnings against unauthorized access to the TOE and any other information that the administrator wishes to communicate.
FTP_ITC.1 FTP_TRP.1/Admin	The TOE implements trusted channels and trusted paths with SSHv2. SSHv2 may be used by a remote syslog server to establish a secure connection with the TOE at the network layer. The secure connection may be used by the TOE to forward audit records to the syslog server for storage and further processing. The TOE allows for remote administration of the TOE. The administrator may use a SSHv2 client of a remote management station to connect to the TOE. Upon successful authentication, the TOE establishes a SSHv2 session with the SSH client of the remote management station. That connection is used for securing all administrative commands and responses thereof. When configured in the Multinode HA Mode, the state of the TOE, when exchanged between two instances of the TOE, is protected with IPsec.

The fulfillment of the selection-based security functional requirements is given in Table 32

Table 32 Fulfillment of the Selection-Based Security Functional Requirements Drawn from the Base-PP

Security Functional Component	Fulfillment
FCS_NTP_EXT.1	The TOE supports synchronization of the time with an external NTP Server. Both NTPv3 and NTPv4 are supported. Timestamps from multicast and broadcast addresses are not accepted. the NTP servers may be configured by the administrator of the TOE. At least three NTP time sources are supported. Protection of the NTP timestamps is with SHA2-256.
FIA_AFL.1	The TOE implements a password-based authentication for local users and for remote users. The authentication is implemented using the hardened FreeBSD which is part of the TOE software. The TOE software implements the login() using the Pluggable Authentication Modules (PAM) Library calls. The password entered by the user is hashed, and the digest value is compared to the stored reference value. The success or failure of the comparison is returned to login(). PAM is used for authentication management, account management, session

	management, and password management. The login() primarily uses the session management and password management functions of PAM. The administrator may configure the retry-options to specify the action to be taken when a remote authentication fails. The retry-options are applied to each user of the TOE. Users are identified by a username. The retry-options configurable to the administrator include the following: 1. The back-off factor: The length of delay (configurable to a value between 5 and 10 seconds) after each failed attempt before a new authentication attempt may occur. 2. The back-off threshold: The increase of the delay for each subsequent failed authentication attempt. 3. The tries-before-disconnect: The maximum number of times (configurable to a value between 1 and 10) the administrator is allowed to attempt password-based authentication through SSH before the connection is disconnected. 4. The lockout-period: The time in minutes (configurable between 1 and 43,200 minutes) before the administrator may attempt to log in to the TOE after being locked out due to the number of failed login attempts. The above concern with remote access to the TOE. Even if an account is locked to disallow remote access, the administrator may attempt a local login from the console. This ensures that the TOE is always accessible. An Administrator accessing the TOE locally may also unlock a remote Administrator account.
FIA_PMG_EXT1	Authentication data for the human users accessing the TOE is a password. Passwords are case-sensitive, alphanumeric strings. A password must of the minimum length. The minimum length may be configured by the administrator to be between 10 characters and 20 characters. Passwords must be composed of any combination of upper- and lower-case letters, numbers, special characters and any other standard ASCII, extended ASCII and Unicode characters. The allowed special characters are "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")".
FIA_UAU.7	For password authentication, the TOE software calls function login() which interacts with a user to request a username and password. The username and the password read from the user are used to identify and authenticate the user. The username entered by the administrator at the username prompt is echoed to the screen. There is no visual or other information presented to the used when the password is entered. This ensures that any potential eavesdropped with a visual access to the terminal the administrator uses for authenticating the TOE gains no information about the length or the content of the password.
FMT_MOF.1/Services	The TOE implements a SSH Server to which the administrators may connect from a remote syslog server of from a remote management station. The administrator may also terminate the SSH session at any time. This allows the administrator to start and stop the trusted channels and trusted paths of the TOE.
FMT_MOF.1/Functions	The administrator may configure the TOE to transmit audit records to an external syslog server. The transmission is over a secure SSHv2 connection which is initiated by the syslog server. If configured, the audit records are sent to the syslog server in real time.

	The administrator may also configure the handling of audit data as detailed in FAU_GEN.1 and FAU_STG_EXT.1. The audit function is stopped, and a log entry indicating exhaustion of the file system generated when the file capacity is exhausted (see FAU_STG_EXT.1). The behavior is not configurable by the administrator.
FPT_APW_EXT.1	The passwords of the users are hashed when stored in the local password file. Hashing may be configured to be with SHA-256 or SHA-512. The CLI implements no functions for accessing the passwords directly. SHA-256 and SHA-512 are cryptographically secure. Even if gaining access to the hashed passwords, the has no practical means of recovering the password from the hash value. Authentication data for public key-based authentication is stored in a directory owned by the user. The directory typically has the same name as the user. The directory contains the files .ssh/authorized_keys and.ssh/authorized_keys2 which are used for SSH public key authentication. The CLI allows no direct access to the files and the authentication data may only be accessed through the CLI commands for managing the keys, or by the privileged processes implementing the SSH Server. They are not directly accessible to the administrators.
FTA_SSL_EXT.1	The administrator may configure the TOE to terminate each local and remote user session after a period of inactivity. The TOE implements a clock and generates an instance of a counter for each user to track the clock cycles since last activity. The count is reset each time the TOE detects activity on the user session. When the instance of a counter reaches the number of clock cycles equating to the configured period of inactivity, the user session is locked out. the Administrator may configure the inactivity period. The default value is 30 seconds. When a user is locked out, the TOE overwrites the display device and makes the current contents unreadable. The session is also terminated to prevent any further interaction with the TOE until a successful re-authentication.

There are no optional security functional requirements drawn from the Base-PP defined in the ST.

6.2 Security Functional Requirements Drawn from the Functional Package

The Functional Package defines only a single mandatory and two selection-based Security Functional Requirements. There are no optional requirements. The fulfillment of the security functional requirements drawn from the Functional Package is given in Table 33.

Table 33 Fulfillment of the Security Functional Requirements Drawn from the Functional Package

Security Functional Component	Fulfillment
FCS_SSH_EXT.1 FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	The TOE implements a SSH server for Trusted Channels between the TOE and a remote audit server, and a SSH Client for Trusted Paths between itself and remote administrators. SSH ensures that the communication over trusted

	channels and trusted paths is protected against unauthorized disclosure or modification. An identical set of supported ciphers is implemented for SSH Server and SSH Client. Secure connection to a secure, remote server is achieved by setting up an event trace monitor that sends event log messages by using NETCONF over SSH to the remote system event logging server. The remote audit server initiates the connection. SSHv2 ensures that the transmitted data cannot be disclosed or altered. The SSH Server also supports trusted paths using SSHv2 protocol to ensures the confidentiality and integrity of remote user sessions. Remote administrators may initiate secure communication to the TOE from the SSH client of the remote management station by initiating a SSH session with the TOE. Assured identification of the parties is assured by password-based and public key-based authentication. SSHv2 protocol ensures that the data transmitted over the session cannot be disclosed or altered by unauthorized parties. The SSH server and client are implemented in accordance with RFCs 4251, 4252, 4253, 4254, 4344, 5647, 5656, 8308 (Sect. 3.1), and 8268, 8332. The cryptographic algorithms used in the TOE implementation of SSH are CAVP validate. The CAVP certificate references together with the algorithm details are given in Sect. 6.7.2. Specifically, the following algorithms and standards are supported for the establishment of shared secrets: diffie-hellman-group16-sha512 (RFC 8268), diffie-hellman-group18-sha512 (RFC 8268), ecdh-sha2-nistp256 (RFC 5656), ecdh-sha2-nistp384 (RFC 5656), and ecdh-sha2-nistp521 (RFC 5656). To conform with RFC 4253, the TOE ensures that each packet in an SSH transport connection which is greater in size than 262 144 bytes is dropped.

6.3 Security Functional Requirements Drawn from MOD_IPS_V1.0

The fulfillment of the mandatory security functional requirements drawn from MOD_IPS_V1.0 is given in Table 34.

Table 34 Fulfillment of the Mandatory Security Functional Requirements Drawn from MOD_IPS_V1.0

Security Functional Component	Fulfillment
FAU_GEN.1/IPS	The TOE implements a universal audit function. Audit data processing is identical independently of the source of audit events. Auditing of the IPS function is performed with mechanisms identical to those described in FAU_GEN.1 The TOE generates the following IPS-specific audit records: – Start-up and shut-down of the IPS functions, – All dissimilar IPS events and reactions,

	– Totals of similar events and reactions occurring within a specified time period, – Modification of an IPS policy element, – Modification of which IPS policies are active on a TOE interface, – Enabling/disabling a TOE interface with IPS policies applied, – Modification of which mode(s) is/are active on a TOE interface, – Inspected traffic matches a list of known-good or known-bad addresses applied to an IPS policy, – Inspected traffic matches a signature-based IPS policy with logging enabled, and – Inspected traffic matches an anomaly-based IPS policy. IPS event log generation often happens in bursts and can generate a large volume of messages during an attack. To manage the volume of log messages, the TOE supports log suppression. Multiple instances of the same log occurring from the same or similar sessions over the same period of time. IPS log suppression is enabled by default and can be customized based on the following configurable attributes: – Source/destination addresses, – Number of log occurrences after which log suppression begins, – Maximum number of logs that log suppression can operate on, and – Time after which suppressed logs are reported. Suppressed logs are reported as single log entries containing the count of occurrences

FMT_SMF.1/IPS	All management functions of the TOE are accessible to the administrators through the CLI. In addition to the management functions identified under FMT_SMF.1, FMT_SMF.1/FFW, and FMT_SMF.1/VPN, the CLI also implements the following IPS-specific management functions:
	
	– Enable, disable signatures applied to sensor interfaces, and determine the behavior of IPS functionality,
	– Modify the parameters that define the network traffic to be collected and analyzed:
	  o Source IP addresses (host address and network address),
	  o Destination IP addresses (host address and network address),
	  o Source port (TCP and UDP),
	  o Destination port (TCP and UDP),
	  o Protocol (Ipv4 and Ipv6), and
	  o ICMP type and code.
	– Update (import) signatures,
	– Create custom signatures,
	– Configure anomaly detection,
	– Enable and disable actions to be taken when signature or anomaly matches are detected,
	– Modify thresholds that trigger IPS reactions,
	– Modify the duration of traffic blocking actions,

	– Modify the known-good and known-bad lists (of IP addresses or address ranges), and – Configure the known-good and known-bad lists to override signature-based IPS policies.
IPS_ABD_EXT.1 IPS_IBD_EXT.1 IPS_NTA_EXT.1 IPS_SBD_EXT.1	The Intrusion Detection and Prevention (IDP) policy allows selective enforcement of attack detection and prevention techniques on network traffic passing through an IDP-enabled device. Policy rules can be defined to match a section of traffic based on a zone, network, and application, and then trigger active or passive preventive actions on that traffic. An IDP policy the TOE enforces is made up of rule bases. Each rule base contains a set of rules that specify rule parameters, such as traffic match conditions, action, and logging requirements. IDP policies can be associated to firewall policies. IDP can be invoked on a firewall rule by rule basis for maximum granularity. Only firewall policies marked for IDP will be processed by the IDP engine. All other rules will only be processed by the firewall. Firewall Policies match Source Zone, Destination Zone, Source IP, Destination IP, Source Port, Destination Port, and Protocol. Interface and VLAN matching can be achieved if zones are used. Rules are organized into a firewall policy rulebase. Within IPS Policies, further matching for specific attacks is done on Source Zone, Destination Zone, Source IP, Destination IP, Source Port, Destination Port, and Protocol. Attack Actions are configurable on a rule-by-rule basis. Rules within policies are processed in an Administrator-defined order when network traffic flows through the TOE network interfaces. Once stateful firewall processing of packets has been performed by the Information Flow subsystem, if a firewall policy that has been marked for IDP processing is triggered, the packets are processed by the IPS subsystem as follows: – Fragmentation Processing. IP Fragments are reordered and reassembled. Duplicate, oversized, undersized, overlapping, incomplete and other invalid fragments are discarded. – Flow Module SSL Decryption. Sessions are checked for existing IP Actions. If none exist, new sessions are created. If a destination is marked for SSL decryption, a copy of the SSL traffic will be sent to the decryption engine. The original packet will be queued until inspection is complete. – Packet Serialization and TCP Reassembly. Packets are ordered and all TCP packets are reassembled into complete application messages. – Application ID. Pattern matching is performed on the traffic to determine which application the traffic is. The traffic will be inspected for Attacks, even if application cannot be determined. – Protocol Decoding. Protocol parsing and decoding is performed. Messages are deconstructed into application "contexts" which identify components of messages. Protocol Anomaly Detection is performed, along with AppDoS (if configured) by thresholds of these contexts.

	– Attack Signature Matching. Signatures are detected via deterministic finite automaton (DFA) pattern matching. – IDP Attack Actions. When an attack is detected, the corresponding policy configured action is executed. Possible actions include: o No Action o Drop packet o Drop connection o Close client (send an RST packet to the client) o Close server (sends an RST packet to the server) o Close client and server (sends an RST packet to both client and server) The TOE supports stateful signature-based attack detection defined as Attack Objects. Attack Objects use context-based matching to match regular expressions in specific locations where they occur. Attack Objects can be composed of multiple signatures and protocol anomalies, including logical expressions between signatures for compound matching. The TOE is capable of inspecting IPv4, IPv6, ICMPv4, ICMPv6, TCP and UPD traffic. Conformance to these RFCs is demonstrated by protocol compliance testing by the product QA team. The TOE can inspect all traffic passing through the TOE's Ethernet interfaces (inline mode). Ethernet interfaces can be assigned to Zones on which firewall and IDP policies are predicated. The TOE supports management through the console port, as well as through a dedicated Ethernet management port whose traffic is never processed for routing. Remote management of the TOE can also be performed via SSH or IPsec. The TOE supports the definition of known-good and known-bad lists of source and/or destination addresses at the firewall rule level. Address ranges can be defined by creating address book entries and attaching them to firewall policies. IPS signatures are articulated at different points along the traffic processing flow implemented in the TOE. Interfaces are grouped into zones. The TOE supports the definition of signatures at the zone level, also known as the screen level. TOE screen options secure a zone by inspecting, then allowing or denying, all connection attempts that require crossing an interface bound to that zone. Sanity checks on IPv4 and IPv6 aimed at detecting malformed packets are performed at the screen level. In addition to attack detection and prevention at the screen level, the TOE implements firewall and IDP policies at the inter-, intra-, and super-zone policy levels (super-zone here means in global policies, where no security zones are referenced). The TOE supports inspection of the following packet header information: – IPv4: Version; Header Length; Packet Length; ID; IP Flags; Fragment Offset; Time to Live (TTL); Protocol; Header Checksum; Source Address; Destination Address; and IP Options. – IPv6: Version; traffic class; flow label; payload length; next header; hop limit; source address; destination address; routing header; home address options.

- ICMPv4: Type; Code; Header Checksum; and Rest of Header (varies based on the ICMP type and code).
- ICMPv6: Type; Code; and Header Checksum.
- TCP: Source port; destination port; sequence number; acknowledgement number; offset; reserved; TCP flags; window; checksum; urgent pointer; and TCP options.
- UDP: Source port; destination port; length; and UDP checksum.

Signatures can be defined to match the any of above header-field values, using the command "`set security idp custom-attack`", along with the actions (allow/block), using the command "`set security idp idp-policy`", that the TOE will perform when a match is found in the processed packets. The matching criteria can be "equal", "greater-than", "less-than" or "not-equal".

The TOE also supports string-based pattern-matching inspection of packet payload data for the above protocols. For TCP payload inspection, the TOE provides pre-defined attack signatures to detect FTP commands, HTTP commands and content, and STMP states. Administrators can also define custom-attack signatures for these application layer protocols using the command "`set security idp custom-attack`".

The TOE can detect the following signatures using Junos predefined screen options:

Signature Name	TOE screen name
IP Fragments Overlap (Teardrop attack, Bonk attack, or Boink attack)	ip tear-drop
IP source address equal to the IP destination (Land attack)	tcp land
Fragmented ICMP Traffic (e.g. Nuke attack)	icmp fragment
Large ICMP Traffic (Ping of Death attack)	icmp ping-death
TCP Null flags	tcp tcp-no-flag
TCP SYN+FIN flags	tcp syn-fin
TCP FIN only flags	tcp fin-no-ack
UDP Bomb Attack	N/A (configured by default)
ICMP flooding (Smurf attack, and ping flood)	icmp flood
TCP flooding (e.g. SYN flood)	tcp syn-flood
IP protocol scanning	ip unknown-protocol
TCP port scanning	tcp port-scan
UDP Port scanning	udp port-scan
ICMP scanning	icmp ip-sweep

	The default action for the above screens is to drop the packets. To allow the packets through, the "alarm-without-drop" action can be defined using the command "set security screen ids-option". The TOE is also capable of detecting the following signatures: – CP SYN+RST flags, by defining an custom attack to match "protocol tcp tcp-flags rst" and "protocol tcp tcp-flags syn" ; – UDP Chargen DoS attack, by configuring a firewall policy to match the predefined "junos-chargen" with the desired allow/block reaction; – Flooding of a network (DoS attack), by the configuration of policers that allow establishing prioritization and bandwidth limits for different type of network traffic. The TOE allows administrators to define signatures for anomalous traffic in terms of throughput (bits per second), time of the day for defined source/destination address and source/destination port, frequency of traffic patterns and thresholds of traffic patterns. Anomaly signatures based on time of day characteristics are implemented by configuring schedulers using the Junos command 'set schedulers' and attaching them to firewall policies, which in turn specify the target traffic in terms of IP addresses and port numbers as well as the action to be perform on signature triggering (allow or block/drop traffic). Anomaly signatures based on throughput characteristics are implemented by configuring policers with a bandwidth limit and the desired signature action (discard or forward), using the Junos command 'set firewall policer', and attaching it to any interface with the Junos command 'set interfaces'. Traffic exceeding the specified throughput limit is dropped when the policer is configured to discard traffic. A policer can be applied to specific inbound or outbound IP packets in a Layer 3 traffic flow at a logical interface by using a stateless firewall filter. If an input firewall filter is configured on the same logical interface as a policer, the policer is executed first. If an output firewall filter is configured on the same logical interface as a policer, the firewall filter is executed first. Time bindings can be used to configure the time attributes for the time binding custom attack object. Time attributes control how the attack object identifies attacks that repeat for a certain number of times. Configuration of the scope and the count of an attack, a sequence of the same attacks over a period of time across sessions can be detected.. The maximum time interval between any two instances of a time binding custom attack can be configured. The maximum time interval between any two instances of a time binding attack is 60 seconds. The interval interval-value statement is introduced at the edit security idp custom-attack attack-name time-binding hierarchy to configure a custom time-binding.

	Attack threshold-based filtering is set using the burst-size-limit policer. For a single-rate two-color policer, the burst size is configured as a number of bytes. The burst size allows for short periods of traffic bursting (back-to-back traffic at average rates that exceed the configured bandwidth limit). Single-rate two-color policing uses the single token bucket algorithm to measure traffic-flow conformance to a two-color policer rate limit. Traffic at the interface which conforms to the bandwidth limit is categorized green. Traffic that exceeds the specified rate is also categorized as green if sufficient tokens remain in the single token bucket. Packets in a green flow are implicitly marked with low packet loss priority and then passed through the interface. Traffic which exceeds the specified rate when insufficient tokens remain in the single token bucket is categorized red. Depending on the configuration of the two-color policer, packets in a red traffic flow might be implicitly discarded; or the packets might be re-marked with a specified forwarding class, a specified PLP, or both, and then passed through the interface. The burst size extends the function of the bandwidth limit to allow bursts of traffic up to a limit based on the overall traffic load: – When a single-rate two-color policer is applied to the input or output traffic at an interface, the initial capacity for traffic bursting is equal to the number of bytes specified by this statement. – During periods of relatively low traffic, unused tokens accumulate in the bucket, but only up to the configured token bucket depth. Single-rate two-color policing allows bursts of traffic for short periods. Single-rate and two-rate three-color policing allows more sustained bursts of traffic. Hierarchical policing is a form of two-color policing which applies different policing actions based on whether the packets are classified for expedited forwarding or for a lower priority. A hierarchical policer is applied to ingress Layer 2 traffic to allows bursts of expedited forwarding traffic for short period and bursts of non- expedited forwarding traffic for short periods, with EF traffic always taking precedence over non-EF traffic. The burst-size limit enforced is based on the configurable burst-size limit. For a rate-limited logical interface, the Packet Forwarding Engine of the TOE calculates the optimum burst-size-limit values and then applies the value closest to the burst-size-limit value specified in the policer configuration.

There are no selection-based Security Functional Requirements defined in MOD_IPS_V1.0.

The ST does not claim any optional security functional requirements drawn from MOD_IPS_V1.0.

6.4 Security Functional Requirements Drawn from MOD_CPP_FW_V1.4

The fulfillment of the security functional requirements drawn from MOD_CPP_FW_V1.4 is given in Table 35.

There are no selection-based security functional requirements defined in MOD_CPP_FW_V1.4.

The ST does not claim any optional security functional requirements drawn from MOD_CPP_FW_V1.4.

Table 35 Fulfillment of the Security Functional Requirements Drawn from MOD_CPP_FW_V1.4

Security Functional Component	Fulfillment
FDP_RIP.2	The only resource made available to the information flowing through a TOE is the temporary storage of packet information when access is requested and when information is being routed. User data is not persistent when resources are released by one user/process and allocated to another user/process. Temporary storage (i.e. the memory of the TOE) used to build network packets is erased when the resource is called into use by the next user/process. The TOE records and keeps track of the length of each packet. When memory allocated from a previous user/process is released and the same memory is used to build the next network packet, the TOE pads any short packet with zeros. This ensures that each packet and the padding thereof fully fills the entire memory allocated for temporary storage of that packet. No residual information from packets in a previous information stream can traverse through the TOE.
FFW_RUL_EXT.1	The boot sequence of the TOE establishes the securing domain utilized for preventing tampering and bypass of the security functionality. This ensures that the packet filtering rules cannot be bypassed once the TOE is operational. The boot sequence for the TOE consists of the following steps: 1. BIOS hardware and memory checks, 2. Loading and initialization of the FreeBSD Kernel OS, 3. Execution of the FIPS self-tests and firmware integrity tests, 4. Starting of the init utility which mounts file systems, sets up network cards to communicate on the network, and generally starts all the processes that usually are run on a FreeBSD system at startup, 5. Starting of the daemon programs such as Internet Service Daemon (INETD), Routing Protocol Daemon (RPD), Syslogd are started, and the initialization of the routing and forwarding tables, 6. Loading the Management Daemon (or MGD) which makes accessible the management interface, and 7. activation of the physical interfaces. Once the interfaces are brought up, they will start to receive and send packets based on the configuration. They shall not receive or send any packets if not configured. The configuration must be set up by the Administrator. Interfaces are brought up only after successful loading of kernel and Information Flow subsystems. MGD is not loaded until after the kernel and INETD are initialized. This ensures that no modification to the security attributes can be made before the network is initialized. The trusted and untrusted network connection interfaces on the security appliance are not enabled until each component on the TOE is fully initialized and the TOE is ready to enforce the security policies. This ensures that

	Administrators are appropriately authorized when entering management commands and all network traffic is subject to the information flow policies. The INETD module implements the internet services for the TOE. It listens on designated ports used by internet services such as FTP. When a TCP or UDP packet arrives with a particular destination port number, INETD launches the appropriate server program (e.g., SSHD) to handle the connection. The TOE is configured to associate network interfaces to IP subnets. Source IP addresses are then associated with the network interface. TOE Software is composed of separate executables, or daemons. If a failure occurs in the flow daemon (flowd) causing it to halt, no packet processing will occur and no packets will be forwarded. A failure in another daemon will not prevent the flow daemon from enforcing the policy rule set. The Information Flow subsystem processes the packets arriving from the network to the TOE's network interface. Based on the Administrator-configured policy and the interface and zone information, the packet flows through the modules of the Information Flow subsystem. The modules enforce the information flow rules on the traffic. Rules within policies are processed in an Administrator-defined order. The default configuration of the TOE is to deny packets when there is no rule match. Alternative behavior may be triggered if another required condition allows the traffic. If a security risk is found in the packet. e.g. denial-of-service attacks, the packet is dropped and an event is logged. If the packet is not dropped by a given module, the interrupt handling routine calls the function for the next relevant module. If an interface is overwhelmed, each packet is dropped. This is recorded by the SNMP mibs as well as in a log. When an interface gets overwhelmed with CPU utilization 99% the packets are dropped with syslog entry 'CPU Utilization greater than 99, expect packet loss'. The Information Flow subsystem consists of the following modules: – IP Classification Module which retrieves information from packets received on the network interface device, classifies packets into several categories, saves classification information in packet processing context, and provides other modules with that information for assisting further processing. – Attack Detection Module which implements inline attack detection such as IP Spoofing. This module monitors arriving traffic, performs predefined attack detection services (prevents attacks), and issues actions when an attack is found. – Session Lookup Module which performs lookups in the session table used by all interfaces based on the information in incoming packets. Specifically, the lookup is based on the exact match of source IP address and port, destination IP address and port, protocol attributes (e.g., SYN, ACK, RST, and FIN), and egress/ingress zone. The input is passed to the module as a set of parameters from the Attack Detection module via a

	function call. The module returns matching wing if a match is found and 0 otherwise. Sessions are removed when terminated. – Session Setup Module is only activated for packets that do not match current established sessions. If a packet has a matched session, it will skip the Session Setup module and proceed to the Security Policy module. The Session Setup module also performs the auditing of denied packets. If there is a policy to specifically deny traffic, traffic matching this deny policy is dropped and logged in traffic log. If there is no policy to deny traffic, traffic that does not match any policy is dropped and not logged. In either case, Session Setup module does not create any sessions for denied traffic. Sessions are only created for allowed traffic. – Security Policy Module examines traffic passing through the TOE (via Session Setup module) and determines if the traffic can pass based on administrator-configured access policies. The Security Policy module is the policy enforcement engine which fulfills the security policy of the user. The Security Policy module will deny packets when there is no policy match unless another policy allows the traffic. – INETD Module provides the internet services for the TOE. The module listens on designated ports used by internet services. When a TCP or UDP packet arrives with a particular destination port number, INETD launches the appropriate server program (e.g., SSHD) to handle the connection. – The RPD (Routing Protocol Daemon) module contains the implementation and algorithms for the routing protocols and route calculations. It creates and maintains the Routing Information Base (RIB), which is a database of routing entries. Each routing entry consists of a destination address and some form of next hop information. RPD module maintains the internal routing table and properly distributes routes from the routing table to Kernel subsystem used for traffic forwarding at the Network interface. The TOE performs stateful network traffic filtering on network packets using the following network traffic protocols and network fields conforming to the described RFCs: – RFC 792 (ICMPv4): Type, Code – RFC 4443 (ICMPv6): Type, Code – RFC 791 (IPv4): Source address, Destination Address, Transport Layer Protocol – RFC 8200 (IPv6): Source address, Destination Address, Transport Layer Protocol – RFC 793 (TCP): Source port, Destination port – RFC 768 (UDP): Source port, Destination port Conformance to the RFCs is demonstrated by protocol compliance testing by the developer's product QA team.

	The TOE shall allow permit, deny, and log operations to be associated with rules and these rules can be assigned to distinct network interfaces. The TOE accepts network packets if it matches an established TCP, UDP or ICMP session using: – TCP: source and destination addresses, source and destination ports, sequence number, flags – UDP: source and destination addresses, source and destination ports – ICMP: source and destination addresses, type, code The TOE will remove existing traffic flows due to session inactivity timeout, or completion of the session. The TOE supports FTP (RFC 959) to dynamically establish sessions allowing network traffic according to Administrator rules. Session events will be logged in accordance with 'log' operations defined in the rules. Source and destination addresses, source and destination ports, transport layer protocol, and TOE Interface are recorded in each log record. The TOE implements what is referred to as an Application Layer gateway (ALG) which inspects FTP traffic to determine the port number used for data sessions. The ALG permits data traffic for the duration of the session, closing the port when the session ends. In this context, "session" refers to the TCP data transfer connection, not the duration of the FTP control session. The TOE enforces the default reject rules with logging on the following network traffic: – Invalid fragments, – Fragmented IP packets which cannot be re-assembled completely, – When the source address is equal to the address of the network interface where the network packet was received, – When the source address does not belong to the networks associated with the network interface where the network packet was received, – When the source address is defined as being on a broadcast network, – When the source address is defined as being on a multicast network – When the source address is defined as being a loopback address, – When the source address is a multicast address, – Where the source or destination address is a link-local address, – Where the source or destination address is defined as being an address "reserved for future use" as specified in RFC 5735 for IPv4, – When the source or destination address is defined as an "unspecified address" or an address "reserved for future definition and use" as specified in RFC 3513 for IPv6, and – With the IP options Loose Source Routing, Strict Source Routing, and Record Route specified. Packets are also checked for validity. The packets and fragments which violate the following rules are considered invalid and dropped with logging: – No overlap,

	– The total fragments in one packet are more than 62 pieces, – The total length of merged fragments is larger than 64k, – Each fragment in one packet does not arrive in 2 seconds, – The total queued fragments has reached platform-specific limitations, and – The total number of concurrent fragment processing for different packet has reached platform-specific limitations. The TOE can be configured to drop connection attempts after a defined number of half-open TCP connections using the screen 'tcp syn-flood', which provides both source and destination thresholds on the number of uncompleted TCP connections, as well as a timeout period. The source threshold option allows administrators to specify the number of SYN segments received per second from a single source IP address—regardless of the destination IP address—before the TOE begins dropping connection requests from that source. Similarly, the destination threshold option allows the Administrator to specify the number of SYN segments received per second for a single destination IP address before the TOE begins dropping connection requests to that destination. The timeout option allows administrators to set the maximum length of time before an uncompleted connection is dropped from the queue.
FMT_SMF.1/FFW	All management functions of the TOE are accessible to the administrators through the CLI. In addition to the management functions identified under FMT_SMF.1, FMT_SMF.1/IPS, and FMT_SMF.1/VPN, the CLI also implements the following Firewall-specific management functions: – Configuring the firewall rules.

6.5 Security Functional Requirements Drawn from MOD_VPNGW_v1.3

The fulfillment of the mandatory security functional requirements drawn from MOD_VPNGW_v1.3 is given in Table 36.

Table 36 Fulfillment of the Mandatory Security Functional Requirements Drawn from MOD_VPNGW_v1.3

Security Functional Component	Fulfillment
FAU_GEN.1/VPN	The TOE implements a universal audit function. Audit data processing is identical independently of the source of audit events. Auditing of the VPN function is performed with mechanisms identical to those described in FAU_GEN.1. In addition to the auditable events listed in FAU_GEN.1 and FAU_GEN.1/IPS, the TOE also generates the following VPN-specific audit records: – Indication that TSF self-test was completed, – Failure of self-tests, and

	– Auditable events defined in the Auditable Events for Mandatory Requirements table
FCS_CKM.1/IKE	The TOE generates cryptographic keys for IKE using RSA Schemes, ECC Schemes, and FFC Schemes: 1. RSA schemes are used to generate cryptographic key sizes of 2048-bit or greater. The keys are generated in accordance with FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix A.1. 2. ECC schemes are used to generate cryptographic keys for use with NIST curves P-256, P-384, and P-521. The ECC Schemes are used in accordance with FIPS PUB 186-5, “Digital Signature Standard (DSS)”, Appendix A.2. 3. FFC Schemes use ‘safe-prime’ groups are used for generating SSH session keys. The FFC Schemes are used in accordance with NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and RFC 3526. The TOE implements each "shall" and each "should" requirement from FIPS PUB 186-5 Appendix A.1 and A.2. It does not implement any of the "shall not" and "should not" requirements.
FMT_SMF.1/VPN	All management functions of the TOE are accessible to the administrators through the CLI. In addition to the management functions identified under FMT_SMF.1, FMT_SMF.1/IPS, and FMT_SMF.1/FFW, the CLI also implements the following VPN-specific management functions: – Definition of packet filtering rules, – Association of packet filtering rules to network interface, and – Ordering of packet filtering rules by priority.
FPF_RUL_EXT.1	See FFW_RUL_EXT.1.
FPT_FLS.1/SelfTest FPT_TST_EXT.3	The TOE runs the following set of self-tests when powered on to verify the correct operation of the TOE software: 1. Power on test to determine that the boot-device responds and performs a memory size check to confirm the amount of available memory. 2. File integrity test to verify each mounted signed package and to assert that system files have not been tampered with. To test the integrity of the firmware, the SHA-1 fingerprints of the executables and other immutable files are regenerated and validated against the reference fingerprints contains in the manifest file. 3. Crypto integrity test to check the integrity of cryptographic keys and major CSPs. 4. Authentication error to verify that veriexec is enabled and operates as expected using /opt/sbin/kats/cannot-exec.real. 5. Kernel, libmd, OpenSSL, QuickSec, SSH tests to verify the output from known answer tests for the cryptographic algorithms.

	The TOE only executes binaries supplied by Juniper Networks. Within the package containing the TOE software, each Junos OS firmware image includes fingerprints of the executables and other immutable files. The TOE will not execute any binary without validating the registered fingerprint. This protects the TOE from unauthorized firmware which might compromise the integrity of the TOE. The self-tests ensure that only authorized executables are allowed to run and ensure the correct operation of the TOE. In case of a corrupt state or a failure in a self-test, the TOE will panic. The event will be logged, the TOE will cease processing network traffic and CLI commands, and restart. When the TOE restarts, the boot process shall not succeed without passing each self-test. This constitutes the automatic recovery and self-test behavior of the TOE
FTP_ITC.1/VPN	The TOE implements trusted channels for VPN connections using IPsec. IPsec may be used in tunnel mode to establish VPN connections with IPsec peers.

The fulfillment of the selection-based security functional requirements drawn from MOD_VPNGW_v1.3 is given in Table 37.

Table 37 Fulfillment of the Selection-Based Security Functional Requirements Drawn from MOD_VPNGW_v1.3

Security Functional Component	Fulfillment
FIA_PSK_EXT.1 FIA_PSK_EXT.2	The TOE accepts generated pre-shared keys used for IPsec and IKEv2. The pre-shared keys are shared between the peers by out of band means.

The ST does not claim any optional security functional requirements drawn from MOD_VPNGW_V1.3.

6.6 Fulfillment of the Security Assurance Requirements

To fulfill the Security Assurance Requirements, the developer implements a set of security assurance measures. Some assurance classes are fulfilled by the evaluator of the TOE. The security assurance measures implemented by the developer and evaluator of the TOE are described in Table 38.

Table 38 Fulfillment of the Security Assurance Requirements

Security Assurance Requirement	Fulfilment
Security Target	The developer authors a Security Target (ST) for the Target of Evaluation. The ST implements all assurance components required by the PP and includes – A ST Introduction which provides a ST Reference, a TOE Reference, a TOE Overview, and a TOE Description.

	– Conformance Claims stating exactly the conformance to the Common Criteria and the Protection Profile. – A Security Problem Definition which is a statement of Threats, Assumptions and Organizational Security Policies applicable to the TOE. – A statement of the security objectives. The PP only defines security requirements for the operational environment of the TOE. None are stated for the TOE. – Extended Components Definition and the statement of the security requirements state exactly the Security Functional Requirements, and the Security Assurance Requirements the TOE fulfills. – TOE Summary Specification which describes for each Security Requirement how the TOE fulfills that Security Requirement. The refinement to ASE_TSS.1.1C is fulfilled by the developer providing a separate, proprietary entropy assessment report.
Functional Specification	Included in the TOE Summary Specification, the developer provides all information required for a basic functional specification of the TOE.
Security Guidance	Attached to the TOE and included in the physical scope of the TOE is a Common Criteria Guidance Supplement for the TOE. The Guidance Supplement gives guidance to the user of the TOE in the secure installation and preparation of the TOE so that the TOE is in an initial secure state. The Guidance Supplement also provides guidance to the user of the TOE so that the TOE always remains in a secure state when the guidance is followed.
Life Cycle Support	The developer labels the TOE with the unique identifier. The label may be examined by the user of the TOE to ensure that the correct version of the TOE is used. When the TOE software is updated, the label of the TOE is updated accordingly. The TOE label is included in the configuration list of the TOE to ensure that the evaluator can be assured of evaluating the intended version of the TOE. The developer ensures that each reported security flaw in each release of the TOE is tracked and corrected using a well-defined, methodical process.
Independent Testing	The evaluator carries out a set of independent tests on the TOE. The independent tests complement the functional testing carried out by the developer and ensure that the TOE passes each applicable test required for conformance with the PP. The evaluator documents the testing in accordance with the requirements stated in the PP and the Common Criteria evaluation and certification scheme followed.
Vulnerability Assessment	The evaluator carries out a vulnerability survey to determine that there are no obvious vulnerabilities in the TOE which could be practically exploited by the threat agents. The evaluator documents the vulnerability survey in accordance with the requirements stated in the PP and the Common Criteria evaluation and certification scheme followed.

6.7 Cryptographic Details and CAVP References

This section provides additional details on the cryptographic algorithms and protocols implemented by the TOE.

6.7.1 Zeroization of Cryptographic Keys and Critical Security Parameters

The timing and method of the zeroization of the cryptographic keys and critical security parameters (CSP) used by the TOE is given in Table 39.

Table 39 Timing and Method of the Zeroization of Cryptographic Keys and Critical Security Parameters

Key/CSP	Storage Format	Storage Location	Zeroization Method
SSH Private Host Key	Plaintext	Non-volatile memory	When the TOE is recommissioned, the config files (including SSH keys) are erased by the administrator using the `request vmhost zeroize no-forwarding` CLI command
	Plaintext	Volatile memory	`free()` performed by the TOE software at session termination
SSH Session Key	Plaintext	Volatile memory	`free()` performed by the TOE software at session termination
User Password	Plaintext when entered	Volatile memory	`free()` performed by the TOE software at the completion of the user authentication
	Hashed when stored	Non-volatile memory	When the TOE is recommissioned, the config files (including user passwords in the password file) are erased by the administrator using the `request vmhost zeroize no-forwarding` CLI command
RNG State	Plaintext	Volatile memory	Overwritten by the kernel of the TOE with zeros at reboot
System Master Password	Plaintext	Non-volatile memory	Zeroized by the administrator by issuing the `request vmhost zeroize no-forwarding` CLI command
IKE Private Host Key	Plaintext	Disk/Memory	`clear security ike security-association` command (‘`clear security IKE security-association ha-link-encryption`’ for the HA control link tunnel) or reboot the box. Private keys stored in flash are not zeroized unless an explicit `request vmhost zeroize` is executed.
IKE-SKEYID	Plaintext	Memory	`clear security ike security-association` command (‘`clear security IKE security-association ha-link-encryption`’ for the HA control link tunnel) or reboot the box.
IKE Session Keys	Plaintext	Memory	`clear security ike security-association` command (‘`clear security IKE`

			security-association ha-link-encryption' for the HA control link tunnel) or reboot the box.
ESP Session Key	Plaintext	Memory	clear security ike security-association command ('clear security IKE security-association ha-link-encryption' for the HA control link tunnel) or reboot the box.
IKE-DH Private Exponent	Plaintext	Memory	clear security ike security-association command ('clear security IKE security-association ha-link-encryption' for the HA control link tunnel) or reboot the box.
IKE-PSK	Encrypted	Disk/Memory	clear security ike security-association command ('clear security IKE security-association ha-link-encryption' for the HA control link tunnel) or reboot the box. Key values stored in flash are not zeroized unless an explicit request vmhost zeroize is executed.
ecdh private keys	Plaintext	Memory	Memory free() operation is performed by Junos upon session termination

6.7.2 CAVP Certificate References

The TOE implements a rich set of cryptographic functions used to protect communications and the integrity of the security functions. Each cryptographic function of the TOE is CAVP validated. The CAVP certificate references, organized by the applicable Security Functional Component, are given in Table 40.

Unless otherwise stated, each CAVP Certificate is applicable to each variant of the TOE.

Table 40 CAVP Certificate References

FCS_CKM.1			
Applicable SFR	**Algorithm and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as a server and client)	RSA Probable Random Prime (2048, 3072, 4096) (KeyGen)	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as a server and client)	ECDSA (P-256, P-384, P-521) (KeyGen, KeyVer)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	ECDSA (P-256, P-384, P-521) (KeyGen, KeyVer) (for ECDH)	OpenSSL v1.1.1	A7999

FCS_CKM.1/IKE			
Applicable SFR	**Algorithm and Parameters**	**Part**	**CAVP Reference**
FCS_IPSEC_EXT.1	RSA Probable Random Prime (2048, 3072, 4096) (KeyGen)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	ECDSA (P-256, P-384, P-521) (KeyGen, KeyVer)	OpenSSL v1.1.1	A7999
FCS_CKM.2			
Applicable SFR	**Algorithms and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as a server and client)	DH Group (16, 18) NOTE: Tested against known-good implementation	OpenSSL v1.1.1	N/A
FCS_SSH_EXT.1 (TOE acting as a server and client)	KAS-ECC-SSC (ECDH) (P-256, P-384, P-521)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	DH (Group 14, 15, 16) NOTE: Tested against known-good implementation	OpenSSL v1.1.1	N/A
FCS_IPSEC_EXT.1	KAS-ECC-SSC (ECDH) (P-256, P-384, P-521)	OpenSSL v1.1.1	A7999
FCS_COP.1/DataEncryption			
Applicable SFR	**Algorithm and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as a server and client)	AES-CBC (128, 256) (Encryption, Decryption)	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as a server and client)	AES-CTR (128, 256) (Encryption, Decryption)	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as a server and client)	AES-GCM (128, 256) (Encryption, Decryption)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	AES-CBC (128, 192, 256) (Encryption, Decryption)	QAT	A8346
FCS_IPSEC_EXT.1	AES-GCM (128, 256) (Encryption, Decryption)	QAT	A8346

FCS_IPSEC_EXT.1 (SRX4700 only)	AES-GCM (128, 256) (Encryption, Decryption)	YT Card (Juniper Trio 6 Chip)	A4664
FCS_IPSEC_EXT.1	AES-CBC (128, 192, 256) (Encryption, Decryption)	QuickSec	A8024
FCS_IPSEC_EXT.1	AES-GCM (128, 256) (Encryption, Decryption)	QuickSec	A8024
FCS_COP.1/SigGen			
Applicable SFR	**Algorithms and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as Server and Client)	ECDSA (P-256 w/SHA-256, P-384 w/SHA-384, P-521 w/SHA-512) (SigGen, SigVer)	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as Server and Client)	RSA PKCS1v1_5 (n=2048 w/ SHA-256, n=3072 w/ SHA-256, n=4096 w/ SHA-256) (SigGen, SigVer)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	ECDSA (P-256 w/SHA-256, P-384 w/SHA-384, P-521 w/SHA-512) (SigGen, SigVer)	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	RSA PKCS1v1_5 (n=2048 w/SHA-256, n=3072 w/SHA-256, n=4096 w/SHA-256) (SigGen, SigVer)	OpenSSL v1.1.1	A7999
FPT_TUD_EXT.1	ECDSA (P-256 w/SHA-256) (SigGen, SigVer)	OpenSSL v1.1.1	A7999
FCS_COP.1/Hash			
Applicable SFR	**Algorithms and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as Server and Client)	SHA-256, SHA-384, SHA-512	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as Server and Client)	SHA-256, SHA-384, SHA-512	Kernel	A7999
FCS_SSH_EXT.1 (TOE acting as Server and Client)	SHA-256	LibMD	A7987
FCS_NTP_EXT.1 FPT_TUD_EXT.1	SHA-256	OpenSSL v1.1.1	A7999
FCS_IPSEC_EXT.1	SHA-256, SHA-384, SHA-512	QAT	A8346

FCS_IPSEC_EXT.1	SHA-256, SHA-384, SHA-512	QuickSec	A8024
FCS_COP.1/KeyedHash			
Applicable SFR	**Algorithms and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as Server and Client)	HMAC-SHA2-256, HMAC-SHA2-512	OpenSSL v1.1.1	A7999
FCS_SSH_EXT.1 (TOE acting as Server and Client)	HMAC-SHA256	Kernel	A7924
FCS_SSH_EXT.1 (TOE acting as Server and Client)	HMAC-SHA256	LidMD	A7987
FCS_IPSEC_EXT.1	HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512	QAT	A8346
FCS_IPSEC_EXT.1	HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512	QuickSec	A8024
FCS_RBG_EXT.1			
Applicable SFR	**Algorithms and Parameters**	**Part**	**CAVP Reference**
FCS_SSH_EXT.1 (TOE acting as Server and Client) FCS_IPSEC_EXT.1	HMAC-DRBG (SHA-512)	Kernel	A7924

7 Acronyms

AES	Advanced Encryption Standard
ASCII	American Standard Code for Information Interchange
BIOS	Basic Input Output System
CA	Certificate Authority
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CC	Common Criteria
CCEVS	Common Criteria Evaluation and Validation Scheme
CEM	Common Evaluation Methodology
CLI	Command Line Interface
CMVP	Cryptographic Module Validation Program
CRL	Certificate Revocation List
CSP	Critical Security Parameter
CSR	Certificate Signing Request
CTR	Counter Mode
DRBG	Deterministic Random Bit Generator
DSS	Digital Signature Standard
ECDSA	Elliptic Curve Digital Signature Algorithm
ESP	Encapsulating Security Payload
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standard
FIPS PUB	FIPS Publication
FTP	File Transfer Protocol
FW	Firewall
GB	Giga-Bit
GCM	Galois Counter Mode
HA	High Availability
HMAC	Hash-Based Message Authentication Code
ICL	Interchassis Link
ICMP	Internet Control Message Protocol

IDS	Intrusion Detection System
IKE	Internet Key Exchange
IKE_SA	Internet Key Exchange Security Association
IKEv2	Internet Key Exchange version 2
IMIX	Internet Mix
IOC	Interface Card
IP	Internet Protocol
IPS	Intrusion Prevention System
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
ISO	International Organization for Standardization
IT	Information Technology
KW	Key Wrap
LAN	Local Area Network
LED	Light Emitting Diode
MAC	Media Access Control or: Message Authentication Code
MPC	Modular Port Concentrator
NAT	Network Address Translation
NGFW	Next-Generation Firewall
NIAP	National Information Assurance Partnership
NIC	Network Interface Card
NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
OID	Object Identity
OS	Operating System
OSP	Organizational Security Policy
PAM	Pluggable Authentication Modules
PDF	Portable Document Format
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure

PRF	Pseudorandom Function
PSS	Improved Probabilistic Signature Scheme
QA	Quality Assurance
RE	Routing Engine
RSA	Rivest-Shamir-Adleman
RSASSA	RSA Signature Scheme with Appendix
RFC	Request For Comments
RBG	Random Bit Generator
RPD	Routing Protocol Daemon
SA	Security Association
SCB	Switch Control Board
SD	Supporting Document
SDN	Software-Defined Networking
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol
SPC	Services Processing Card
SSH	Secure Shell
SSHD	SSH Daemon
SSL	Secure Socket Layer
TLS	Transport Layer Security
TSF	TOE Security Function
TTL	Time To Live
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
VM	Virtual Machine
VPN	Virtual Private Network