Juniper vSRX3.0 with Junos OS 22.2R2 Security Target

Document Version: 0.9

2400 Research Blvd
Suite 395
Rockville, MD 20850

Revision History

Version	Date	Changes
Version 0.1	June 05, 2021	First internal review version
Version 0.2	August 11, 2021	Incorporated comments from the developer's review
Version 0.3	March 02, 2023	Updated to include NTP Claim and exclude TUD X.509 Certificate-based Claim
Version 0.4	April 04, 2023	Updated according to mod_vpngw_v1.2
Version 0.5	May 18, 2023	Updated according to ECR Comments
Version 0.6	June 19, 2023	Updated according to AAR Comments
Version 0.7	August 18, 2023	Updated according to Check-in ECR Comments
Version 0.8	September 27, 2023	Updated according to QA Comments
Version 0.9	December 14, 2023	Updated according to ECR comments.

Contents

1 Introduction

The Security Target (ST) serves as the basis for the Common Criteria (CC) evaluation and identifies the Target of Evaluation (TOE), the scope of the evaluation, and the assumptions made throughout. This document will also describe the intended operational environment of the TOE, and the functional and assurance requirements that the TOE meets.

1.1 Security Target and TOE Reference

This section provides the information needed to identify and control the TOE and the ST. ST and TOE identification data is given in Table 1.

Table 1 – TOE/ST Identification

Category	Identifier
ST Title	Juniper vSRX3.0 with Junos OS 22.2R2 Security Target
ST Version	0.9
ST Date	November 29, 2023
ST Author	Acumen Security, LLC
TOE Identifier	Juniper vSRX3.0 with Junos OS 22.2R2
TOE Version	vSRX3.0 with Junos OS 22.2R2
TOE Developer	Juniper Networks, Inc.
Key Words	Network Device, Virtual Firewall, Packet Filtering, Stateful Filtering, Intrusion Prevention, Virtual Private Network, Cluster Mode, High Availability

1.2 TOE Overview

The TOE is the Juniper Networks, Inc. Juniper vSRX3.0 with Junos OS 22.2R2 Virtual Firewall. It is intended for deployment with service providers and large enterprises. The TOE may be operated in single mode or in cluster mode. Cluster mode is a High Availability (HA) mode in which two instances of a TOE are connected and configured to operate like a single device. This ensures high availability in the case of equipment malfunction in one of the devices.

The TOE allows definition of packet filtering policies which are enforced on all traffic traversing to, from or through it. Each packet is also subjected to stateful inspection. Further security is added by an intrusion prevention function. All traffic is monitored against signatures of known attacks and for abnormalities in traffic patterns. If potentially malicious traffic is detected, protective action is taken. Security policies are managed, and the TOE configuration controlled by Security Administrators. Management occurs via a Command Line Interface (CLI) from a local or remote management station.

The TOE is deployed as a gatekeeper between two networks so that all traffic between the two networks passes through an instance of the TOE. This ensures that all traffic between the two networks is subject to the security policies the TOE enforces. Traffic and information flows are controlled based on the rules set by TOE Administrators concerning network node addresses, protocol, type of access requested, and the service requested. The TOE implements a default deny rule, i.e. it drops any network traffic not explicitly allowed by the rules. All security relevant activities and events are audited.

Additionally, the TOE implements a multi-site Virtual Private Network (VPN) gateway functionality for tunneling traffic between itself and a VPN peer. In Cluster Mode, the link between the two instances of TOE may also be secured with IPsec. If the audit records are forwarded to an external syslog server, the connection between the TOE and the syslog server may be protected with SSH. The connection between the TOE and a remote management station is also protected by SSH.

TOE software is deployed with a hypervisor and a x86 server. The user configures the hypervisor on the selected server and installs the TOE software on the hypervisor. The software is downloaded from the Juniper web site. TOE Software is protected with a digital signature and hash values. The TOE verifies the signatures and hash values at the boot up and executes a full suite of self-tests to ensure that the TOE functions correctly and only authentic TOE software is executed.

1.3 TOE Description

The TOE implements Junos Control Plane (JCP) and Packet Forwarding Engine (PFE) which constitutes the Junos data plane. JCP and PFE are executed on the virtual CPUs (vCPU) which are part of the environment of the TOE. JCP is executed on one vCPU and the PFE on at least one vCPU. The vCPU number can be increased for improved performance. The complete vSRX3.0 architecture and the TOE within it is illustrated in Figure 1.

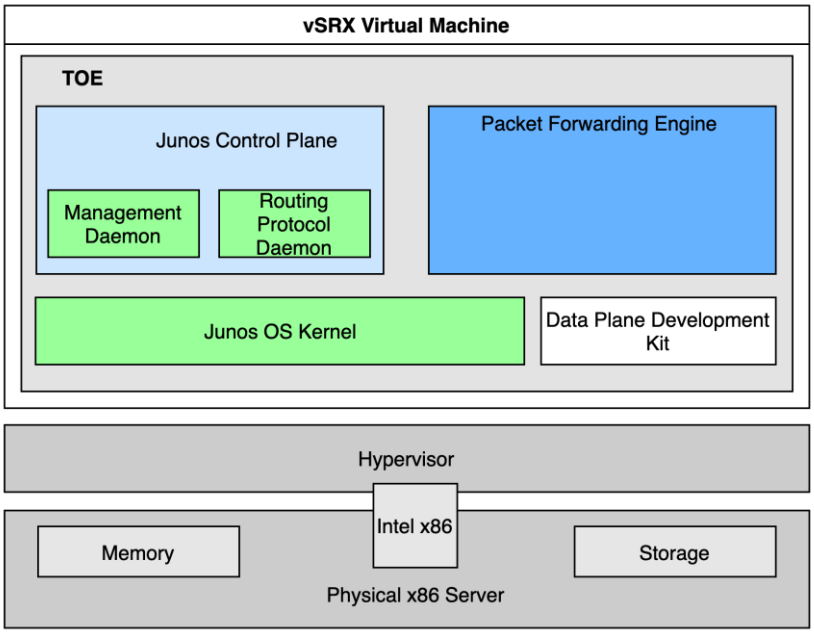

Figure 1 – vSRX3.0 Architecture

Junos Control Plane (JCP) is the virtual Routing Engine (vRE) which implements Layer 3 routing services. It also implements all network management functions for the configuration and operation of the TOE and controls the flow of information through the TOE. Controlling the flow of information through the TOE includes Network Address Translation (NAT) and the encryption and decryption of packets for secure communication over IPsec.

Packet Forwarding engine (PFE) implements all operations necessary for the forwarding of transit packets. That includes Flow Processing and Advanced Services.

JCP and PFE operate independently but communicate constantly over a high-speed internal link implemented by the Junos OS. This ensures effective forwarding and routing control and the capability to run Internet-scale networks at high speeds.

The Junos OS kernel uses the underlying hypervisor as a virtualization infrastructure to create multiple virtual machines (VMs). Only a single VM is allowed in an evaluated configuration and no additional appliances may be installed. The hypervisor is not part of the TOE and functions as a pass-through layer only.

The TOE is configured with from three to eight virtual Network Interface Cards (vNIC). Each vNIC must be mapped to a different physical NIC. The physical server must have at least as many physical NICs as the number of vNICs configured in vSRX3.0.

The default mode for the TOE is a single mode but it may be configured for Cluster Mode by connecting ge-0/0/1 on node 0 to ge-0/0/1 on node 1. An example of a Cluster Mode configuration is given in Figure 2. Any other configuration of the physical ports has to be removed prior to the Cluster Mode configuration. The two instances of the TOE must be in an identical configuration except for one being configured to node 0 and the other to node 1.

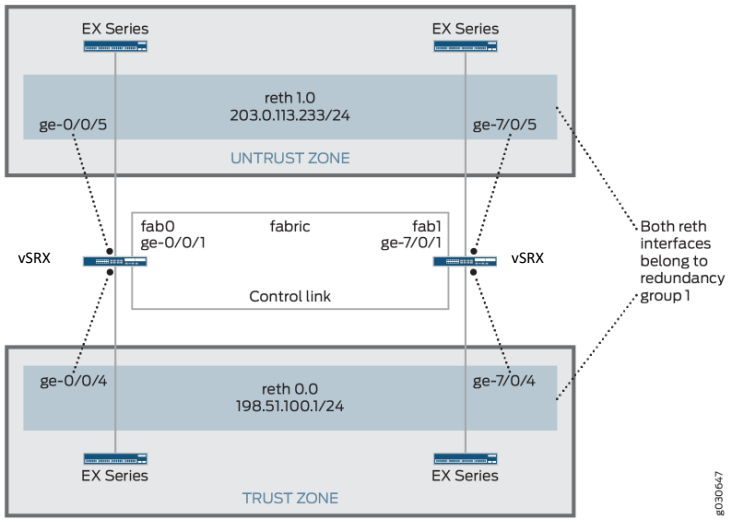

Figure 2 – Cluster Mode Configuration of the TOE

A dedicated physical interface acts as the fxp0 interface for the HA management of the TOE. The fxp1 interface for HA control link is ge-0/0/1. Administrators may define the preferred fiber interface. Once the Administrator has defined and set up the cluster, the two devices constitute a chassis cluster have an identical cluster-id, but each has a different node ID. One of the hosts has node ID 0 and the other one node ID 1.

Node 1 renumbers its interfaces by adding the total number of system FPCs to the interface's original FPC number. The fabric interface remains Administrator-defined. Critical security parameters shared by the two instances of the TOE are protected by IPsec.

1.3.1 Physical Boundaries

The Physical scope of the TOE includes TOE Software, TOE Hardware and TOE Security guidance.

TOE Software consists of the following:

- Junos OS 22.2R2 for vSRX3.0 software, including the vSRX3.0 Virtual Machine

TOE Hardware must have at least the number of NICs as there are vNICs configured in the TOE. It must be one of the following:

- HP ProLiant DL380p Gen9 with Intel Xeon E5-2600 v4 series
- PacStar 451 with Intel Xeon E-2200M series

TOE Guidance is the following:

- Junos OS Common Criteria Guide for vSRX3.0 Release 22.2R2, September 29, 2023

1.3.2 Security Functions Provided by the TOE

The TOE implements the security functions required by the Collaborative Protection Profile for Network Devices, referred to as [CPP_ND_V2.2E]. The TOE also implements the additional security functions required by the PP-Module for Stateful Traffic Filter Firewalls ([MOD_FW_V1.4E]), PP-Module for Virtual Private Network (VPN) Gateways ([MOD_VPNGW_V1.2]) and PP-Module for Intrusion Prevention Systems (IPS) ([MOD_IPS_V1.0]). The security functions the TOE implements are summarized in the following.

1.3.2.1 Security Audit

The TOE implements an audit function which generates an audit record for each auditable event. Audit logs containing audit records are stored in protected syslog files in the VM filesystem. Syslog files may be forwarded to an external log server via Netconf over SSH.

Auditable events include start-up and shutdown of the audit functions, authentication events, configuration changes, management operations on cryptographic keys, resetting of passwords, IPS events, service requests, and each other event listed in Table 9 and Table 10. Audit records include, where applicable, the date and time of the event, event category, event type, username of the user causing the event, and the success or failure of the event.

The amount of storage available for local syslog files is configurable by the Administrator. The TOE monitors the size of the syslog file against the configured limits. If the storage limit is reached, the TOE shall overwrite the existing audit records. The oldest audit records shall be overwritten first.

1.3.2.2 Cryptographic Support

The TOE implements IPSec with Internet Key Exchange IKEv1 and IKEv2 as well as SSH to protect communication with peer entities. All required cryptographic algorithms, key management functions and random bit generation methods are implemented by the TOE.

IPSec is implemented in tunnel mode with the payload encrypted with AES in GCM, CBC and CTR modes with 128-bit, 192-bit and 256-bit key lengths using HMAC-SHA-256. The symmetric keys used for IPSec are generated using IKEv1 and IKEv2. The TOE implements a random bit generator which generates the secret exponent for use in IKE DH-key exchange. Random bits are generated using HMAC-DRBG seeded by hardware and software noise sources.

Both RSA and ECDSA are implemented. DH Groups 14, 19 and 20 are implemented and the secret exponent is generated to be of appropriate length for each, i.e. 112 bits for DH Group 14, 128 bits for DH Group 19 and 192 bits for DH Group 20. The random number generator is also used for generating the nonces. Both RSA and ECDSA may be used for peer authentication with RFC 4945-conformant X.509 certificates in the IKE exchange. Pre-shared keys may also be used.

SSH is implemented in accordance with the relevant RFC suite. Both public key based and password based authentication are implemented. Public key authentication uses ECDSA with SHA on NIST curve P-256, P-384 or P-521. Key exchange may additionally use DH Group 14 with SHA-1. AES is used for protecting the payload in CBC or CTR mode with a 128-bit or 256-bit key and with HMAC-SHA1, HMAC-SHA2-256 or HMAC-SHA2-256 as data integrity algorithm. Maximum key life-time is one hour or at most one GB of data. After any of the thresholds are met, a rekeying shall be performed. Cryptographic keys are destroyed both from the volatile and the non-volatile memory when no longer required.

1.3.2.3 Identification and Authentication

Identification and authentication concerns with human users and with the peer entities of the TOE. Human users are identified with a user name and authenticated with a password. IPsec peer entities are

identified by a Security Association (SA) established by IKE and authenticated using X.509 certificates or pre-shared key. SSH peer entities are identified by IP address and authenticated by a public key protocol or a password.

Human users accessing the TOE from the console are authenticated by password. When human users access the TOE from a remote management station the TOE first establishes an SSH connection between the management workstation and itself, then authenticates the human user over the SSH connection with a password.

When authenticating peer entities for VPN connection establishment, the TOE first authenticates the IPSec peer entity using X.509 certificates or pre-shared keys, and then the SSH peer entity using public-key or password based authentication. Upon successful IPSec and SSH peer entity authentication the TOE establishes a VPN session with SSH tunneled over IPSec.

The TOE may have several human users but only implements a single role, Security Administrator. Each user has a unique user name and a password which shall be entered to the TOE for identification and authentication. The password should be known only to the user and is not displayed in clear by the TOE. Only upon successful identification and authentication shall the user be assigned to the role of a Security Administrator. The TOE displays a warning banner at the authentication window to inform the users of the restrictions on access and of the consequences of unauthorized use.

If a user authentication attempt fails, the user is required to re-attempt authentication. The TOE keeps track of the number of failed attempts and compares it to an Administrator-configured number of maximum allowed authentication attempts. If the maximum number is met, the TOE shall prevent the user from establishing any remote sessions with the TOE until an Administrator-defined time-out period has elapsed.

Users may select their own password, but the TOE only accepts them if they meet a defined quality criterion. A password must contain characters of at least two of the character groups (upper case letters, lower case letters, numbers, and special characters). The length of a password must be at least the minimum length configured by Administrators but not less than ten characters.

The TOE implements X.509 authentication for IPSec peers. Alternatively, pre-shared key based authentication may be used. X.509 certificates are validated against pre-defined rules and require a minimum path length of three certificates. The certificates are validated from the Root CA upon the TOE receiving a CA Certificate Response. In case the TOE cannot establish the validity of a certificate, the Administrator is prompted to reject or accept the certificate.

1.3.2.4 Security Management

The TOE implements a rich Command Line Interface (CLI) the Administrators (i.e. users successfully authenticated and assigned to the role Security Administrator) may use to configure and manage the TOE. No other method of management but the CLI exists. All security management functions are available to the Administrators via the CLI locally from the console or remotely over SSH.

1.3.2.5 Protection of the TSF

The TOE ensures that the cryptographic keys and passwords are stored in a secure manner.

The TOE provides a reliable timestamp for its own use. The reliable timestamp can be set by a security administrator or authenticated NTP.

Cryptographic keys may only be accessed by authorized processes. Keys are erased when no longer required. Passwords are hashed before storage and may not be recovered to. When a password is read from a user, it is obscured on the screen and hashed prior to comparison to the stored password.

TOE Software is associated to a digital signature and a hash value. The signature and the hash value may be used by Administrators to verify the integrity of the software. In case of an upgraded software being made available by the developers, the Administrator may download the upgraded software from the developer's web site for installation on the TOE. The upgrade is associated with a digital signature which must be successfully verified prior to the installation of the upgrade.

The TOE also implements a set of critical self-tests at the start-up. These tests include power-on tests, file integrity test, cryptographic function and key integrity test, authentication test, known answer tests for cryptographic algorithms, and health tests for the noise sources used in the random bit generation. If any of the power-on self-tests, verification of the TOE software digital signature, or the testing of the noise source health fails, the TOE shall shut itself down as a defensive measure.

1.3.2.6 TOE Access

The TOE implements measures to ensure that inactive sessions may not be used by unauthorized users. The TOE keeps track of session inactivity and terminates any session where the maximum inactivity time has been reached. The CLI used for administering the TOE includes a `logout` call which the users may use to terminate their own sessions.

The TOE also displays an access banner at each authentication exchange. The access banner may be configured by the Administrators and contains an advisory notice and consent warning informing users of the legitimate use of the TOE and the sanctions for attempts of unauthorized use.

1.3.2.7 Trusted Path/Channels

The TOE implements a VPN gateway functionality which allows a trusted channel between itself and VPN peers for tunneling network traffic. The VPN peer may request a VPN connection between itself and the TOE. The VPN connection is an SSH connection tunneled over IPSec.

In Cluster Mode, the Administrators may configure the communication between the two nodes be protected with IPsec.

The TOE also implements an SSH server for a trusted path between itself and a remote management station. The management station may request establishment of an SSH connection. Upon successful connection establishment, all communication between the remote management station and the TOE, including the authentication exchange between the user and the TOE as well as all subsequent CLI commands, shall be exchanged over SSH.

1.3.2.8 Packet Filtering

Administrators may define rules for the TOE to use to filter each TCP/IP packet. The rules may be assigned to any network interface. Each packet is inspected individually, and the TOE ensures that each packet is erased from the buffer it is stored for inspection prior to the storage of the next packet in the same buffer. This ensures that each inspection is only on the fields of the packet and no residual information from previously inspected packets influences the inspection.

Packet filtering rules may be defined on IPv4, IPv6, TCP and UDP header information. IP packet rules may use source and destination addresses as well as the protocol (or next header in IPv6). TCP and UDP datagrams are filtered by rules using source and destination ports. Each packet is handled according to the first matching rule in the rule base. Any traffic for which there is no matching rule shall be dropped. For a matching rule, the TOE shall permit the packet depending on the rule and may also log the event.

1.3.2.9 Firewall

Administrators of the TOE may also define rules for stateful traffic filtering of network traffic based on ICMPv4, ICMPv6, IPv4, IPv6, TCP and UDP protocol fields. Rules may be defined for each network port of the TOE individually. The TOE inspects all incoming and outgoing traffic against those rules and permits or denies the traffic based on the rules. Additional rules are enforced to ensure that traffic which is illegitimate on high likelihood shall be dropped and a log entry generated. The rules are examined in order and the traffic is examined and filtered according to the first matching rule. If no rule matches the traffic, the traffic shall be dropped.

1.3.2.10 Intrusion Prevention

The TOE implements intrusion prevention capabilities to prevent potentially malicious network traffic from reaching the protected network. The capabilities are implemented by three distinct means:

- The TOE allows Administrators to define lists of known-good and known-bad IP source and destination addresses. Any IP datagram matching an entry in the known-bad list shall be dropped and any IP datagram matching an entry in the known-good list shall be allowed.
- The TOE maintains a list of attack signature on network traffic headers and payload and inspects all network traffic against those attack signatures. Administrator action is not required to define the rules, but the Administrator may configure the TOE action taken when traffic matches an attack signature. The Administrator may decide to allow the traffic flow, send a TCP reset to the source or destination of the malicious traffic, or block the traffic flow.
- Administrators may also define patterns for regular network traffic and express threshold values indicating a deviation from the expected, regular traffic. Deviations may occur because of an unusual traffic volume, an unusual time of the day, or an unusual frequency of traffic. Upon detecting a deviation from regular traffic, the TOE shall take Administrator-configured action to prevent the potentially malicious traffic from reaching the protected network.

1.4 TOE Environment

The following environmental components are required to operate the TOE in the evaluated configuration:

- VMWare ESXi 7 Update 3 Hypervisor
- Syslog server supporting SSHv2 connections for the TOE to send audit logs to,
- SSHv2 client running in the remote management station,
- Serial connection client for the local management station, and
- IPSec peer and SSHv2 client on any VPN peer.
- NTP Server

1.5 Product Functionality not Included in the Scope of the Evaluation

The following product functionality is not included in the CC evaluation:

- Telnet, FTP, SSL and SNMP as they violate the secure access requirements,
- Management of the TOE via J-Web, JUNOScript or JUNOScope,
- Any use of CLI account super-user or Linux root account,
- Hosting of more than one Virtual Machines on one physical platform, and
- Hardware of the x86 server hosting the VMWare ESXi 7 Hypervisor.

2 Conformance Claims

This section identifies the TOE conformance claims, conformance rationale, and relevant Technical Decisions (TDs).

2.1 CC Conformance Claims

The TOE is conformant to the following:

- Common Criteria for Information Technology Security Evaluations Part 1, Version 3.1, Revision 5, April 2017
- Common Criteria for Information Technology Security Evaluations Part 2, Version 3.1, Revision 5, April 2017 (Extended)
- Common Criteria for Information Technology Security Evaluations Part 3, Version 3.1, Revision 5, April 2017 (Conformant)

2.2 Protection Profile Conformance

This ST claims exact conformance to the following:

- PP-Configuration for Network Device, Intrusion Protection Systems (IPS), Stateful Traffic Filter Firewalls, and Virtual Private Network (VPN) Gateways, 2022-04-06 [CFG_NDcPP-IPS-FW-VPNGW_V1.1]
    - Base-PP: collaborative Protection Profile for Network Devices, Version 2.2e (CPP_ND_V2.2E)
    - PP-Module: PP-Module for Intrusion Protection Systems (IPS), Version 1.0 (MOD_IPS_V1.0)
    - PP-Module: PP-Module for Stateful Traffic Filter Firewalls, Version 1.4 + Errata 20200625 (MOD_FW_1.4E)
    - PP-Module: PP-Module for Virtual Private Network (VPN) Gateways, Version 1.2 (MOD_VPNGW_1.2)

2.3 Conformance Rationale

This ST provides exact conformance to the items listed in the previous section. The security problem definition, security objectives, and security requirements in this ST are all taken from [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2], performing only the operations defined therein.

2.4 Technical Decisions

All NIAP TDs issued to date and applicable to [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2] have been considered. Table 2 identifies all applicable TDs.

Table 2 – Relevant Technical Decisions

Technical Decision	Applicable (Y/N)	Exclusion Rationale (if applicable)
TD0527: Updated to Certificate Revocation Testing (FIA_X509_EXT.1)	Y	
TD0528: NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	Y	

Technical Decision	Applicable (Y/N)	Exclusion Rationale (if applicable)
TD0536: NIT Technical Decision for Update Verification Inconsistency	Y	
TD0537: NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	N	The TOE does not implement TLS.
TD0545: NIT Technical Decision for Conflicting FW rules cannot be configured (extension of RfI#201837)	Y	
TD0546: NIT Technical Decision for DTLS – clarification of Application Note 63	N	The TOE is not distributed. Therefore, DTLS requirements do not apply.
TD0547: NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	Y	
TD0551: NIT Technical Decision for Incomplete Mappings of OEs in FW Module v1.4+Errata	Y	
TD0555: NIT Technical Decision for RFC Reference incorrect in TLSS Test	N	The TOE does not implement TLS.
TD0556: NIT Technical Decisions for RFC 5077 question	N	The TOE does not implement TLS.
TD0563: NIT Technical Decision for Clarification of audit date information	Y	
TD0564: NIT Technical Decision for Vulnerability Analysis Search Criteria	Y	
TD0569: NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7	N	The TOE does not implement TLS.
TD0570: NIT Technical Decision for Clarification about FIA_AFL.1	Y	
TD0571: NIT Technical Decision for Guidance on how to handle FIA_AFL.1	Y	
TD0572: NIT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	N	The TOE does not implement TLS.
TD0580: NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	Y	
TD0581: NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	Y	
TD0591: NIT Technical Decision for Virtual TOEs and hypervisors	Y	
TD0592: NIT Technical Decision for Local Storage of Audit Records	Y	

Technical Decision	Applicable (Y/N)	Exclusion Rationale (if applicable)
TD0595: Administrative corrections to IPS PP-Module	Y	
TD0631: NIT Technical Decision for Clarification of public key authentication for SSH Server	Y	
TD0632: NIT Technical Decision for Consistency with Time Data for vNDs	Y	
TD0635: NIT Technical Decision for TLS Server and Key Agreement Parameters	N	The TOE does not implement TLS.
TD0636: NIT Technical Decision for Clarification of Public Key User Authentication for SSH	N	The TOE does not claims SSH Client.
TD0638: NIT Technical Decision for Key Pair Generation for Authentication	Y	
TD0639: NIT Technical Decision for Clarification for NTP MAC Keys	Y	
TD0656: Missing EAs for VPN GW Optional Headend SFRs	N	Device is not a headend device for VPN
TD0657: IPSEC_EXT.1.6 GCM support for VPN GW	Y	
TD0670: NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	N	The TOE does not implement TLS.
TD0683: RFC 2460 to be replaced with RFC 8200	Y	
TD0722: IPS_SBD_EXT.1.1 EA Correction	Y	
TD0723: Correction to ECDSA Curve Selection	Y	
TD0738: NIT Technical Decision for Link to Allowed-With List	Y	
TD0771: Correction to FIA_PSK_EXT.3 EA	N	TOE does not claim SFR FIA_PSK_EXT.3
TD0790: NIT Technical Decision: Clarification Required for testing IPv6	N	The TOE does not implement TLS.
TD0792: NIT Technical Decision: FIA_PMG_EXT.1 - TSS EA not in line with SFR	Y	
TD0800: Updated NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	Y	

3 Security Problem Definition

The security problem definition has been taken directly from the claimed PP and any relevant EPs/Modules/Packages specified in Section 2.2 and is reproduced here for the convenience of the reader. The security problem is defined in terms of the threats that the TOE is expected to address, assumptions about the operational environment, and any Organizational Security Policies (OSPs) that the TOE is expected to enforce.

3.1 Threats

Threats applicable to the TOE are drawn from [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2]. The threat statements are given in Table 3. Each one states explicitly from which source it is drawn.

Table 3 – Threats

ID	Threat
T.UNAUTHORIZED_ADMINISTRATOR_ACCESS (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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

ID	Threat
	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 (Drawn from [CPP_ND_V2.2E])	Threat agents may attempt to provide a compromised update of the software or firmware which undermines the security functionality of the device. Non-validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.
T.UNDETECTED_ACTIVITY (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	Threat agents may compromise credentials and device data enabling continued access to the Network Device and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or device credentials for use by the attacker.
T.PASSWORD_CRACKING (Drawn from [CPP_ND_V2.2E])	Threat agents may be able to take advantage of weak administrative passwords to gain privileged access to the device. Having privileged access to the device provides the attacker unfettered access to the network traffic and may allow them to take advantage of any trust relationships with other Network Devices.
T.SECURITY_FUNCTIONALITY_FAILURE (Drawn from [CPP_ND_V2.2E])	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.
T.NETWORK_DISCLOSURE (Drawn from [MOD_CPP_FW_V1.4E])	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. **RATIONALE**: [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2] offer alternative wordings for the threat description. Effectively, they are identical as they refer to an attacker monitoring the traffic to and from the TOE to determine potentially sensitive information that may be used for attacking the target system in the protected network. Therefore, it is sufficient to include the wording as in [MOD_CPP_FW_V1.4E].
T.NETWORK_ACCESS (Drawn from [MOD_IPS_V1.0])	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

ID	Threat
	communicate with devices on the protected network via a backdoor then those devices may be susceptible to the unauthorized disclosure of information. **RATIONALE**: [MOD_CPP_FW_V1.4E] offers an alternative wording. That wording states a threat which is a subset of the wording of the threat stated in [MOD_IPS_V1.0]. The broader wording is included as it shall ensure that the threat is covered to the full extent required by [MOD_IPS_V1.0] as well as [MOD_CPP_FW_V1.4E]. Furthermore, the wording of the threat statement in [MOD_IPS_V1.0] is effectively identical to the corresponding statement in [MOD_VPNGW_V1.2]. Including the wording as in [MOD_CPP_FW_V1.4E] would not fully address the threat as stated in [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2].
T.NETWORK_MISUSE (Drawn from [MOD_CPP_FW_V1.4E])	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. **RATIONALE**: [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2] offer alternative wordings for the threat description. Effectively, they are identical as they refer to the services available in a protected network being used in a manner not allowed by the security policy. Therefore, it is sufficient to include the wording as in [MOD_CPP_FW_V1.4E].
T.MALICIOUS_TRAFFIC (Drawn from [MOD_CPP_FW_V1.4E])	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.
T.NETWORK_DOS (Drawn from [MOD_IPS_V1.0])	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.DATA_INTEGRITY (Drawn from [MOD_VPNGW_V1.2])	Devices on a protected network may be exposed to threats presented by devices located outside the protected network, which 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 within the communications may be susceptible to a loss of integrity.
T.REPLAY_ATTACK (Drawn from [MOD_VPNGW_V1.2])	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:

ID	Threat
	• 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 included in Table 4 are drawn directly from [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2]. For each assumption, the source of the statement is explicitly stated.

Table 4 – Assumptions

ID	Assumption
A.PHYSICAL_PROTECTION (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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. **Application note**: Revised in accordance with TD0591.

ID	Assumption
A.NO_THRU_TRAFFIC_PROTECTION (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.
A.RESIDUAL_INFORMATION (Drawn from [CPP_ND_V2.2E])	The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.
A.VS_TRUSTED_ADMINISTRATOR (Drawn from [CPP_ND_V2.2E])	The Security Administrators for the VS are assumed to be trusted and to act in the best interest of security for the organization. This includes not interfering with the correct operation of the device. The Network Device is not expected to be capable of defending against a malicious VS Administrator that actively works to bypass or compromise the security of the device.

ID	Assumption
A.VS_REGULAR_UPDATES (Drawn from [CPP_ND_V2.2E])	The VS software is assumed to be updated by the VS Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.VS_ISOLATION (Drawn from [CPP_ND_V2.2E])	For vNDs, it is assumed that the VS provides, and is configured to provide sufficient isolation between software running in VMs on the same physical platform. Furthermore, it is assumed that the VS adequately protects itself from software running inside VMs on the same physical platform.
A.VS_CORRECT_CONFIGURATION (Drawn from [CPP_ND_V2.2E])	For vNDs, it is assumed that the VS and VMs are correctly configured to support ND functionality implemented in VMs.
A.CONNECTIONS (Drawn from [MOD_VPNGW_V1.2] and [MOD_IPS_V1.0])	It is assumed that 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. **RATIONALE**: The statement of assumption is not included in [CPP_ND_V2.2E] or [MOD_FW_V1.4E]. The statement of the assumption is identical in [MOD_VPNGW_V1.2] and [MOD_IPS_V1.0].

3.3 Organizational Security Policies

The OSPs included in Table 5 are drawn directly from [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2]. For each OSP, the source of the statement is explicitly stated.

Table 5 – OSPs

ID	OSP
P.ACCESS_BANNER (Drawn from [CPP_ND_V2.2E])	The TOE shall display an initial banner describing restrictions of use, legal agreements, or any other appropriate information to which users consent by accessing the TOE.
P.ANALYZE (Drawn from [MOD_IPS_V1.0])	Analytical processes and information to derive conclusions about potential intrusions must be applied to IPS data and appropriate response actions taken.

4 Security Objectives

The security objectives have been taken directly from the claimed PP and any relevant EPs/Modules/Packages and are reproduced here for the convenience of the reader.

4.1 Security Objectives for the TOE

The security objectives in Table 6 apply to the TOE. As [CPP_ND_V2.2E] does explicitly state any security objectives for the TOE, the statements are drawn from [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2]. For each security objective, the source of the statement is explicitly stated.

Table 6 – Security Objectives

ID	Security Objectives
O.RESIDUAL_INFORMATION (Drawn from [MOD_CPP_FW_V1.4E])	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 (Drawn from [MOD_CPP_FW_V1.4E])	The TOE shall perform stateful traffic filtering on network packets that it processes. 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).
O.ADDRESS_FILTERING (Drawn from [MOD_VPNGW_V1.2])	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) and/or receiving (destination) applicable network traffic as well as on established connection information.
O.AUTHENTICATION (Drawn from [MOD_VPNGW_V1.2])	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 (Drawn from [MOD_VPNGW_V1.2])	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 a cryptographic capabilities. These capabilities are intended to maintain confidentiality and

ID	Security Objectives
	allow for detection and modification of data that is transmitted outside of the TOE.
O.FAIL_SECURE (Drawn from [MOD_VPNGW_V1.2])	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 (Drawn from [MOD_VPNGW_V1.2])	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) and/or receiving (destination) port (or service) identified in the network traffic as well as on established connection information.
O.SYSTEM_MONITORING (Drawn from [MOD_VPNGW_V1.2] and [MOD_IPS_V1.0])	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). The IPS must collect and store information about all events that may indicate an IPS policy violation related to misuse, inappropriate access, or malicious activity on monitored networks. **RATIONALE**: A different security objective statement with an identical ID has been provided in [MOD_VPNGW_V1.2] and [MOD_IPS_V1.0]. The two statements do not overlap as one is specific to the system monitoring capabilities of the VPN functions of the TOE and the other one to the system monitoring capabilities of the IPS functions of the TOE. Therefore, both wordings are merged to the statement of O.SYSTEM_MONITORING. This ensures that the security objective is fully enforced by the TOE as intended in both sources.
O.TOE_ADMINISTRATION (Drawn from [MOD_VPNGW_V1.2] and [MOD_IPS_V1.0])	TOEs will provide the functions necessary for an administrator to configure the packet filtering rules, **as well as the administrator-defined IPS policies** and the cryptographic aspects of the IPsec protocol that are enforced by the TOE. **RATIONALE**: The statement of security objective with an identical ID in [MOD_IPS_V1.0] concerns with the IPS providing an authorized administrator with the method to configure the administrator-defined IPS policies. That

ID	Security Objectives
	statement explicitly requires the TOE to allow Administrator to define the IPS policies which is excluded from [MOD_VPNGW_V1.2]. The statement in this Security Target merges the two by refining the statement of O.TOE_ADMINISTRATION taken from [MOD_VPNGW_V1.2] with the text added in bold font.
O.IPS_ANALYZE (Drawn from [MOD_IPS_V1.0])	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 (Drawn from [MOD_IPS_V1.0])	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.TRUSTED_COMMUNICATIONS (Drawn from [MOD_IPS_V1.0])	The IPS will ensure that communications between distributed components of the TOE are not subject to unauthorized modification or disclosure.

4.2 Security Objectives for the Operational Environment

Security objectives for the operational environment assist the TOE in correctly providing its security functionality. These objectives, which are found in the table below, track with the assumptions about the TOE operational environment.

Table 7 – Security Objectives for the Operational Environment

ID	Objectives for the Operational Environment
OE.PHYSICAL (Drawn from [CPP_ND_V2.2E])	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.
OE.NO_GENERAL_PURPOSE (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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.

ID	Objectives for the Operational Environment
OE.TRUSTED_ADMN (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	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 (Drawn from [CPP_ND_V2.2E])	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment. For vNDs, this applies when the physical platform on which the VM runs is removed from its operational environment.
OE.VM_CONFIGURATION (Drawn from [CPP_ND_V2.2E])	For vNDs, the Security Administrator ensures that the VS and VMs are configured to Reduce the attack surface of VMs as much as possible while supporting ND functionality (e.g., remove unnecessary virtual hardware, turn off unused inter-VM communications mechanisms), and Correctly implement ND functionality (e.g., ensure virtual networking is properly configured to support network traffic, management channels, and audit reporting).
OE.CONNECTIONS (Drawn from [MOD_VPNGW_V1.2])	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. **RATIONALE**: The security objective with the same ID stated in [MOD_IPS_V1.0] is effectively identical to the statement in [MOD_VPNGW_V1.2]. Therefore, it is sufficient that the statement from [MOD_VPNGW_V1.2] s included.

5 Security Requirements

This section identifies the Security Functional Requirements (SFRs) for the TOE. The SFRs included in this section are derived from Part 2 of the Common Criteria for Information Technology Security Evaluation, Version 3.1, Revisions 5, September 2017, all applicable international interpretations and from [CPP_ND_V2.2E], [MOD_CPP_FW_V1.4E], [MOD_IPS_V1.0] and [MOD_VPNGW_V1.2]. If not otherwise stated in an application note, a statement of SFR is drawn from [CPP_ND_V2.2E]. Each extended security functional requirement is defined in the cPP or EP from which it is drawn.

Table 8 – SFRs

Requirement	Description
FAU_GEN.1	Audit Data Generation
FAU_GEN.1/IPS	Audit Data Generation (IPS)
FAU_GEN.2	User Identity Association
FAU_STG.1	Protected Audit Trail Storage
FAU_STG_EXT.1	Protected Audit Event Storage
FCS_CKM.1	Cryptographic Key Generation
FCS_CKM.1/IKE	Cryptographic key generation (for IKE Peer Authentication)
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_NTP_EXT.1	NTP Protocol
FCS_IPSEC_EXT.1	IPsec Protocol
FCS_RBG_EXT.1	Random Bit Generation
FCS_SSHS_EXT.1	SSH Server Protocol
FDP_RIP.2	Full Residual Information Protection
FFW_RUL_EXT.1	Stateful Traffic Filtering
FFW_RUL_EXT.2	Stateful Filtering of Dynamic Protocols
FIA_AFL.1	Authentication Failure Management
FIA_PMG_EXT.1	Password Management
FIA_PSK_EXT.1	Pre-Shared Keys
FIA_PSK_EXT.2	Generated Pre-Shared Keys
FIA_UIA_EXT.1	User Identification and Authentication
FIA_UAU_EXT.2	Password-based Authentication Mechanism
FIA_UAU.7	Protected Authentication Feedback
FIA_X509_EXT.1/Rev	X.509 Certificate Validation
FIA_X509_EXT.2	X.509 Certificate Authentication

Requirement	Description
FIA_X509_EXT.3	X.509 Certificate Requests
FMT_MOF.1/Functions	Management of Security Functions Behaviour
FMT_MOF.1/ManualUpdate	Management of Security Functions Behaviour
FMT_MOF.1/Services	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
FPF_RUL_EXT.1	Rules for Packet Filtering
FPT_SKP_EXT.1	Protection of TSF Data (for reading of all pre-shared, symmetric and private keys)
FPT_APW_EXT.1	Protection of Administrator Passwords
FPT_FLS.1/SelfTest	Self-Test Failures
FPT_TST_EXT.1	TSF Testing
FPT_TST_EXT.3	TSF Self-Test with Defined Methods
FPT_STM_EXT.1	Reliable Time Stamps
FPT_TUD_EXT.1	Trusted Update
FTA_SSL.3	TSF-initiated Termination
FTA_SSL.4	User-initiated Termination
FTA_SSL_EXT.1	TSF-initiated Session Locking
FTA_TAB.1	Default TOE Access Banner
FTP_ITC.1	Inter-TSF Trusted Channel
FTP_TRP.1/Admin	Trusted Path
IPS_ABD_EXT.1	Anomaly-Based IPS Functionality
IPS_IPB_EXT.1	IP Blocking
IPS_NTA_EXT.1	Network Traffic Analysis
IPS_SBD_EXT.1	Signature-Based IPS Functionality

5.1 Conventions

The CC allows the following types of operations to be performed on the functional requirements: assignments, selections, refinements, and iterations. The following conventions are used within this document to identify operations defined by CC:

- Where operations were completed in the PP and relevant EPs/Modules/Packages or other conventions are used, the formatting shall be retained;
- Extended SFRs are identified by the addition of "_EXT" in the identification of the SFR;
- Assignment: Indicated with *italicized* text;
- Refinement: Indicated with ~~overstricken~~ and **bold** text;
- A rationale for refinements is given immediately following the statement of the SFR;
- Selection: Indicated with <u>underlined</u> text; and

- Iteration: Indicated by appending the iteration identifier after a slash, e.g., /SigGen.

5.2 Security Functional Requirements

This section includes the security functional requirements for this ST.

5.2.1 Security Audit (FAU)

5.2.1.1 FAU_GEN.1 Audit Data Generation

FAU_GEN.1.1

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

a) Start-up and shut-down of the audit functions;
b) Indication that TSF self-test was completed
c) Failure of self-test
d) All auditable events for the not specified level of audit; and
e) *All administrative actions comprising:*
   - *Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).*
   - *Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).*
   - *Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).*
   - *Resetting passwords (name of related user account shall be logged).*
   - *[Cluster mode configuration,*
   - *Cluster mode management,*
   - *Kernel state synchronization of two instances of a TOE configured in Cluster Mode];*
f) *Specifically defined auditable events listed in* Table 9.

FAU_GEN.1.2

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and
b) For each audit event type, based on the auditable event definitions of the functional components included in the cPP/ST, *information specified in column three of* Table 9.

Table 9 – Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None.	None.
FAU_GEN.1/VPN	No events specified.	N/A
FAU_GEN.2	None.	None.
FAU_STG.1	None.	None.
FAU_STG_EXT.1	None.	None.
FCS_CKM.1	None.	None.
FCS_CKM.1/IKE	No events specified.	N/A
FCS_CKM.2	None.	None.

Requirement	Auditable Events	Additional Audit Record Contents
FCS_CKM.4	None.	None.
FCS_COP.1/DataEncryption	None.	None.
FCS_COP.1/SigGen	None.	None.
FCS_COP.1/Hash	None.	None.
FCS_COP.1/KeyedHash	None.	None.
FCS_NTP_EXT.1	• Configuration of a new time server • Removal of configured time server	Identity if new/removed time server
FCS_IPSEC_EXT.1	Failure to establish an IPsec SA.	Reason for failure
FCS_RBG_EXT.1	None	None.
FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure
FDP_RIP.2	None.	None.
FFW_RUL_EXT.1	Application of rules configured with the 'log' operation	Source and destination addresses Source and destination ports Transport Layer Protocol TOE Interface
FFW_RUL_EXT.2	Dynamical definition of rule Establishment of a session	None.
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded	Origin of the attempt (e.g., IP address)
FIA_PMG_EXT.1	None.	None.
FIA_UIA_EXT.1	All use of identification and authentication mechanism	Origin of the attempt (e.g., IP address)
FIA_UAU_EXT.2	All use of identification and authentication mechanism	Origin of the attempt (e.g., IP address)
FIA_UAU.7	None.	None.
FIA_X509_EXT.1/Rev	Unsuccessful attempt to validate a certificate Any addition, replacement or removal of trust anchors in the TOE's trust store	Reason for failure of certificate validation Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.2	None.	None.
FIA_X509_EXT.3	None.	None.
FIA_PSK_EXT.1	None.	None.
FIA_PSK_EXT.2	None.	None.
FMT_MOF.1/Functions	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None.

Requirement	Auditable Events	Additional Audit Record Contents
FMT_MOF.1/Services	None.	None.
FMT_MTD.1/CoreData	All management activities of TSF data	None.
FMT_MTD.1/CryptoKeys	None.	None.
FMT_SMF.1	All management activities of TSF data.	None.
FMT_SMF.1/FFW	All management activities of TSF data (including creation, modification and deletion of firewall rules).	None.
FMT_SMF.1/VPN	All administrative actions	No additional information.
FMT_SMR.2	None.	None.
FPF_RUL_EXT.1	Application of rules configured with the 'log' operation	Source and destination addresses Source and destination ports Transport Layer Protocol
FPT_FLS.1/SelfTest	No events specified.	N/A
FPT_SKP_EXT.1	None.	None.
FPT_APW_EXT.1	None.	None.
FPT_TST_EXT.1	None.	None.
FPT_TST_EXT.3	No events specified.	N/A
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism	None.
FTA_SSL.4	The termination of an interactive session	None.
FTA_SSL_EXT.1 (if "terminate the session" is selected)	The termination of a local session by the session locking mechanism	None.
FTA_TAB.1	None.	None.

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

Application note: FAU_GEN.1 combines the FAU_GEN.1 from [CPP_ND_V2.2E], FAU_GEN.1/VPN from [MOD_VPNGW_V1.2] and FAU_GEN.1/FW from [MOD_FW_V1.4E]. This is possible because the statements of the SFRs only add auditable events to the list of events. None of these requirements are contradictory and can be incorporated into a single statement of FAU_GEN.1. However, FAU_GEN.1/IPS as taken from [MOD_IPS_V1.0] includes a refinement of the statement of the SFR and therefore cannot be incorporated into the FAU_GEN.1 above and is included as a separate iteration of FAU_GEN.1 below.

Furthermore, each iteration of FAU_GEN.1 merged into the single FAU_GEN.1 above contains a reference to the auditable events applicable to the corresponding iteration of FMT_SMF.1. As a similar combination is implemented on the iterations into a single FMT_SMF.1, all references to the iterations of FMT_SMF.1 are replaced with a reference to a single FMT_SMF.1.

FTP_ITC.1 combines the FTP_ITC.1 from [CPP_ND_V2.2E] and FTP_ITC.1/VPN from [MOD_VPNGW_V1.2] None of these requirements are contradictory and can be incorporated into a single statement of FTP_ITC.1.

5.2.1.2 FAU_GEN.1/IPS Audit Data Generation (IPS)

FAU_GEN.1/IPS

The TSF shall be able to generate an **IPS** audit record of the following auditable **IPS** 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]

FAU_GEN.1.2/IPS Refinement

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*].

Table 10 – IPS Events

Requirement	Auditable Events	Additional Audit Record Contents
FMT_SMF.1	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. throughput, 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 interface mode (if applicable).

Requirement	Auditable Events	Additional Audit Record Contents
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).

Application note: Requirement FMT_SMF.1/IPS is merged with FMT_SMF.1. Therefore, reference to FMT_SMF.1/IPS is replaced with a reference to FMT_SMF.1 in the above table.

5.2.1.3 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.

5.2.1.4 FAU_STG.1 Protected Audit Trail Storage

FAU_STG.1.1

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

FAU_STG.1.2

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

5.2.1.5 FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1

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

FAU_STG_EXT.1.2

The TSF Shall be able to store generated audit data on the TOE itself. In addition [*The TOE shall consist of a single standalone component that stores audit data locally*].

FAU_STG_EXT.1.3

The TSF shall [*overwrite previous audit records according to the following rule: [oldest log entry is overwritten*]] when the local storage space for audit data is full.

5.2.2 Cryptographic Support (FCS)

5.2.2.1 FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1

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

- *RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;*
- *ECC schemes using "NIST curves" [P-256, P-384, P-521] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4;*
- *FFC Schemes using '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].~~

5.2.2.2 FCS_CKM.1 Cryptographic key generation (for IKE Peer Authentication)

FCS_CKM.1.1/IKE

The TSF shall generate **asymmetric** cryptographic keys **used for IKE peer authentication** in accordance with a specified cryptographic key generation algorithm [

- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3 for RSA schemes****;*
- ***FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4 for ECDSA schemes and implementing "NIST curves" P-384 and [P-256]****]*

and

- ***[no other key generation algorithms***]

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

Application note: From [MOD_VPNGW_V1.2] and has been updated as per TD0723

5.2.2.3 FCS_CKM.2 Cryptographic Key Establishment

FCS_CKM.2.1

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

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

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

Application Note: This SFR has been updated as per TD0580 and TD0581

5.2.2.4 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 [*
    - *instructs a part of the TSF to destroy the abstraction that represents the key*]

that meets the following: *No Standard.*

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

FCS_COP.1.1/DataEncryption

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

Application note: The wording is from [CPP_ND_V2.2E]. An alternative wording is provided in [MOD_VPNGW_V1.2] but with the operations implement in the statement the two wordings are identical in content.

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

FCS_COP.1.1/SigGen

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

- *RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [2048 bits and 4096 bits]*,
- *Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256 bits, 384 bits and 521 bits]*

]

that meet the following: [

- *For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or RSASSA-PKCS1v1_5; ISO/IEC 9796-2, Digital signature scheme 2 or Digital Signature scheme 3*,
- *For ECDSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 6 and Appendix D, Implementing "NIST curves" [P-256, P-384, P-521]; ISO/IEC 14888-3, Section 6.4*

].

5.2.2.7 FCS_COP.1/Hash Cryptographic Operations (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*.

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

FCS_COP.1.1/KeyedHash

The TSF shall perform *keyed-hash message authentication* in accordance with a specified cryptographic algorithm [*HMAC-SHA-1*, *HMAC-SHA-256*, *HMAC-SHA-384*, *HMAC-SHA-512*] and cryptographic key sizes *[160 bits, 256 bits, 384 bits and 512 bits]* **and message digest sizes [*160*, *256*, *384*, *512*] bits** that meet the following: *ISO/IEC 9797-2:2011, Section 7 "MAC Algorithm 2"*.

5.2.2.9 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 [selection:

- Authentication using [*SHA1, 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.

5.2.2.10 FCS_IPSEC_EXT.1 IPSec Protocol

FCS_IPSEC_EXT.1.1

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

FCS_IPSEC_EXT.1.2

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

FCS_IPSEC_EXT.1.3

The TSF shall implement [*tunnel mode*].

FCS_IPSEC_EXT.1.4

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

Application Note: The wording is from [CPP_ND_V2.2E]. An alternative wording is provided in [MOD_VPNGW_V1.2] but with the operations implement in the statement the two wordings are identical in content.

FCS_IPSEC_EXT.1.5

The TSF shall implement the protocol: [

- *IKEv1, using Main Mode for Phase 1 exchanges, as defined in RFCs 2407, 2408, 2409, RFC 4109,* [*no other RFCs for extended sequence numbers*]*, and* [*RFC 4868 for Hash Functions*]*;*
- *IKEv2 as defined in RFC 5996 and* [*with no support for NAT traversal*]*, and* [*RFC 4868 for hash functions*]

].

FCS_IPSEC_EXT.1.6

The TSF shall ensure the encrypted payload in the [*IKEv1*, *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)*].

TD0657 Applied

FCS_IPSEC_EXT.1.7

The TSF shall ensure that [

- *IKEv1 Phase 1 SA lifetimes can be configured by a Security Administrator based on* [
  - *length of time, where the time values can be configured within [0.05 to 24]hours;*

  ];
- *IKEv2 SA lifetimes can be configured by a Security Administrator based on* [
  - *length of time, where the time values can be configured within [0.05 to 24] hours*

  ]

].

FCS_IPSEC_EXT.1.8

The TSF shall ensure that [

- *IKEv1 Phase 2 SA lifetimes can be configured by a Security Administrator based on* [
  - *length of time, where the time values can be configured within [0.05 to 8] hours;*

  ];
- *IKEv2 Child SA lifetimes can be configured by a Security Administrator based on* [
  - *number of bytes;*
  - *length of time, where the time values can be configured within [0.05 to 8] hours;*

  ]

].

FCS_IPSEC_EXT.1.9

The TSF shall generate the secret value x used in the IKE Diffie-Hellman key exchange ("x" in g^x mod p) using the random bit generator specified in FCS_RBG_EXT.1, and having a length of at least [*224, 256 or 384*] bits.

Application note: 224-bits is for DH Group 14, 256-bits is for DH Group 19 and 384-bits is for DH Group 20.

FCS_IPSEC_EXT.1.10

The TSF shall generate nonces used in [*IKEv1*, *IKEv2*] exchanges of length [

- *according to the security strength associated with the negotiated Diffie-Hellman group;*

  ].

FCS_IPSEC_EXT.1.11

The TSF shall ensure that IKE protocols implement DH Group(s) [

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

].

Application Note: The wording is from [CPP_ND_V2.2E]. An alternative wording is provided in [MOD_VPNGW_V1.2] but with the operations implement in the statement the two wordings are identical in content.

FCS_IPSEC_EXT.1.12

The TSF shall be able to ensure by default that the strength of the symmetric algorithm (in terms of the number of bits in the key) negotiated to protect the [*IKEv1 Phase 1*, *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 [*IKEv1 Phase 2*, *IKEv2 CHILD_SA*] connection.

FCS_IPSEC_EXT.1.13

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

FCS_IPSEC_EXT.1.14

The TSF shall only establish a trusted channel if the presented identifier in the received certificate matches the configured reference identifier, where the presented and reference identifiers are of the following fields and types: [*SAN: IP address*, *SAN: Fully Qualified Domain Name (FQDN)*, *SAN: user FQDN*, *Distinguished Name (DN)] and* [*[no other reference identifier type]*].

Application Note: The wording is from [CPP_ND_V2.2E]. An alternative wording is provided in [MOD_VPNGW_V1.2] but with the operations implement in the statement the two wordings are identical in content.

5.2.2.11 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 (any)*].

FCS_RBG_EXT.1.2

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

5.2.2.12 FCS_SSHS_EXT.1 SSH Server Protocol

FCS_SSHS_EXT.1.1

The TSF shall implement the SSH protocol in accordance with: RFCs 4251, 4252, 4253, 4254, [*4344*, *5656*, *6668*].

FCS_SSHS_EXT.1.2

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

Application note: This SFR has been updated as per TD0631

FCS_SSHS_EXT.1.3

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

FCS_SSHS_EXT.1.4

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

FCS_SSHS_EXT.1.5 –

The TSF shall ensure that the SSH public-key based authentication implementation uses [*ecdsa-sha2-nistp256*, *ecdsa-sha2-nistp384*, *ecdsa-sha2-nistp521*] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6

The TSF shall ensure that the SSH transport implementation uses [*hmac-sha1*, *hmac-sha2-256*, *hmac-sha2-512*] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7

The TSF shall ensure that [*diffie-hellman-group14-sha1*, *ecdh-sha2-nistp256*] and [*ecdh-sha2-nistp384*, *ecdh-sha2-nistp521*] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8

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

5.2.3 User Data Protection (FDP)

5.2.3.1 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.

5.2.4 Firewall (FFW)

5.2.4.1 FFW_RUL_EXT.1 Stateful Traffic Filtering

FFW_RUL_EXT.1.1

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

FFW_RUL_EXT.1.2

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

- ICMPv4
  - Type
  - Code
- ICMPv6
  - Type

    - Code
- IPv4
    - Source address
    - Destination Address
    - Transport Layer Protocol
- IPv6
    - Source address
    - Destination Address
    - Transport Layer Protocol
    - [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].

Application Note: From [MOD_FW_V1.4E].

5.2.4.2 FFW_RUL_EXT.2 Stateful Filtering of Dynamic Protocols

FFW_RUL_EXT.2.1

The TSF shall dynamically define rules or establish sessions allowing network traffic to flow for the following network protocols [FTP].

Application Note: From [MOD_FW_V1.4E].

5.2.5 Identification and Authentication (FIA)

5.2.5.1 FIA_AFL.1 Authentication Failure Management

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 an Administrator defined time period has elapsed*].

5.2.5.2 FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1

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

a) Passwords shall be able to be composed of any combination of upper and lower case letters, numbers, and the following special characters: [*"!", "@", "#", "$", "%", "^", "&", "*", "(", ")", [and all other standard ASCII, extended ASCII and Unicode Characters]*]

b) Minimum password length shall be configurable to between [*10*] and [*20*] characters.

5.2.5.3 FIA_PSK_EXT.1 Pre-Shared Keys

FIA_PSK_EXT.1.1

The TSF shall be able to use pre-shared keys for IPsec and [*no other protocols*].

FIA_PSK_EXT.1.2

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

5.2.5.4 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

]

5.2.5.5 FIA_UIA_EXT.1 User Identification and Authentication

FIA_UIA_EXT.1.1

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

- Display the warning banner in accordance with FTA_TAB.1;
- [[*ICMP Echo, Establishment of an SSH session with a remote management station*]].

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.

5.2.5.6 FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1

The TSF shall provide a local [*password-based, SSH public key-based*] authentication mechanism to perform local administrative user authentication.

5.2.5.7 FIA_UAU.7.1 Protected Authentication Feedback

FIA_UAU.7.1

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

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

FIA_X509_EXT.1.2/Rev

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

5.2.5.9 FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1

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

Application Note: The wording is from [CPP_ND_V2.2E]. An alternative wording is provided in [MOD_VPNGW_V1.2] but with the operations implement in the statement the two wordings are identical in content.

FIA_X509_EXT.2.2

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

Application Note: This SFR has been updated as per TD0537.

5.2.5.10 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.

5.2.6 Security Management (FMT)

5.2.6.1 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*.

5.2.6.2 FMT_MOF.1/ManualUpdate Management of Security Functions Behavior

FMT_MOF.1.1/ManualUpdate

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

5.2.6.3 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*.

5.2.6.4 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.*

5.2.6.5 FMT_MTD.1/CryptoKeys Management of TSF Data

FMT_MTD.1.1/CryptoKeys

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

Application note: The wording is from [MOD_VPNGW_V1.2].

5.2.6.6 FMT_SMF.1 Specification of Management Functions

FMT_SMF.1.1

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

- *Ability to administer the TOE locally and remotely;*
- *Ability to configure the access banner;*
- *Ability to configure the session inactivity time before session termination or locking;*
- *Ability to update the TOE, and to verify the updates using* ***digital signature and [published hash]*** *capability prior to installing those updates;*

- *Ability to configure the authentication failure parameters for FIA_AFL.1;*
- *Definition of packet filtering rules;*
- *Association of packet filtering rules to network interfaces;*
- *Ordering of packet filtering rules by priority;*
- *Ability to configure firewall rules;*
- *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*
- *Ability to manage the trusted public keys database;*
- *Ability to manage the cryptographic keys;*
- *Ability to configure the cryptographic functionality;*
- *Ability to configure the lifetime for IPsec SAs;*
- *Ability to import X.509v3 certificates to the TOE's trust store;*
- [
    - *Ability to start and stop services;*
    - *Ability to configure audit behaviour (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full);*
    - *Ability to modify the behavior of the transmission of audit data to an external IT entity;*
    - *Ability to configure thresholds for SSH rekeying;*
    - *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;*
    - *No other capabilities*].

Application note: From [CPP_ND_V2.2E] with modifications as per [MOD_VPNGW_V1.2]. Also contains FMT_SMT.1/VPN as defined in [MOD_VPNGW_V1.2], FMT_SMF.1/FFW as defined in [MOD_FW_V1.4E] and FMT_SMF.1/IPS as defined in [MOD_IPS_V1.0] and as per TD0631.

5.2.6.7 FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1

The TSF shall maintain the roles:

- *Security Administrator.*

FMT_SMR.2.2

The TSF shall be able to associate users with roles.

FMT_SMR.2.3

The TSF shall ensure that the conditions

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

are satisfied.

5.2.7 Packet Filtering (FPF)

5.2.7.1 FPF_RUL_EXT.1 Rules for Packet Filtering

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

].

Application Note: This SFR has been updated as per TD0683.

FPF_RUL_EXT.1.3

The TSF shall allow the following operations to be associated with packet filtering rules: permit and drop with the 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.

Application note: From [MOD_VPNGW_V1.2]

5.2.8 Protection of the TSF (FPT)

5.2.8.1 FTP_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.

5.2.8.2 FPT_FLS.1 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*].

Application note: From [MOD_VPNGW_V1.2]

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

FPT_SKP_EXT.1.1

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

5.2.8.4 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*].

5.2.8.5 FPT_TST_EXT.1 TSF Testing

FPT_TST_EXT.1.1

The TSF shall run a suite of the following self-tests [*during initial start-up (on power on)*] to demonstrate the correct operation of the TSF: **noise source health test,** [*Power on test, File integrity test, Crypto integrity test, Authentication test, Algorithm known answer tests*].

Application note: The wording is from [MOD_VPNGW_V1.2].

5.2.8.6 FPT_TST_EXT.3 TSF 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].

Application note: From [MOD_VPNGW_V1.2].

5.2.8.7 FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1

The TSF shall provide *Security Administrators* the ability to query the currently executing version of the TOE firmware/software and [*no other TOE firmware/software version*].

FPT_TUD_EXT.1.2

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

FPT_TUD_EXT.1.3

The TSF shall provide means to authenticate firmware/software updates to the TOE using a **digital signature mechanism and** [*published hash*] prior to installing those updates.

Application note: The wording is from [MOD_VPNGW_V1.2].

5.2.9 TOE Access (FTA)

5.2.9.1 FTA_SSL_EXT.1 TSF-initiated Session Locking

FTA_SSL_EXT.1.1

The TSF Shall, for local interactive sessions, [

- *terminate the session*]

after a Security Administrator-specified time period of inactivity

5.2.9.2 FTA_SSL.3 TSF-initiated Termination

FTA_SSL.3.1

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

5.2.9.3 FTA_SSL.4 User-initiated Termination

FTA_SSL.4.1

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

5.2.9.4 FTA_TAB.1 Default TOE Access Banners

FTA_TAB.1.1

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

5.2.10 Trusted Path/Channels (FTP)

5.2.10.1 FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1

The TSF shall **be capable of using [*IPsec*, *SSH*] to** provide a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server, [[*VPN Peer entity, node configured in Cluster Mode*]]** that is logically distinct from other communication channels and provides assured identification of its end points and protection of the channel data from **disclosure and detection of modification of the channel data**.

FTP_ITC.1.2

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

FTP_ITC.1.3

The TSF shall initiate communication via the trusted channel for *[audit server, remote VPN gateways or peers]*.

Application note: The wording for this SFR is taken from [CPP_ND_V2.2E]. An iteration FTP_ITC.1/VPN is defined in [MOD_VPNGW_V1.2]. FTP_ITC.1/VPN iteration is not separately included because it is covered by FTP_ITC.1. The statement of FPT_ITC.1 becomes identical with the statement of FTP_ITC.1/VPN in [MOD_VPNGW_V1.2]. Therefore, FTP_ITC.1/VPN is included in FTP_ITC.1 and shall not be separately produced.

5.2.10.2 FTP_TRP.1/Admin Trusted Path

FTP_TRP.1.1/Admin

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

FTP_TRP.1.2/Admin

The TSF shall permit remote **Administrators** to initiate communication via the trusted path.

FTP_TRP.1.3/Admin

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

5.2.11 Intrusion Prevention (IPS)

5.2.11.1 IPS_ABD_EXT.1 Anomaly-Based IPS Functionality

IPS_ABD_EXT.1.1

The TSF shall support the definition of [anomaly ('unexpected') traffic patterns] including the specification of [

- throughput ([*bits per second*]);
- time of day;
- frequency;
- *thresholds*;
- [*no other methods*]]

and the following network protocol fields:

- [[*IPv4: source address; destination address*
- *IPv6: source address; destination address*
- *TCP: source port; destination port*
- *UDP: source port; destination port*]]

IPS_ABD_EXT.1.2

The TSF shall support the definition of anomaly activity through [manual configuration by administrators].

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: [
  - 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:
  - allow the traffic flow
  - block/drop the traffic flow
  - and [no other actions]

Application note: From [MOD_IPS_V1.0].

5.2.11.2 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, no other IPS policy elements].

Application note: From [MOD_IPS_V1.0].

5.2.11.3 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

The TSF shall process (be capable of inspecting) the following network traffic protocols:

- [*Internet Protocol (IPv4), RFC 791*
- *Internet Protocol version 6 (IPv6), RFC 2460*
- *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: [*FastEthernet interface: dedicated management Ethernet interface*];
- [
  - *Session-reset-capable interfaces: [Ethernet interfaces];*
  - *and no other interface types*].

Application note: From [MOD_IPS_V1.0].

5.2.11.4 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)].
- 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. [no other types of TCP payload inspection];
- UDP data: characters beyond the first 8 bytes of the UDP header;
- [*no other types of packet payload inspection*];

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]

Application note: From [MOD_IPS_V1.0].

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.

5.3 TOE SFR Dependencies Rationale for SFRs

The PP and any relevant EPs/Modules/Packages contain(s) all the requirements claimed in this ST. As such, the dependencies are not applicable since the PP has been approved.

5.4 Security Assurance Requirements

The TOE assurance requirements for this ST are taken directly from the PP and any relevant EPs/Modules/Packages, which is/are derived from Common Criteria Version 3.1, Revision 5. The assurance requirements are summarized in Table 11.

Table 11 – Security Assurance Requirements

Assurance Class	Assurance Components	Component Description
Security Target	ASE_CCL.1	Conformance claims
	ASE_ECD.1	Extended components definition
	ASE_INT.1	ST introduction
	ASE_OBJ.1	Security objectives for the operational environment
	ASE_REQ.1	Stated security requirements
	ASE_SPD.1	Security problem definition
	ASE_TSS.1	TOE Summary Specification
Development	ADV_FSP.1	Basic functionality specification
Guidance Documents	AGD_OPE.1	Operational user guidance
	AGD_PRE.1	Preparative Procedures
Life Cycle Support	ALC_CMC.1	Labelling of the TOE
	ALC_CMS.1	TOE CM coverage
Tests	ATE_IND.1	Independent testing – conformance
Vulnerability Assessment	AVA_VAN.1	Vulnerability survey

5.5 Assurance Measures

The TOE satisfied the identified assurance requirements. This section identifies the Assurance Measures applied by Juniper Networks, Inc. to satisfy the assurance requirements. The following table lists the details.

Table 12 – TOE Security Assurance Measures

SAR Component	How the SAR will be met
ADV_FSP.1	The functional specification describes the external interfaces of the TOE; such as the means for a user to invoke a service and the corresponding response of those services. The description includes the interface(s) that enforces a security functional requirement, the interface(s) that supports the enforcement of a security functional requirement, and the interface(s) that does not enforce any security functional requirements. The interfaces are described in terms of their purpose (general goal of the interface), method of use (how the interface is to be used), parameters (explicit inputs to and outputs from an interface that control the behavior of that interface), parameter descriptions (tells what the parameter is in some meaningful way), and error messages (identifies the condition that generated it, what the message is, and the meaning of any error codes).
AGD_OPE.1	The Administrative Guide provides the descriptions of the processes and procedures of how the administrative users of the TOE can securely administer the TOE using the interfaces that provide the features and functions detailed in the guidance.
AGD_PRE.1	The Installation Guide describes the installation, generation, and startup procedures so that the users of the TOE can put the components of the TOE in the evaluated configuration.
ALC_CMC.1 ALC_CMS.1	The Configuration Management (CM) documents describe how the consumer identifies the evaluated TOE. The CM documents identify the configuration items, how those configuration items are uniquely identified, and the adequacy of the procedures that are used to control and track changes that are made to the TOE. This includes details on what changes are tracked and how potential changes are incorporated.
ATE_IND.1	Vendor will provide the TOE for testing.
AVA_VAN.1	Vendor will provide the TOE for testing. Vendor will provide a document identifying the list of software and hardware components.

6 TOE Summary Specification

This chapter identifies and describes how the Security Functional Requirements identified above are met by the TOE.

The following table relates cryptographic algorithms to the protocols implemented in the TOE. The TOE acts as both sender and recipient for IPsec and only as the server for SSH in the supported protocols listed in Table 13

Table 13 – Protocol Usage of Cryptographic Algorithms

Protocol	Key Exchange	Auth	Cipher	Integrity
IKEv1	Group 14 (modp 2048) Group 19 (P-256) Group 20 (P-384)	RSA 2048 ECDSA P-256 ECDSA P-384 Pre-Shared Key	AES CBC 128 AES-CBC-192 AES CBC 256	HMAC-SHA-256-128 HMAC-SHA-384-192
IKEv2	Group 14 (modp 2048) Group 19 (P-256) Group 20 (P-384)	RSA 2048 ECDSA P-256 ECDSA P-384 Pre-Shared Key	AES CBC 128 AES-CBC-192 AES CBC 256 AES GCM 128 AES GCM 256	HMAC-SHA-256-128 HMAC-SHA-384-192
IPsec ESP	IKEv1 with optional: • Group 14 (modp 2048) • Group 19 (P-256) • Group 20 (P-384)	IKEv1	AES CBC 128 AES-CBC-192 AES CBC 256	HMAC-SHA-256-128
	IKEv2 with optional: • Group 14 (modp 2048) • Group 19 (P-256) • Group 20 (P-384)	IKEv2	AES CBC 128 AES-CBC-192 AES CBC 256 AES GCM 128 AES GCM 256	HMAC-SHA-256-128
SSHv2	Diffie-Hellman Group 14 (modp 2048) ECDH-sha2-nistp256 ECDH-sha2-nistp384 ECDH-sha2-nistp521	ECDSA P-256 ECDSA P-384 ECDSA P-521	AES CTR 128 AES CTR 256 AES CBC 128 AES CBC 256	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-512

Table 14 – TOE Summary Specification SFR Description

Requirement	TSS Description
FAU_GEN.1	The TOE implements an audit function using syslog. It generates and stores audit records for the following events as well as for each event listed in Table 9. The auditing of the IPS events is described separately under FAU_GEN.1/IPS. • Start-up and shut-down of the audit functions;

Requirement	TSS Description
	• All administrative actions comprising of administrative login and logout, including the user account, • Changes to TSF data related to configuration changes, including the information that a change occurred and what was changed, • Generating/import of, changing, or deleting of cryptographic keys. The action itself and a unique key name or key reference shall be logged, • Resetting passwords, including the name of related user account, • Cluster mode configuration, • Cluster mode management, and • Kernel state synchronization of two instances of a TOE configured in Cluster Mode. For each audit log entry, the TOE stores the date and time of the event and/or reaction, the type of the event and/or reaction, subject identity when applicable, the outcome of the event when applicable, and the additional data stated in Table 9. For cryptographic keys, the following details are recorded when keys are generated, imported, changed or deleted: • PKID: certificate id will be recorded when generating or deleting a key pair. • IKE SPI: IP address of the initiator and responder, together with the SPI, will be recorded when generating a key pair. IP address of the initiator and responder provide the unique link to the key identifier (SPI) of the key that has been destroyed in the session termination. • SSH session keys: key reference as provided by process id. • SSH key configured for SSH public key authentication: hash of the public key used for authentication. For SSH (ephemeral) session keys the PID is used as the key reference to relate the audit events on key generation and key destruction. The key destruction event is recorded as a session disconnect event. The clock function of the TOE software provides a source of date and time information used in audit timestamps. The Junos kernel provides the current time when it bootstraps the TOE VM. Once the TOE VM is started it maintains its own time using the hardware Time Stamp Counter as the clock source.
FAU_GEN.1/IPS	Auditing of IPS events is different from other events given the nature of the IPS function. The following, together with the events listed in Table 10, are considered IPS auditable events: • Start-up and shut-down of the IPS functions; • All dissimilar IPS events; • All dissimilar IPS reactions; • Totals of similar events occurring within a specified time period; and • Totals of similar reactions occurring within a specified time period. For each audit log entry, the TOE stores the date and time of the event, type of event and/or reaction, and all additional data stated in Table 10. IPS events often happen in bursts which generate a large volume of audit data during an attack. To manage the volume of log messages, the TOE implements

Requirement	TSS Description
	log suppression. Log suppression suppresses multiple instances of the same log entry occurring from the same or similar session over a period of time. IPS log suppression is enabled by default and can be customized based on source/destination addresses, number of log occurrences after which log suppression begins, maximum number of logs that log suppression can operate on, and the time after which suppressed logs are reported. Suppressed logs are reported as a single log entry containing the event information and the count of occurrences.
FAU_GEN.2	None
FAU_STG.1	Local audit logs are stored in /var/log/ in the TOE filesystem. Only successfully authenticated Security Administrator can read log files or delete log and archive files. Access is through the CLI interface or direct access to the filesystem. The syslogs are automatically deleted locally according to configurable limits on storage volume. The default maximum size is 1Gb. The default maximum size can be modified using the `set system syslog` CLI command with the `size` argument
FAU_STG_EXT.1	Syslog can be configured to store the audit logs locally or to send them to one or more syslog log servers in real time via Netconf over SSH. Local audit logs are stored in /var/log/ in the TOE filesystem. Only a Security Administrator can read, delete or archive log files through the CLI or through direct access to the filesystem. The TOE is a standalone device. The locally stored syslog files are automatically deleted according to configurable limits on storage volume. The default maximum size is 1Gb, but the size can be modified by the `set system syslog` CLI command. The TOE maintains an active log file and a number of archive files. The default number of archive files is 10 but the number is configurable to any value between 1 and 1000. When the active log file reaches its maximum size, the logging function closes the file, compresses it, and names the compressed file 'logfile.0.gz'. The TOE then opens and writes to a new active log file. When the new active log file reaches the configured maximum size, 'logfile.0.gz' is renamed 'logfile.1.gz', and the active log file is closed, compressed, and named 'logfile.0.gz'. If the maximum number of archive files is reached and the size of the active file reaches the maximum size, the oldest archive file is deleted so the current active file can be archived. A 1Gb syslog file takes approximately 0.25Gb of storage when archived. Syslog files may reach the platform specific complete storage allocated to /var filesystem. When the filesystem reaches 92% storage capacity an event is generated but the privileged eventd process can continue to use 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 and the /var filesystem storage becomes exhausted a final entry is recorded in the log reporting "No space left on device" and logging is terminated. The TOE will continue to operate but audit log generation will fail.
FCS_CKM.1	Asymmetric keys for SSH are generated in accordance with FIPS PUB 186-4, Appendix B.3.3 for RSA Schemes and Appendix B.4.2 for ECC Schemes.

Requirement	TSS Description
	The TOE's cryptographic module generates asymmetric keys. The asymmetric keys produced are: • RSA 2048, 4096 bit • ECC (P-256, P-384, P-521) • DH group 14 (2048 bits) Usage of the keys in protocols is specified in Table 13
FCS_CKM.1/IKE	Asymmetric keys are generated in accordance with FIPS PUB 186-4 for IKE with IPsec. The TOE implements all of the "shall" and "should" requirements and none of the "shall not" or "should not" from FIPS PUB 186-4 Appendix B.3.3 and B.4.2. There are no other TOE-specific extensions or processes not included in the Appendices or alternative Implementations allowances that may impact the security requirements.
FCS_CKM.2	The TOE implements Diffie-Hellman group 14 key establishment using the modulus and generator specified in Section 3 of RFC3526. Asymmetric key pair are established in accordance with Section 5.6 of NIST SP 800-56A. Usage of key agreement in protocols is specified in Table 13
FCS_CKM.4	Cryptographic keys the TOE uses are enumerated and their methods of storage and destruction are given in Table 16 (Sect. 6.2). There are no configurations that do not conform to the key destruction requirement.
FCS_COP.1/DataEncryption	The TOE implements the following cryptographic protocols with the stated methods of key exchange (KE) and the authentication, cipher and integrity algorithms. The details and CAVP validation certificate numbers are given in Table 15 (Sect. 6.1). **IKE v1** KE: DH Group 14 (modp 2048), DH Group 19 (P-256) and DH Group 20 (P-384) Authentication: RSA 2048, ECDSA P-256, ECDSA P-384, pre-shared key Cipher: AES-CBC-128, AES-CBC-192, AES CBC-256 Integrity: HMAC-SHA-256-128, HMAC-SHA-384-192 **IKE v2** KE: DH Group 14 (modp 2048), DH Group 19 (P-256) and DH Group 20 (P-384) Authentication: RSA 2048, ECDSA P-256, ECDSA P-384, pre-shared key Cipher: AES-CBC-128, AES-CBC-192, AES CBC-256, AES-GCM-128, AES-GCM-256 Integrity: HMAC-SHA-256-128, HMAC-SHA-384-192 **IPSec ESP (IKE v1)** KE: IKE v1 with optional DH Group 14 (modp 2048), DH Group 19 (P-256) and DH Group 20 (P-384) Authentication: IKE v1 Cipher: AES-CBC-128, AES-CBC-192, AES CBC-256 Integrity: HMAC-SHA-256-128 **IPSec ESP (IKE v2)** KE: IKE v2 with optional DH Group 14 (modp 2048), DH Group 19 (P-256) and DH Group 20 (P-384) Authentication: IKE v2 Cipher: AES-CBC-128, AES-CBC-192, AES CBC-256, AES-GCM-128, AES-GCM-192, AES-GCM-256

Requirement	TSS Description
	Integrity: HMAC-SHA-256-128 **SSH v2** KE: DH Group 14 (modp 2048), ECDH-sha2-nistp256, ECDH-sha2-nistp384, ECDH-sha2-nistp521 Authentication: ECDSA P-256, ECDSA P-384, ECDSA P-521 Cipher: AES-CTR-128, AES-CTR-256, AES-CBC-128, AES-CBC-256 Integrity: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512
FCS_COP.1/Hash	The TOE implements SHA-1, SHA-256, SHA-384 and SHA-512. They are used for constructing HMACs as stated in FCS_COP.1/KeyedHash, for verifying the digital signatures of the TOE software and software upgrades as described in FPT_FLS.1 and FPT_TUD_EXT.1, and for storing user passwords as described in FPT_APW_EXT.1. The TSF performs SHA-1, SHA-256 hashing for the NTP.
FCS_COP.1/KeyedHash	The TOE implements the following HMAC-algorithms: • HMAC-SHA-1 with SHA-1 and key length of 160 bits, block size of 512 bits and output MAC length of 160 bits. • HMAC-SHA-256 with SHA-256 and key length of 256 bits, block size of 512 bits and output length of 256 bits. • HMAC-SHA-384 with SHA-384 and key length of 384 bits, block size of 1024 bits and output length of 384 bits. • HMAC-SHA-512 with SHA-512 and key length of 512 bits, block size of 1024 bits and an output length of 512 bits.
FCS_COP.1/SigGen	The details of the digital signature generation and verification algorithms and their CAVP validation certificate numbers are given in Table 15 (Sect. 6.1).
FCS_NTP_EXT.1	The TSF supports time updates using NTPv3 and NTPv4. The TSF authentications update using an administrator-configured symmetric key and SHA-1, and SHA-256. The TOE rejects broadcast and multicast time updates. The TOE does not place a limit on the number of NTP time sources that can be configured.
FCS_IPSEC_EXT.1	The TOE implements IPSec in accordance with RFC 4301 in tunnel mode only. A description of the implementation of packet filtering in association with IPSec is given under FPF_RUL_EXT.1. Each packet is compared to the entries in the security policy rule set in sequential order until a rule that matches the packet is found or the end of the rule set is reached. If a matching rule is found, the action stated in that rule shall be taken. If the end of the rule set is reached, the packet is discarded. When a packet is processed by the TOE, the route is checked to see if it meets a defined security policy. If the packet meets the security policy, it is processed according to the rules of that policy. For inbound traffic, the TOE looks up the SA by using the destination IP address, security protocol, and security parameter index (SPI) value. For outbound VPN traffic, the policy invokes the SA associated with the VPN tunnel. If a packet arrives and there is not an active SA for that tunnel, the packet is dropped. The TOE will then begin to establish a tunnel, so that when the packet is resent, the SA is active. After the SA is established all subsequent packets in the session will use the IPsec tunnel.

Requirement	TSS Description

	When the network traffic is encrypted, the header information may not be readily available for the enforcement of the security policy rules. Additional configuration options are available to configure the packet filtering to a specific mode for IPsec VPN tunnels. The following modes may be defined:
	• Bypass mode. Directs traffic traversing the TOE through the stateful firewall inspection, but not through the IPsec VPN tunnel
	• Discard. Inspects and drops all packets that do not match any Permit policies.
	• Protect. Traffic is routed through an IPsec tunnel based on a combination of route lookup and Permit policy inspection.
	• Log. Logs traffic and session information for all modes.
	
	Additionally, the evaluator compared the described rules to the operation of the TOE during testing and found the description of the available SPD to be consistent with the implementation of the TOE.
	
	AES-GCM-128, AES-GCM-192, AES-GCM-256, AES-CBC-128, AES-CBC-192 and AES-CBC-256 using HMAC SHA-256 are implemented for ESP protection.
	Both IKEv1 and IKEv2 are implemented. IKEv1 as defined in RFCs 2407, 2408, 2409, RFC 4109 and RFC 4868 for hash functions; IKEv2 as defined in RFCs 5996 (with no support for NAT traversal) and RFC 4868 for hash functions. For IKEv1, only main mode is supported, while aggressive mode is not.
	The TOE implements AES-CBC-128, AES-CBC-192 and AES-CBC-256 for payload protection in IKEv1 and IKEv2, and also AES-GCM-128 and AES-GCM-256 for payload protection in IKEv2.
	In the evaluated configuration, the TOE permits configuration of the:
	• IKEv1 Phase 1 and IKEv2 SA lifetimes in terms of length of time (180 to 86,400 seconds i.e. 0.05 to 24 hours),
	• IKEv1 Phase 2 SA in terms of length of time (180 to 28,800 seconds i.e. 0.05 to 8 hours)
	• IKEv2 Child SA lifetimes in terms of (kilo)bytes (64 to 4,294,967,294) and length of time (180 to 28,800 seconds i.e. 0.05 to 8 hours).
	
	The TOE implements the following CLI commands to configure the Phase 1 lifetime in seconds:
	`set security ike proposal <name> lifetime-seconds <seconds>`
	
	Phase 2 lifetime can be configured in either kilobytes or seconds using the following commands:
	`set security ipsec proposal <name> lifetime-kilobytes <kb>`
	
	`set security ipsec proposal <name> lifetime-seconds <seconds>`
	
	The TOE implements Diffie-Hellman Groups 14, 19, 20. In the IKEv1 phase 1 and phase 2 exchanges, the TOE and peer will agree on the best DH group both can support. When the TOE receives an IKE proposal, it will select the first DH group that matches the acceptable DH groups (one or more of DH Groups 14, 19, 20). The negotiation will fail if there is no match. Similarly, when the peer initiates

Requirement	TSS Description
	the IKE protocol, the TOE will select the first match from the IKE proposal sent by the peer and the negotiation fails is no acceptable match is found. The TOE uses HMAC DRBG with SHA-256 for the generation of DH exponents and nonces. Nonces in the IKE key exchange protocol are of length 224 bits (for DH Group 14), 256 bits (for DH Group 19), 384 bits (for DH Group 20). The generation of random bits is described at FCS_RBG_EXT.1. The TOE checks the strengths of the configured IKE algorithms prior to committing a tunnel configuration. This ensures that the strength of the symmetric algorithm (128, 192 or 256 bits) negotiated to protect the IKEv1 Phase 1 or IKEv2 IKE_SA connection is greater than or equal to the strength of the symmetric algorithm negotiated to protect the IKEv1 Phase 2 or IKEv2 CHILD_SA connection. If the strength is not greater, an error is displayed, and the configuration fails. The TOE uses pre-shared keys for IPsec as described in FIA_PSK_EXT.1. The TOE uses X.509v3 certificates with RSA and ECDSA as defined in RFC 4945. Certificate Request Messages are generated in accordance with RFC 2986 when validating certificates for IPsec connections. The use of certificates is described in FIA_X509_EXT.1/Rev and FIA_X509_EXT.3.
FCS_RBG_EXT.1	The TOE generates random bits in accordance with NIST Special Publication 800-90 using HMAC_DRBG, SHA-256. The RBG does not require any configuration and is seeded from single designated primary entropy source: Junos OS credits a single designated primary entropy source: bits 2-9 of the timestamp associated with software interrupts associated with the clock0 (RANDOM_SWI_CLOCK0). The RANDOM_SWI_CLOCK0 source produces 1-byte raw samples which are bits 2-9 of the hardware high-resolution clock, where bit 0 denotes the least significant bit (lsb), bit 1 the next least significant bit, and so on. This high-resolution clock is the lower 32 bits of the Intel CPU's TSC counter.
FCS_SSHS_EXT.1	The TOE implements an SSH server in accordance with the following. Below are supported Ciphers for **SSH v2** KE: DH Group 14 (modp 2048), ECDH-sha2-nistp256, ECDH-sha2-nistp384, ECDH-sha2-nistp521 Authentication: ECDSA P-256, ECDSA P-384, ECDSA P-521 Cipher: AES-CTR-128, AES-CTR-256, AES-CBC-128, AES-CBC-256 Integrity: HMAC-SHA-1, HMAC-SHA-256, HMAC-SHA-512 **RFC 4251 (SSH Protocol Architecture)** The TOE uses a 256-bit ECDSA Host Key for SSH v2. The key is generated randomly at the initial setup of SSH and is with an overwhelming probability unique to each host. The key may be de-configured (erased) with a CLI command. The TOE presents the client with its public key which the client matches against its known_hosts list of keys. When a client connects to the TOE, the client will be able to determine if the same host key was used in previous connections, or if the key is different (per the SSHv2 protocol).

Requirement	TSS Description
	The TOE implements all mandatory algorithms and methods and may be configured to accept public-key based and/or password-based authentication. Multiple authentication mechanisms for users is not required. Port forwarding and sessions to clients are allowed. X11 forwarding is prohibited. The TOE does not accept the “none” cipher and implements AES-CBC-128, AES-CBC-256, AES-CTR-128, AES-CTR-256 for the protection of data over SSH and uses keys generated in accordance “ecdsa-sha2-nistp256, ecdsa-sha2-nistp384 and ecdsa-sha2-nistp521” for public-key based device authentication. For ciphers whose block size >= 16, the TOE rekeys every (2^32-1) bytes. The client may request a rekeying event as a valid SSHv2 message at any time and the TOE will honor this request. Re-keying of session keys can be configured using the sshd_config knob. The data-limit must be set between 51200 and 1Gbyte and the time-limit must be set within 1 and 60 minutes. The TOE will rekey based on whichever limit is reached first. When a connection is brought down, the TOE does not attempt to re-establish it. Key exchange is performed only using the supported key exchange algorithms ordered as follows: ecdh-sha2-nistp256 (RFC 5656), ecdh-sha2-nistp384 (RFC 5656), ecdh-sha2-nistp521 (RFC 5656), diffie-hellman-group14-sha1 (RFC 4253). The TOE sshd server does not support debug messages via the CLI. **RFC 4252 (SSH Authentication Protocol)** The TOE implements a timeout period of 30 seconds for authentication of the SSHv2 protocol and enforces a limit of three failed authentication attempts before sending a disconnect to the client. The TOE does not accept authentication if the requested service does not exist. Authentication requests for non-existent user names will not succeed. The TOE returns a disconnect message as it would for failed authentications. This prevents attackers from enumerating valid usernames. Authentication method "none" is not allowed. The TOE responds to it with a list of permitted authentication methods. The TOE implements public key authentication for SSHv2 session authentication. Authentication succeeds if the correct private key is used. This is verified by checking that the private key corresponds to the public key stored in the authorized_keys file on the TOE filesystem. The TOE does not require multiple authentications (public key and password) for users. The TOE also supports password authentication. Expired passwords are not supported and cannot be used for authentication. The TOE does not support the configuration of host-based authentication methods. **RFC 4253 (SSH Transport Layer Protocol)** The TOE implements the following encryption methods for SSH sessions: aes128-cbc, aes256-cbc, aes128-ctr, and aes256-ctr. Negotiation of encryption algorithms in each direction is allowed. Encryption algorithm “none” is not allowed.

Requirement	TSS Description
	The TOE reads the packet payload size in the TCP packet to determine the packet length. Packets greater than 256K bytes are dropped and the connection is terminated. Negotiation of HMAC-SHA1 in each direction for SSH transport is allowed. diffie-hellman-group14-sha1 is supported. Key re-exchange is performed when SSH_MSG_KEXINIT is received. **RFC4344 (SSH Transport Layer Encryption Modes)** The TOE implements AES128-ctr and AES256-ctr for encryption. The TOE does not implement the recommended modes AES192-ctr or 3des-ctr. None of the optional modes are implemented. **RFC5656 (SSH ECC Algorithm Integration)** The TOE implements key exchange method using a recommended curve ecdh-sha2-nistp256. The client matches the key against its known_hosts list of keys. Cryptographic hashing algorithms SHA-1, SHA-256 and SHA-384 are implemented. None of the Recommended Curves are supported. **RFC6668 (sha-2 Transport Layer Protocol)** The recommended and optional algorithms hmac-sha2-256 and hmac-sha2-512 are implemented for SSH transport.
FDP_RIP.2	The TOE reads incoming data from network interfaces and stores the assembled datagrams in a temporary data structure. After a datagram is processed, the content of the structure is cleared prior to the storing and processing of the next datagram. The TOE keeps track of the length of each datagram. When erasing the content, the data structure is padded with zeros to ensure that the entire structure is cleared prior to the accepting the next datagram. This ensures that no residual data of previously processed datagram may affect the inspection of the current datagram.
FFW_RUL_EXT.1	The TOE allows Administrator to configure the stateful packet filtering rules. The rules are applied to all network traffic processed by the TOE. The TOE is configured to associate network interfaces to IP subnets. Source IP addresses are then associated with the network interface. When the TOE boots up, it executes a suite of self-tests. Only if each self-test passes, shall the boot sequence commence. The exact boot sequence is the following: • BIOS hardware and memory checks, • Loading and initialization of the Kernel OS, • FIPS self-tests and firmware integrity tests, • The init utility is started to mount file systems, set up network cards, and to start the processes that are run on system at startup, • Internet Service Daemon, Routing Protocol Daemon and Syslog Daemon are started, Routing and forwarding tables are initialized, • Management Daemon (or MGD) is started, and

Requirement	TSS Description
	• Physical interfaces are activated. The network interfaces are only activated when all functions required for processing the datagrams are verified and loaded. This ensures that the TOE is fully operational, and the rules enforced before the physical interfaces may receive any traffic. Packet processing is controlled by a Flow Daemon. If for any reason the Flow Daemon fails, the processing of the packets will stop, and none will be forwarded. A failure in other Daemons will not prevent the Flow Daemon from enforcing the TOE security policies. Also, any failure of the Flow Daemon will stop all processing of the packets. This ensures that packets will only be processed by a correctly functioning Flow Daemon. The traffic flow is implemented by an information flow subsystem consisting of a number of modules. The modules are executed in sequence on the data and may drop or allow traffic independently to the subsequence model. The modules are summarized in the following: • IP Classification module retrieves information from packets received on a NIC, 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 detects inline attack (e.g. IP Spoofing). It monitors arriving traffic, performs predefined attack detection services, and triggers actions when an attack is discovered. • Session Lookup module performs lookups in the session table used for all interfaces based on the information on incoming packets. 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 called for packets which do not match any already established session. If a packet matches an existing session, it will be forwarded to the Security Policy module instead. Session Setup module performs the auditing of denied packets. If there exists no policy to allow traffic, datagrams that do not match any policy are dropped and not logged. Session Setup module does not create sessions for denied traffic, only for allowed traffic. • The 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 policy enforcement engine that fulfills the security requirements of the user. The Security Policy module only allows traffic if the policy rule base contains a rule explicitly allowing the traffic. • The RPD (Routing Protocol Daemon) module provides the implementations and algorithms for the routing protocols and route calculations.

Requirement	TSS Description
	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 2460 (IPv6): Source address, Destination Address, Transport Layer Protocol • RFC 793 (TCP): Source port, Destination port • RFC 768 (UDP): Source port, Destination port Conformance to these RFCs is demonstrated by protocol compliance testing by the 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. Junos implements what is referred to as an Application Layer gateway (ALG) that 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. The TOE enforces the following default reject rules with logging on all network traffic: • invalid fragments; • fragmented IP packets which cannot be re-assembled completely; • where the source address is equal to the address of the network interface where the network packet was received; • where the source address does not belong to the networks associated with the network interface where the network packet was received; • where the source address is defined as being on a broadcast network; • where the source address is defined as being on a multicast network; • where the source address is defined as being a loopback address;

Requirement	TSS Description

	• packets 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;
	• where 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;
	• with the IP options: Loose Source Routing, Strict Source Routing, or Record Route specified;
	• packets are checked for validity. "Invalid fragments" are those that violate these rules:
	  ○ No overlap
	  ○ The total fragments in one packet should not be more than 62 pieces
	  ○ The total length of merged fragments should not larger than 64k
	  ○ All fragments in one packet should arrive in 2 seconds
	  ○ The total queued fragments has limitation, depending on the platform
	  ○ The total number of concurrent fragment processing for different packet has limitations depending on platform
	The TOE can be configured to drop connection attempts after a defined number of half-open TCP connections using the Junos 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 Junos OS begins dropping connection requests from that source. Similarly, the destination threshold option allows administrators to specify the number of SYN segments received per second for a single destination IP address before Junos OS begins dropping connection requests to that destination.

FFW_RUL_EXT.2	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 the rules. Source and destination addresses, source and destination ports, transport layer protocol, and TOE Interface are recorded in each log record.
FIA_AFL.1	Security Administrators may configure the retry-options to specify the rules for handling failed user authentication attempts. The retry-options are applied following the first failed login attempt and for each username separately. The length of delay (5-10 seconds) after each failed attempt is specified by the backoff-factor, and the increase of the delay for each subsequent failed attempt is specified by the backoff-threshold (1-3). The tries-before-disconnect sets the maximum number of times (1-10) the user is allowed to attempt login over SSH before the connection is disconnected. The handling of authentication failures in SSH connection establishment is stated in FCS_SSHS_EXT.1.

Requirement	TSS Description
	Each failed attempt is tracked. When the tries-before-disconnect number is reached for any user, that user account is locked and cannot be used to authenticate remotely. The lockout-period sets the duration of account locking (1-43,200 minutes). If an account is locked, the user may login locally from the console but not remotely until the lockout period has passed.
FIA_PMG_EXT.1	The password used for user authentication is a case-sensitive, alphanumeric string. The minimum length is 10 characters and Administrator-defined maximum length of up to 20 characters. A password must contain characters from at least two different character sets (upper, lower, numeric, special). Allowed special characters are "!", "@", "#", "$", "%", "^", "&", "*", "(", and ")". Any standard ASCII, extended ASCII and Unicode characters are allowed.
FIA_PSK_EXT.1	The TOE supports IPsec pre-shared keys. It accepts Unicode characters to specify generated bit-based pre-shared keys. Unicode characters are encoded as UTF-8 and treated as multiple bytes – up to 4 bytes depending on the character. The maximum length limit for text-based pre-shared keys enforced by the TOE is 255 bytes. When a pre-shared key is only composed of ASCII characters this limit is equivalent to 255 characters. The TOE accepts pre-shared keys and converts the string into an authentication value as per RFC 2409 for IKEv1 or RFC 4306 for IKEv2, using the PRF that is configured as the hash algorithm for the IKE exchanges.
FIA_UIA_EXT.1	Security Administrators may access the TOE from console or from a remote management station over SSHv2. In both cases, the access method is the CLI. Once connected from the console or over SSH, the user is granted a logon window displaying an access banner and requiring the user to enter username and a password. The entered password is verified against the reference password for the user. The reference password is stored hashed. The password is obscured when entered, a hash computed, and the hash compared to the reference password. Successful logon occurs when the entered user name exists and the entered password matches the reference password of that user. The TOE maintains a retry counter for each user to track the number of consecutive failed authentication attempts. Upon each successful authentication, the retry counter value is reset. If the user name does not exist or the entered password does not match the stored reference password, the TOE denies the user access to the CLI and increments the retry counter value. If the retry counter has reached the maximum number of allowed consecutive, failed authentication attempts the TOE shall take the configured defensive action. Otherwise, a re-authentication is required. None of the CLI functions shall be made available to a user until successfully authenticated. The user may only establish an SSH connection (if attempting to access the TOE remotely) and read the access banner. The TOE shall respond to an ICMP Echo but not allow any other services to the user.
FIA_UAU_EXT.2 FIA_UAU.7	Junos users are configured under "system login user" and are exported to the password database '/var/etc/master.passwd'. A Junos user is therefore an entry in the password database. Each entry in the password database has fields corresponding to the attributes of "system login user", including username, (obfuscated) password and login class.

Requirement	TSS Description
	The internal architecture supporting Authentication includes an active process, associated linked libraries and supporting configuration data. The Authentication process and library are: • login() • PAM Library module Following TOE initialization, the login() process is listening for a connection at the local console. This 'login' process can be accessed through either direct connection to the local console or following successful establishment of a remote management connection over SSH, when a login prompt is displayed. This login process identifies and authenticates the user using PAM operations. The login process does two things; it first establishes that the requesting user is whom they claim to be and second provides them with an interactive Junos Command interactive command line interface (CLI). The SSH daemon supports public key authentication by looking up a public key in an authorized keys file located in the directory '.ssh' in the user's home directory (i.e. '~/.ssh/') and this authentication method will be attempted before any other if the client has a key available. The SSH daemon will ignore the authorized keys file if it or the directory '.ssh' or the user's home directory are not owned by the user or are writeable by anyone else. For password authentication, login() interacts with a user to request a username and password to establish and verify the user's identity. The username entered by the administrator at the username prompt is reflected to the screen, but no feedback to screen is provided while the entry made by the administrator at the password prompt until the Enter key is pressed. login() uses PAM Library calls for the actual verification of this data. The password is hashed and compared to the stored value, and success/failure is indicated to login(). PAM is used in the TOE to support authentication management, account management, session management and password management. Login primarily uses the session management and password management functionality offered by PAM.
FIA_X509_EXT.1/Rev	The TOE checks the validity of X.509 certificates each time a certificate is presented for IPsec authentication. The validation is by the following steps. If each step passes, the certificate is considered valid. 1. Fields subject, issuer, subjects public key, signature, basicConstraints and validity period are extracted. Absence of any of the fields causes the validation to fail. 2. The issuer is looked up in the PKI database. Absence of the issuer or the issuer certificate not having the CA:true flag set in the basicConstraints section causes the validation to fail. 3. The TOE verifies the signature. If the signature verification fails, the validation fails. 4. The TOE confirms that the current date and time is within the validity period specified in the certificate. If not, the validation fails. 5. The TOE may be configured to perform a revocation check using CRL (specified in Sect. 6.3 of RFC 5280). If the CRL fails to download, the validation fails unless the option to skip CRL checking on download failure has been set. Revocation check is performed on end-entity and intermediate certificates.

Requirement	TSS Description
	6. The TOE validates a certificate path by building a chain of at least three certificates based upon issuer and subject linkage. Each certificate in the chain is validated with steps (1) through to (5) 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. 7. The TOE determines if a certificate is a CA certificate by requiring the CA:true flag to be present in the basicConstraints section. 8. The TOE validates the extendedKeyUsage field according to the following rules: a. Server certificates presented for TLS must have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field. b. Client certificates presented for TLS must have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field. c. 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.
FIA_X509_EXT.2	The TOE requires that the configured IKE identity of the local and remote endpoints match the contents of the X.509 certificate associated with a SA endpoint. The identity may be an email address, a fully qualified domain name or an IP address. The IKE policy of the TOE must be configured by the administrator so that the TOE knows which certificate to use for authentication. If the certificate does not validate or the contents do not match the configured identity, the SA will not be established. When configuring the IKE identity of the remote endpoint the administrator must specify an email address, fully qualified domain name, or IP address that will be matched against the SAN field, or a distinguished name, in the presented certificate. If the TSF cannot establish a connection to determine the validity of a certificate, the Administrator is prompted to accept or reject the certificate.
FIA_X509_EXT.3	None (no "device-specific information" selected). The TOE generates Certificate Request Messages as specified in RFC 2986 when validating certificates for IPsec connections. Device-specific information, Common Name, Organization, Organizational Unit, Country and public key details are provided in the CSR.
FMT_MOF.1/Functions	The CLI contains all functions for the configuring the handling of the audit data. The CLI, and therefore the functions, are only available to successfully authenticated Security Administrators. The functions include transmission of audit data to an external IT entity, and local handling of the audit data.
FMT_MOF.1/ManualUpdate	The TOE allows manual upgrading of the TOE software when upgrades are available. Any software component of the TOE may be updated individually:

Requirement	TSS Description
	the ESXi hypervisor and Junos OS. Each package is digitally signed and shall be verified and installed as described in FPT_TUD_EXT.1.
FMT_MOF.1/Services	Most services of the TOE may not be stopped and shall automatically start at the boot up of the TOE. The Security Administrator may start and stop the forwarding of the audit files to an external syslog server, Cluster Mode operation of the TOE, and TOE Software upgrade.
FMT_MTD.1/CoreData	The TOE only allows three services prior to the identification and authentication of the Security Administrator: 1. Displaying of the access banner. This is a display only and does not contain any user input mechanism. Therefore, it does not allow any means for the non-authentic users to manipulate the TOE. 2. Responding to an ICMP Echo. Echo protocol is a simple IP layer protocol for querying the status of the TOE. It does not contain any session establishment and does not carry any payload. Therefore, the protocol cannot be used for modifying the TOE or TSF data. 3. Establishment of a SSHv2 connection between the TOE and a remote management station. SSH is an IP-layer connection between the TOE and a remote management station. It will make available to the remote administrator a shell in which the user may be identified and authenticated. All management of the TOE is through a CLI which shall only be made available to the remote user upon successful identification and authentication. SSHv2 itself cannot be used for issuing any management commands to the TOE. A subset of the CLI implements the functions for managing the TOEs trust store for holding the public key certificates. Access to the trust store is only through the CLI (i.e. only granted to successfully authenticated Security Administrators) or to trusted processes. This ensures that only authorized accesses are allowed.
FMT_MTD.1/CryptoKeys	The TOE implements a rich set of cryptographic protocols and algorithms. The users are only granted limited access to the keys directly. All cryptographic protocols and algorithms the TOE implements are listed in Table 15 (Sect. 6.1). Cryptographic keys the TOE uses together with their storage and method of destruction are listed in Table 16 (Sect. 6.2) Management of cryptographic keys is through the CLI as part of managing and configuring SSHv2, IPSec, IKEv1 and IKEv2. All key management operations occur through the CLI commands. Additionally, some long term keys used as TOE identity keys are uploaded when the TOE is initialized for use and may be destroyed by the user decommissioning the TOE - also through CLI commands.
FMT_SMF.1	The TOE implements a CLI where a command exists for each management and configuration function of the TOE. The TOE may be administered locally from console or remotely from a management station. All management functions (i.e. the entire CLI) are available to all successfully authenticated Security Administrators whether accessing the TOE locally or remotely. The Security Administrator is associated with the defined login class “security-admin”, which has the necessary permission set to permit the administrator to perform all tasks necessary to manage Junos OS in accordance with the requirements of [ND_cPP], using both the local as well as the remote administrative interface. The Security Administrator has the capability to:

Requirement	TSS Description

	• Administer the TOE locally via the serial ports on the physical device or remotely over an SSH connection.
	• Initiate a manual update of TOE software:
	  ○ Query currently executing version of TOE software (both Junos OS and underlying Wind River Linux Host OS)
	  ○ Verify update using digital signature and published hash.
	• Manage Functions:
	  ○ Transmission of audit data to an external IT entity, including Start/stop and modify the behaviour of the trusted communication channel to external syslog server (netconf over SSH) and the trusted path for remote Administrative sessions (SSH)
	  ○ Handling of audit data, including setting limits of log file size and behaviour when the maximum size threshold is hit.
	• Manage TSF data:
	  ○ Create, modify, delete administrator accounts, including configuration of authentication failure parameters
	  ○ Reset administrator passwords
	• Re-enable an Administrator account
	• Start and stop services
	• Manage crypto keys:
	  ○ SSH key generation (ecdsa, ssh-rsa)
	• Manage the trusted public keys database
	• Perform management functions:
	  ○ Configure the access banner
	  ○ Configure the session inactivity time before session termination or locking, including termination of session when serial console cable is disconnected
	  ○ Manage the TOE's trust store and designate X509.v3 certificates as trust anchors;
	  ○ Import X.509v3 certificates
	  ○ Manage cryptographic functionality, including:
	  ▪ ssh ciphers
	  ▪ hostkey algorithm
	  ▪ key exchange algorithm
	  ▪ hashed message authentication code
	  ▪ thresholds for SSH rekeying
	  ○ Set the system time
	  ○ Configure NTP
	  ○ Configure Firewall rules;
	  ○ Configure the VPN-associated cryptographic functionality;
	  ○ Definition of packet filtering rules;
	  ○ Association of packet filtering rules to network interfaces;

Requirement	TSS Description
	○ Ordering of packet filtering rules by priority; ○ Configure the IPsec functionality, including configuration of IKE lifetime-seconds (within range 180 to 86400 i.e. 0.05 to 25 hours , with default value of 180 seconds), IPsec lifetime-seconds (within range 180 to 28800 i.e. 0.05 to 8 hours, with default value of 28800 seconds), and Lifetime-kilobytes (within range 64 to 4294967294 kilobytes) and ability to configure the reference identifier for the peer; ○ 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 analysed: ▪ 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) IPS signatures; ○ Create custom IPS 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. Security Administrators are able to initiate an update of the TOE firmware if a new version of the TOE firmware is available. Updates are downloaded and applied manually (there is no automatic updating of the Junos OS).
FMT_SMR.2	The TOE implements a Security Administrator role ‘super-user’. It is the only role authorized to administer the TOE. Each user assigned to the Security Administrator role gains access to the full CLI. Each human super-user is identified and authenticated with a username and password and assigned a Security Administrator role upon successful authentication. The role assignment remains until the session is terminated.
FPT_APW_EXT.1	The TOE stores authentication data locally and protects it by three means: • Passwords stored in password files are hashed with sha-256 or sha-512, • All CLI commands implement appropriate measures to not disclose passwords when entered by the user or processed by the corresponding TOE functions, and

Requirement	TSS Description
	• Authentication data for public key-based authentication methods are stored in a directory owned by the user and typically shares the name with the user. This directory contains the files '.ssh/authorized_keys' and '.ssh/authorized_keys2' which are used for SSH public key authentication. No other users may access that directory.
FPF_RUL_EXT.1	The boot sequence of the TOE appliances also aids in establishing the securing domain and preventing tamping or bypass of security functionality. This includes ensuring the packet filtering rules cannot be bypassed during the boot sequence of the TOE. The following steps list the boot sequence for the TOE: • BIOS hardware and memory checks • Loading and initialization of the FreeBSD Kernel OS • FIPS self-tests and firmware integrity tests are executed • The init utility is started (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) • Daemon programs such as Internet Service Daemon (INETD), Routing Protocol Daemon (RPD), Syslogd are started; Routing and forwarding tables are initialized • Management Daemon (or MGD) is loaded, allowing access to management interface • Physical interfaces are active Once the interfaces are brought up, they will start to receive and send packets based on the current configuration (or not receive or send any packets if they have not been previously configured). Interfaces are brought up only after successful loading of kernel and Information Flow subsystems, and these interfaces cannot send or receive packets unless previously configured by an Administrator. The RPD (Routing Protocol Daemon) module provides the implementations and algorithms for the routing protocols and route calculations. The primary goal of the RPD is to create and maintain 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 implements a default policy which disallows all traffic through it. The default policy may not be changed but Security Administrators may define packet filtering rules which allow explicitly defined traffic. Each distinct network interface may be assigned a different set of rules. The security policy rule set is an ordered list of entries stating the firewall rules. Each entry contains a specification of a network flow and an action. The action may be to permit, discard or log the traffic. The protocol fields which may be used for specifying the network flow to which the action is to be applied are the following: • **IPv4 (RFC 791)** • **source address**

Requirement	TSS Description
	• **destination address** • **protocol** • **IPv6 (RFC 8200)** • **source address** • **destination address** • **next header (protocol)** • **TCP (RFC 793)** • **source port** • **destination port** • **UDP (RFC768)** • **source port** Conformance to these RFCs is demonstrated by protocol compliance testing by the product QA team. Each packet is compared to the entries in the security policy rule set in sequential order until a rule that matches the packet is found or the end of the rule set is reached. If a matching rule is found, the action stated in that rule shall be taken. If the end of the rule set is reached, the packet is discarded. When a packet is processed by the TOE, the route is checked to see if it meets a defined security policy. If the packet meets the security policy, it is processed according to the rules of that policy. When the network traffic is encrypted, the header information may not be readily available for the enforcement of the security policy rules. Additional configuration options are available to configure the packet filtering to a specific mode for IPsec VPN tunnels. The following modes may be defined: • Bypass mode. Directs traffic traversing the TOE through the stateful firewall inspection, but not through the IPsec VPN tunnel • Discard. Inspects and drops all packets that do not match any Permit policies. • Protect. Traffic is routed through an IPsec tunnel based on a combination of route lookup and Permit policy inspection. • Log. Logs traffic and session information for all modes. For inbound traffic, the TOE looks up the SA by using the destination IP address, security protocol, and security parameter index (SPI) value. For outbound VPN traffic, the policy invokes the SA associated with the VPN tunnel. If a packet arrives and there is not an active SA for that tunnel, the packet is dropped. The TOE will then begin to establish a tunnel, so that when the packet is resent, the SA is active. After the SA is established all subsequent packets in the session will use the IPsec tunnel. The following protocols are not supported and will be dropped before the packet is matched to an ACL; therefore, any “permit” or “deny” entries won’t be captured in the logs. • IPv4- none. • IPv6 - Protocols 43 (IPv6-Route), 44 (IPv6-Frag), 51 (AH), 60 (IPv6-Opts)
FPT_FLS.1/SelfTest	The TOE implements fail-safety mechanisms to be taken in case of any of the self-tests (FPT_TST_EXT.1) fails.

Requirement	TSS Description
	If encountering a transiently corrupt state or a failure condition, the event will be logged, and the system shall cease processing any network traffic and restart. When the TOE restarts, the boot process shall re-execute all self-tests and shall not complete without each test passing. Any failed self-test shall halt the TOE and transition to an error state. In an error state the TOE shall not accept any command line input or traffic to any network interface. Power cycle is required to attempt to return to operation.
FPT_SKP_EXT.1	The CLI does not include commands or other mechanisms for viewing the cryptographic keys. The keys are protected by kernel-level file access rights. The rights are set up to limit access to the contents of cryptographic key containers to processes with cryptographic rights and to shell users with root permission. Security Administrators do not have root permission in shell.
FPT_STM_EXT.1	The TOE allows the Security Administrator to set the system time. The TOE implements a real time system clock which may be used for time stamps when the date and time is required. The system clock may also be used as a source of clock cycles which may be counted to implement inactivity timers. The time can be manually updated by a Security Administrator or automatically updated using NTP synchronization.
FPT_TST_EXT.1	When powered on, the TOE runs the following self-tests to check the correct operation: • **Power on test** to determine that the boot-device responds, and to check the memory size to confirm the amount of available memory. • **File integrity test** to assert the integrity of the mounted signed packages. Integrity of the firmware is verified by digital signature verification (FPT_TST_EXT.3) and regenerating the fingerprints on the executables and other immutable files and by comparing them to the SHA1 fingerprints stored in the manifest file. • **Crypto integrity test** to verify the integrity of CSPs, including SSH hostkeys and iked credentials (CAs, certificates, cryptographic keys). • **Authentication error test** to verify that veriexec is enabled and operates correctly using /opt/sbin/kats/cannot-exec.real. • **Kernel, Libmd, OpenSSL, Quicksec, SSH and IPsec tests** to verify correct output from known answer tests for the algorithms. • Noise source health tests to verify the correct operation of the noise source. Tests include a repetitive count test and an adaptive proportion test. Each Junos OS firmware image includes fingerprints of executables and other immutable files. The TOE validates each binary against a registered fingerprint prior to execution This ensures that the TOE is protected from undetected injection of unauthorized software and ensures the integrity of the TOE software. Only authorized executables are allowed to run which ensures the correct operation of the TOE.
FPT_TST_EXT.3	When the TOE boots up, it implements a File Integrity Test (see FPT_TST_EXT.1) to verify the integrity of the executable files. The integrity test uses ECDSA (P-256) digital signature function defined in FCS_COP.1/SigGen.
FPT_TUD_EXT.1	Users may query the version of the TOE firmware using the CLI command `show version`. if a new version of the TOE firmware is available at the developer

Requirement	TSS Description
	web site, Security Administrator may execute a firmware upgrade. Upgrades are downloaded and installed manually. Automated upgrade is not supported. Partial upgrades are supported as the ESXi hypervisor and Junos OS software may each be upgraded separately. Each upgrade is associated to a digitally signature which is verified prior to installation. The authenticity of the signature may be verified by validating the associated X.509 certificate. The signature of the package is verified at the beginning of the installation before the expansion of the package. If the signature verification fails, an error message is displayed, and the package is not installed. Once the upgraded package is loaded, the Administrator shall disable the loading of additional VMs. The TOE will reboot at the completion the installation. Upgrading commences with the installation of the Junos OS. If the kernel installation fails, an error log message will be output to the screen and the TOE will halt for administrator intervention. An audit event is not generated as the Audit function is not yet running. The Administrator will be aware of the failure due to the system halt. Successful Junos OS installation will be followed by the VM installation. The Junos OS kernel maintains a set of fingerprints (SHA1 digests) for executable files and other files which should be immutable. The manifest file is signed using the Juniper package signing key and is verified by the TOE using the corresponding public key. The verification key is stored on the TOE filesystem in clear. Access to it is controlled by filesystem access rights. ECDSA (P-256) with SHA-256 is used for digital signature package verification. The fingerprint loader will only process a manifest for which it can successfully verify the digital signature. Without a valid digital signature an executable cannot be run. 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 are not correctly verified, the TOE uses the last known verified image. When software updates are made available, an administrator can obtain, verify the integrity of the software by manually verifying the hash of the downloaded software with the hash published on the website, and install those updates. The updates can be downloaded from https://support.juniper.net/support/downloads/?p=vsrx3 . During the execution of the image, an integrity check will be performed. Only if the hash is correct, will the image be installed.
FTA_SSL.3 FTA_SSL.4	Session termination, both local and remote, may be due to the user issuing an `exit` or `quit` command or by the inactivity timer triggering the termination of a session. When the user issues an `exit` or `quit` command, the TOE makes the current session inactive and all content inaccessible. Successful authentication is required for re-gaining access.
FTA_SSL_EXT.1	Security Administrators may configure the session inactivity time for session termination. The TOE maintains for each user a counter of clock cycles since last activity. The clock cycles are read from the system clock. The counter is reset on each activity on the user's session. When the counter reaches the number of clock

Requirement	TSS Description
	cycles equal to the configured period of inactivity the user session is terminated. To terminate a session, the TOE exits the display device to the login prompt.
FTA_TAB.1	Security Administrators may access the TOE from console or from a remote management station over SSH. In both cases, the access method is the CLI. The TOE allows Security Administrators to configure an access banner for the authentication prompt. The banner is displayed at the login dialogue and can provide warnings against unauthorized access to the secure switch as well as any other information that the Security Administrator wishes to communicate. As the login dialogue is identical independently of whether the TOE is accessed locally or remotely, the banner shall be displayed at both methods of access.
FTP_ITC.1	The TOE implements an SSH server to protect confidentiality and integrity of communication with a remote syslog server. The Security Administrator sets up an event trace monitor which sends event log messages by netconf over SSH to a remote syslog server. The remote audit server initiates the connection. The TOE also implements IPsec in tunnel mode which is used for two purposes: 1. When the TOE is configured in a cluster mode, the communication between the two nodes may be protected with IPsec. 2. When the TOE is configured to act as a VPN gateway, the communication between the TOE and the VPN peer may be protected with IPsec tunnel. The TOE provides secure communication by using IPSEC between itself and Audit server, and between itself and VPN Gateway. The TOE uses IPSEC protocol with X.509 certificate-based authentication. The protocols listed are consistent with those specified in the requirement.
FTP_TRP.1/Admin	The TOE allows Security Administrators to manage the TOE locally or remotely. For remote access the remote management station is required to run an SSH client. The SSH client requests an SSHv2 connection between itself and the TOE. Upon successful SSH connection, user authentication and all subsequent administration of the TOE occurs over SSH.
IPS_ABD_EXT.1	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 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 CLI command `set schedulers` and attaching them to firewall policies. Anomaly signatures based on throughput characteristics are implemented by configuring policers with a bandwidth limit and the desired signature action (discard or forward). That is done by the CLI `set firewall policer` and attaching it to any interface with the CLI 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

Requirement	TSS Description
	is executed first. If an output firewall filter is configured on the same logical interface as a policer, the firewall filter is executed first.
IPS_IPB_EXT.1	The TOE supports definition of known-good and known-bad lists of source and/or destination addresses at the firewall rule level. Address ranges are defined by creating address book entries and attaching them to firewall policies along with policy-related attributes like permit/deny etc. which will subsequently dictate how the TOE reacts to traffic matching the policy. Only authorized users assigned the Security Administrator role can access and configure the IPS policies.
IPS_NTA_EXT.1	The TOE allows selective enforcement of attack detection and prevention techniques on network traffic passing through it. Policy rules can be defined to match a section of traffic based on a zone, network, and application, and the TOE configured to take active or passive preventive actions on matching traffic. An Intrusion Detection and Prevention (IDP) policy is made up of rule bases. Each rule base contains a set of rules that specify traffic match conditions, action taken on matching traffic, and logging requirements. IDP policies may 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. Other rules will only be processed by the firewall. IPS Policies extend firewall policies to the matching for specific attacks by Source Zone, Destination Zone, Source IP, Destination IP, Source Port, Destination Port, and Protocol. Interface matching can be achieved through the use of zones. 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. Following stateful packet filtering, if a firewall policy is marked for IDP processing, the packets are processed for IPS enforcement as follows: • Fragmentation Processing: IP Fragments are reordered and reassembled. Duplicate, over/undersized, overlapping, incomplete and other invalid fragments are discarded. • Flow Module SSL Decryption: Sessions are checked for existing IP Actions. If none exist, a new session is 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 in the queue 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 for. The traffic will be inspected for attacks even if the application cannot be determined. • Protocol parsing and decoding: 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: Attack signatures are detected via Deterministic Finite Automaton (DFA) pattern matching.

<table>
<tr><th>Requirement</th><th>TSS Description</th></tr>
<tr><td></td><td>When an attack is detected the corresponding policy configured action is executed. The action may be one of the following:
No Action
Drop packet
Drop connection
Close client (send an RST packet to the client)
Close server (sends an RST packet to the server)
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 UDP traffic. Conformance to these RFCs is demonstrated by protocol compliance testing by the product QA team.
The TOE is capable of inspecting 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.
IDP management is through the CLI locally from console or remotely over an SSH connection.</td></tr>
<tr><td>IPS_SBD_EXT.1</td><td>Signatures can be defined to match any header-field value using command <code>set security idp custom-attack</code> along with the actions (allow/block), and using command <code>set security idp idp-policy</code> that defines the IDP policy the TOE enforces on matching 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 supported protocols. For TCP payload inspection, the TOE implements pre-defined attack signatures to detect FTP commands, HTTP commands and content, and SMTP states. Administrators can also define custom-attack signatures for application layer protocols using the command <code>set security idp custom-attack</code>.
The TOE implements the following pre-defined attack signatures:
<table>
<tr><th>MOD_IPS signature name</th><th>Junos screen name</th></tr>
<tr><td>IP Fragments Overlap (Teardrop attack, Bonk attack, or Boink attack)</td><td>ip tear-drop</td></tr>
<tr><td>IP source address equal to the IP destination (Land attack)</td><td>tcp land</td></tr>
<tr><td>Fragmented ICMP Traffic (e.g. Nuke attack)</td><td>icmp fragment</td></tr>
<tr><td>Large ICMP Traffic (Ping of Death attack)</td><td>icmp ping-death</td></tr>
<tr><td>TCP NULL flags</td><td>tcp tcp-no-flag</td></tr>
</table></td></tr>
</table>

Requirement	TSS Description
	<table><tr><td>**TCP SYN+FIN flags**</td><td>**tcp syn-fin**</td></tr><tr><td>**TCP FIN only flags**</td><td>**tcp fin-no-ack**</td></tr><tr><td>**UDP Bomb Attack**</td><td>**udp length-error**</td></tr><tr><td>**ICMP flooding (Smurf attack, and ping flood)**</td><td>**icmp flood**</td></tr><tr><td>**TCP flooding (e.g. SYN flood)**</td><td>**tcp syn-flood**</td></tr><tr><td>**IP protocol scanning**</td><td>**ip unknown-protocol**</td></tr><tr><td>**TCP port scanning**</td><td>**tcp port-scan**</td></tr><tr><td>**UDP port scanning**</td><td>**udp port-scan**</td></tr><tr><td>**ICMP scanning**</td><td>**icmp ip-sweep**</td></tr></table> Attack Actions are configurable on a rule by rule basis. The default action for the above is to drop the packets. To allow the packets through, the `alarm-without-drop` action can be defined using command `set security screen ids-option`. The rules can be applied to any defined interface capable of receiving network traffic. The TOE is also capable of detecting the following signatures: • TCP SYN+RST flags, by defining a 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, and • Flooding of a network (DoS attack), by the configuration of policers that allow establishing prioritization and bandwidth limits for different type of network traffic.

6.1 CAVP Algorithm Certificate Details

Each of these cryptographic algorithms have been validated as identified in the table below. Each algorithm runs on Intel® Xeon® E5-2600 v4 series, Intel® Xeon® E-2200M series CPU.

Table 15 – CAVP Algorithm Certificate References

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
AES (Encrypt, Decrypt)	AES-CBC (128, 192, 256)	A3335	Junos OS 22.2R2 Kernel	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
	AES-CTR (128, 192, 256) AES-GCM (128, 192, 256)			Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3339	Junos OS 22.2R2 Dataplane	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3343	Junos OS 22.2R2 Quicksec	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
SHS (Message Digest Generation)	SHA-1 SHA-256 SHA-384	A3335	Junos OS 22.2R2 Kernel	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
	SHA-512			Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3339	Junos OS 22.2R2 Dataplane	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3340	Junos OS 22.2R2 LibMD	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3343	Junos OS 22.2R2 Quicksec	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
HMAC (Message Authentication)	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-384 HMAC-SHA-512	A3335	Junos OS 22.2R2 Kernel	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3339	Junos OS 22.2R2 Dataplane	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3340	Junos OS 22.2R2 LibMD	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3343	Junos OS 22.2R2 Quicksec	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
DRBG (Random Bit Generation)	HMAC-SHA2-256	A3335	Junos OS 22.2R2 Kernel	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
		A3343	Junos OS 22.2R2 Quicksec	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server

Algorithm and usage	Mode(s) and keysizes Supported	Cert #	Name	Operating Environment
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
RSA KeyGen, RSA SigGen/ SigVer	n=2048, 4096	A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
ECDSA KeyGen, ECDSA SigGen/ SigVer	P-256 (SHA-256), P-384 (SHA-384), P-521 (SHA-512)	A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server
KAS-ECC-SSC	P-256, P-384, P-521	A3342	Junos OS 22.2R2 OpenSSL	Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) CPU E5-2660 v4 (Broadwell) on HP ProLiant DL380 Gen9 Server
				Junos OS 22.2R2 on VMWare ESXi v7.0 with Intel(R) Xeon(R) E-2254ML (Coffee Lake) on PacStar 451 Server

6.2 Cryptographic Key Destruction

The table below describes the key zeroization provided by the TOE and as referenced in FCS_CKM.4.

Table 16 – Storage and Destruction of Cryptographic Keys

Keys/CSPs	Purpose	Storage Location	Method of Zeroization
SSH Private Host Key	Generated with the random number generator when the SSH is first set up. Used to identify the host. ecdsa-sha2-nistp256 (ECDSA P-256, ECDSA P-384, ECDSA P-521) and/or ssh-rsa (RSA 2048)	Plaintext on the virtual disk.	When the TOE is recommissioned, the config files (including CSP files) are removed using the Linux `shred` command to wipe the persistent storage media.
SSH Private Host Key	Loaded into memory to complete session establishment	Plaintext in volatile memory.	The TOE calls `bzero()` at session termination. The hypervisor erases the released memory before it is placed in the free pool.
SSH Session Key	Session keys used with SSH, AES 128, 256, hmac-sha-1, hmac-sha2-256 or hmac-sha2-512 key (160, 256 or 512), DH Private Key (2048 or elliptic curve 256/384/521-bits)	Plaintext in volatile memory	The TOE calls `bzero()` at session termination. The hypervisor erases the released memory before it is placed in the free pool.
RNG state	Internal state and seed key of the RNG	Plaintext in volatile memory	Handled by kernel, overwritten with zeros at reboot.
IKE Private Host Key	Private authentication key used in IKE. RSA 2048, ECDSA P-256, ECDSA P-384	Plaintext in virtual disc or in flash memory.	Erased by the Administrator issuing `clear security IKE security-association` command or zeroized at rebooting the TOE. Private keys stored in flash are not zeroized unless an explicit `request system zeroize` command is executed.
IKE-SKEYID	IKE master secret used to derive IKE and IPsec ESP session keys	Plaintext in volatile memory	Erased by the Administrator issuing `clear security IKE security-association` command or zeroized at rebooting the TOE.
IKE Session Key	IKE Session keys. AES, HMAC.	Plaintext in volatile memory	Erased by the Administrator issuing `clear security IKE security-association` command or zeroized at rebooting the TOE.
ESP Session Key	ESP Session Keys. AES, HMAC.	Plaintext in volatile memory	Erased by the Administrator issuing `clear security ipsec security-`

Keys/CSPs	Purpose	Storage Location	Method of Zeroization
			`association` command or zeroized at rebooting the TOE.
IKE-DH Private Exponent	Ephemeral DH private exponent used in IKE. DH N = 224 bit, ECDH P-256, or ECDH P-384	Plaintext in volatile memory.	Erased by the Administrator issuing `clear security IKE security-association` command or zeroized at rebooting the TOE.
IKE-PSK	Pre-shared authentication key used in IKE	Hashed in virtual disc or flash memory.	Erased by Administrator issuing a `clear security IKE security-association` command or zeroized at rebooting the TOE. Keys stored in flash are not zeroized unless an Administrator issues a `request system zeroize` command.
ecdh private keys	Loaded into memory to complete key exchange in session establishment	Plaintext in volatile memory.	The TOE calls `bzero()` at session termination. The hypervisor erases the released memory before it is placed in the free pool.

7 Acronym Table

Table 17 – Acronyms

Acronym	Definition
AES	Advanced Encryption Standard
CA	Certification Authority
CBC	Cipher Block Chaining
CC	Common Criteria
CLI	Command Line Interface
CM	Configuration Management
cPP	collaborative Protection Profile
CPU	Central Processing Unit
CRL	Certificate Revocation List
CTR	Counter Mode
CVL	Component Validation List
DFA	Deterministic Finite Automaton
DH	Diffie-Hellman
DoS	Denial of Service
DTLS	Datagram Transport Layer Security
ECC	Elliptic Curve Cryptography
ECDSA	Elliptic Curve Digital Signature Algorithm
EP	Extended Package
ESP	Encapsulating Security Payload
FFC	Finite Field Cryptography
FPC	Flexible PIC Concentrator
FTP	File Transfer Protocol
GB	Giga Byte
GCM	Galois/Counter Mode
GUI	Graphical User Interface
HA	High Availability
ICMP	Internet Control Message Protocol
ID	Identity
IDP	Intrusion Detection and Prevention
IKE	Internet Key Exchange
IP	Internet Protocol
IPS	Intrusion Prevention System
IPSec	IP Security
JCP	Junos Control Plane
KASVS	Key Agreement Scheme Validation System
KDF	Key Derivation Function
KE	Key Exchange
NAT	Network Address Translation

Acronym	Definition
NDcPP	Network Device Collaborative Protection Profile
NIAP	Nation Information Assurance Partnership
NIC	Network Interface Card
NIST	National Institute in Standards and Technology
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
OS	Operating System
PIC	Physical Interface Card
PP	Protection Profile
QA	Quality Assurance
RAM	Random Access Memory
RE	Routing Engine
RFC	Request For Comments
RSA	Rivest, Shamir & Adleman
SFR	Security Functional Requirement
SHA-2	Secure Hash Algorithm 2
SNMP	Simple Network Management Protocol
SPI	Security Parameter Index
SSH	Secure Shell
SSL	Secure Socket Layer
ST	Security Target
TCP	Transmission Control Protocol
TOE	Target of Evaluation
TLS	Transport Layer Security
TSF	TOE Security Function
TSS	TOE Summary Specification
UDP	User Datagram Protocol
vCPU	Virtual CPU
VM	Virtual Machine
vNIC	Virtual NIC
VPN	Virtual Private Network
vRAM	Virtual RAM
vRE	Virtual RE