Security Target

Juniper SSR Software v7.0.1 on Juniper SSR120, SSR130, SSR1200, SSR1300, SSR1400, SSR1500, and HPE ProLiant DL345 Gen 11

ST Version: 1.0.1

Date: March 29, 2026

Prepared for:

JUNIPER NETWORKS

https://www.juniper.net

Prepared by:

TERON LABS

www.teronlabs.com

Revision History

Version	Date	Author(s)	Description of Change
1.0	March 04, 2026	Teron Labs	Certification release
1.0.1	March 29, 2026	Teron Labs	Finalization on the scope of the TOE

Contents

1 Security Target Introduction 5
1.1 Security Target Reference 5
1.2 TOE Reference 5
1.3 TOE Overview 5
1.3.1 Intended Method of Use 5
1.3.2 Major Security Features of the TOE 7
1.3.3 TOE Type 8
1.3.4 Non-TOE Hardware, Software and Firmware 8
1.3.5 Disallowed Protocols and Services 8
1.4 TOE Description 9
1.4.1 Physical Scope of the TOE 9
1.4.2 Logical Scope of the TOE 10
2 Conformance Claims 13
2.1 Statement of Conformance Claims 13
2.2 Conformance Claim Rationale 13
2.2.1 TOE Type Consistency Rationale 13
2.2.2 Security Problem Definition Consistency 14
2.2.3 Security Objective Consistency 14
2.2.4 Security Requirements Consistency 14
2.3 Technical Decisions 14
3 Security Problem Definition 17
3.1 Threats 17
3.2 Assumptions 19
3.3 Organizational Security Policies 20
4 Security Objectives 21
4.1 Security Objectives for the TOE 21
4.2 Security Objectives for the Operational Environment 21
4.3 Security Objectives Rationale 22
5 Security Requirements 23
5.1 Extended Components Definition 23
5.2 Notation and Conventions 23
5.3 Security Functional Requirements 23
5.3.1 Security Audit (FAU) 23

5.3.2 Class FCO: Communication 28
5.3.3 Class FCS: Cryptographic Support 28
5.3.4 Class FDP: User Data Protection 34
5.3.5 Class FFW: Firewall 34
5.3.6 Class FIA: Identification and Authentication 36
5.3.7 Class FMT: Security management 37
5.3.8 Class FPT: Protection of the TSF 39
5.3.9 Class FTA: TOE Access 40
5.3.10 Class FTP: Trusted Path/Channels 40
5.4 Security Assurance Requirements 41
5.5 Security Requirements Rationale 41
6 TOE Summary Specification 43
6.1 Fulfillment of the Security Functional Requirements 43
6.2 Fulfillment of the Security Assurance Requirements 59
6.3 Cryptographic Details and CAVP References 60
6.3.1 Zeroization of Cryptographic Keys and Critical Security Parameters 60
6.3.2 CAVP Certificate References 61
7 Acronyms 64

List of Tables

Table 1 Parts Included in the Physical Scope of the TOE 9
Table 2 TOE Hardware Variants 9
Table 3 Major Security Features of the TOE 10
Table 4 Technical Decisions applicable to the Base-PP and the PP-Module 14
Table 5 Threats Drawn from the Base-PP 17
Table 6 Threats Drawn from the PP-Module 18
Table 7 Assumptions 19
Table 8 OSPs 20
Table 9 Security Objectives for the TOE 21
Table 10 Security Objective for the Operational Environment 21
Table 11 Security Functional Requirements and Auditable Events 24
Table 12 Security Assurance Requirements 41
Table 13 Fulfillment of the Mandatory Security Functional Requirements 43
Table 14 Fulfillment of the Security Assurance Requirements 59
Table 15 Timing and Method of the Zeroization of Cryptographic Keys and Critical Security Parameters 61
Table 16 CAVP Certificate References 61

1 Security Target Introduction

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

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

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

The TOE and the ST claim conformance to Common Criteria CCv3.1 Revision 5. The ST claims conformance to the collaborative Protection Profile for Network Devices, Version: 2.2e, 23-March-2020 (CPP_ND_V2.2E). CPP_ND_V2.2E is the Base-PP.

The ST also claims conformance to the PP-Module for Stateful Traffic Filter Firewalls, Version 1.4 + Errata 20200625, 25-June-2020 [MOD_CPP_FW_V1.4E].

Conformance to the Base-PP and the PP-Configuration is claimed in accordance with the PP-Configuration for Network Device and Stateful Traffic Filter Firewalls, Version 1.4 +Errata20200625, 25 June 2020 (CFG_NDcPP-FW_V1.4E).

1.1 Security Target Reference

Security Target Title	Security Target Juniper SSR Software v7.0.1 on Juniper SSR120, SSR130, SSR1200, SSR1300, SSR1400, SSR1500, and HPE ProLiant DL345 Gen 11
Security Target Version	1.0.1
Security Target Date	March 29, 2026

1.2 TOE Reference

TOE Identification	Juniper SSR Software v7.0.1 on Juniper SSR120, SSR130, SSR1200, SSR1300, SSR1400, SSR1500, and HPE ProLiant DL345 Gen 11
TOE Developer	Juniper Networks
Evaluation Sponsor	Juniper Networks

1.3 TOE Overview

1.3.1 Intended Method of Use

Juniper Session Smart Routing (SSR) is a technology concept for service-centric networking. SSR is ideal for digital businesses, allowing the development of agile, secure, and resilient applications and solutions with Wide Area Network (WAN) connections.

The TOE is a family of Juniper SSR appliances. They consist of software executing on Juniper and Hewlett Packard Enterprise (HPE) branded platforms. The TOE is the entire appliance which consists of the platform and the software. The software-only SSR solution, which the user may deploy on third party platforms, is not included in the evaluation. Also, the virtual SSR product is not included in the evaluation. Each variant of the TOE is a bare metal variant.

Each TOE variant includes the same software. Each variant may be provisioned and configured by the user to be a Session Smart Router (in short: Router) or a Session Smart Conductor (in short: Conductor). The TOE configured as a Router implements the data plane and control plane functions of the TOE and performs most functions. The TOE configured as a Conductor implements a centralized management and policy engine allowing provisioning and management of several Routers. A Conductor also acts as an information aggregation repository. SSR appliances may be managed from Juniper MIST cloud, but the cloud-based management is out of scope of the evaluation.

The functioning of the TOE as Conductor and Router and the composition of a functioning TOE from Routers and Conductors renders the TOE a distributed TOE.

Administrator of the TOE may connect to both configurations of the TOE. The connection may be locally from console, remotely from a remote management station, or from a web management station. The TOE implements the Representational State Transfer (REST) Application Programming Interface (API) which is a Juniper-specific interface which allows secure connection to the Juniper devices using Remote Procedure Calls (RPC) over HTTPS.

The changes in the configuration of a Conductor may be propagated to the associated variants of the TOE configured as Routers. The connection between a remote management station and the TOE is protected with SSH. The connection between a web management station and the TOE is protected with HTTPS over TLS. The TOE may be configured to use RADIUS over TLS (RADSEC) for secure forwarding of the authentication requests.

For local administration, the administrator authenticates to the TOE with a username and a password. For administration from a remote management station, password-based authentication and public key-based authentication using pre-shared keys are supported. For administration from a web management station, X.509 certificates are used to authenticate the public keys.

Local administration and administration from a remote management station are through the Command Line Interface (CLI) the TOE implements. Administration from a web management station is using the web management interface, called WebUI, accessed from a browser running on the web management station. REST API may also be used over HTTPS to execute RPC calls on the TOE.

When a Router is associated to a Conductor, an Administrator may manage the Router remotely from the Conductor. This allows remote management of several Routers from a single Conductor. The Administrator first establishes a local or remote management connection to the Conductor. The Router and Conductor have established a secure connection between each other using SSH and pre-shared keys. Like in Figure 14 of [CPP_ND_V2.2E], there is no separate registration channel. All communication between the instances of the TOE is protected with SSH. The Administrator may issue management commands to the Conductor and those management commands are relayed to the Router over the secure connection.

The duality of the configuration and the ability to use one instance of a TOE to administer another instance renders the TOE distributed even if the software image is the same for both configurations. The TOE is like the Use Case 3 for a distributed TOE described in [CPP_ND_V2.2E].

The TOE implements all security functions of a network device. It also implements a stateful traffic filtering firewall to guard access to the protected network. A typical deployment is illustrated in Figure 1. Instances of the TOE configured as Routers are deployed in various data centers, branches, and other facilities to protect the network connection. The Routers are associated to one or more instances of TOE configured as Conductors for information

aggregation, life-cycle management, and configuration management. The Conductor may additionally be connected to other services to utilize the collected information.

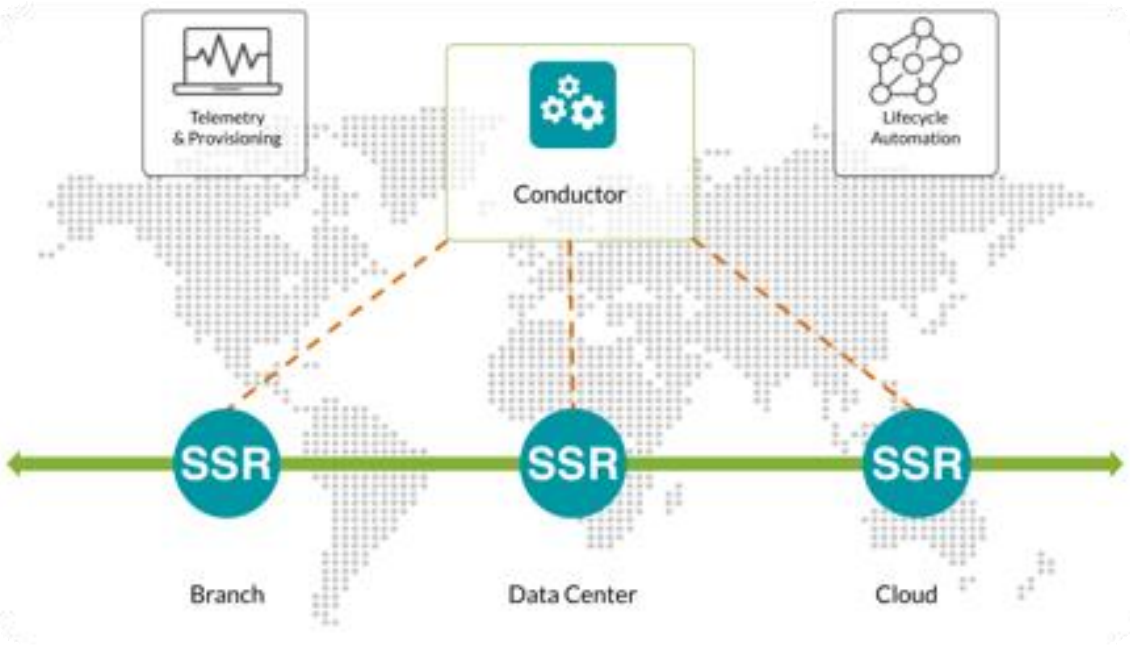

Figure 1 Representative Deployment of the TOE

The TOE is the entire appliance, including the platform hardware and the TOE software. The platforms are Juniper branded platforms Juniper SSR120, SSR130, SSR1200, SSR1300, SSR1400, SSR1500 and HPE ProLiant DL345 Gen 11. The TOE software is Juniper SSR Software v7.0.1. The TOE software is deployed in an ISO package file SSR-7.0.1-1.r1.el9.x86_64.ibu-v1.iso which includes Oracle Linux 9 operating system with kernel version 5.15.10. TOE software also includes OpenSSL Version 3.2 and GnuTLS Version 3.8.3.

The TOE implements SSH on Port 22 for remote administration of a single TOE using OpenSSL. A remote syslog server can also be connected to the TOE using TLS. GnuTLS is used for NTP authentication. Each cryptographic algorithm implemented by the TOE is validated under the Cryptographic Algorithm Validation Program (CAVP).

All software of the TOE is implemented to minimize the attack surface and to only allow the minimum number of connections with the outside users and products. Administration of the TOE is through the CLI, WebUI or REST API. The TOE implements several security mechanisms to protect itself and the critical data, and to ensure that attempts to tamper with the TOE or the data thereof are detected with a high likelihood.

1.3.2 Major Security Features of the TOE

The TOE implements a set of security functions and security mechanisms required for conformance with the Base-PP and the PP-Module. The major security features implemented by the TOE are the following:

1. Security Audit. The TOE implements an audit function to collect detailed information about the state of the TOE to allow the administrator to troubleshoot the TOE and investigate possible security-related incidents.
2. Cryptography. The TOE implements a suite of cryptographic algorithms and protocols. Each cryptographic algorithm implemented by the TOE is validated against the Cryptographic Algorithm Validation Program

(CAVP). The cryptographic algorithms and protocols are used to implement the critical security functions of the TOE but are also used for implementing the essential network security features.

3. The TOE implements trusted paths and trusted channels for secure connectivity between the TOE and remote IT systems and a remote management station. Trusted paths and trusted channels are implemented with SSH, TLS and HTTPS. The TOE implements SSH Client and SSH Server, and TLS Client and TLS Server.
4. Identification, Authentication, Authorization and Access Control. The TOE ensures that access to the administrative functions is only granted to successfully identified and authenticated users. Illegitimate users are deterred and prevented from gaining access.
5. Security Management. The TOE implements a management interface made available to the administrators. The management interface may be accessed locally from console or remotely over SSH or HTTPS. REST API over HTTPS may be used to execute RPC calls on the TOE.
6. Protection. The TOE protects itself from tampering by passive and active means to ensure that the TOE always boots into a secure state and remains so when operated.

1.3.3 TOE Type

The TOE is a non-virtual, distributed network device distributed as an appliance. The TOE resembles Use Case 3 for a distributed TOE with no dedicated registration channel as illustrated in Figure 14 of [CPP_ND_V2.2E]. The TOE implements all functions required by [CPP_ND_V2.2E] with the additional stateful packet filtering functions required by [MOD_CPP_FW_V1.4e].

1.3.4 Non-TOE Hardware, Software and Firmware

A typical deployment of the TOE is given in Figure 1. Three instances of the TOE are configured as SSR Router and one as a Conductor. The Routers are connected to the Conductor through an interconnecting network. The instances of the TOE protect the network interfaces of the premises or facility in which they are deployed. None of the networks are part of the TOE but the network connections are required for the TOE to function.

Each TOE may be administered locally or remotely. Neither the local nor the remote management workstation is part of the TOE but is required for the TOE to be administered.

The remote management workstation must connect to the TOE using SSH or HTTPS. The TOE implements a SSH Server allowing the connection, but the management workstation must implement the SSH Client for the connection. The TOE implements TSL Client and Server and HTTPS for the management from a web management station. The web management station and the HTTPS Client implementation within are not part of the TOE but are required for accessing the TOE over HTTPS.

TLS uses X.509 certificates which are verified and validated by the TOE. An Online Certificate Status Protocol (OCSP) Servers is required for secure operation of the TOE but are not part of the TOE.

The TOE connects to an external audit server for storing audit records for safe keeping and further processing. The audit server is mandatory but is not part of the TOE. it must implement a TLS Client for connecting to the TOE.

The TOE connects to a NTP server for synchronizing the clock. The NTP server is not part of the TOE but is required for the operation of the TOE.

The user may deploy additional tools for telemetry and provisioning or to the management of the TOE lifecycle. These are optional and the TOE may be administered and operated without them.

1.3.5 Disallowed Protocols and Services

The following items are specifically not included in the physical and logical scope of the TOE:

- Hardware platforms not explicitly included in the physical scope of the TOE, even if Juniper or HPE branded;
- REST API must only be used over HTTPS, not over HTTP;
- VPN Gateway and IPsec are not included in the TOE;
- Juniper SSR Software for virtual platforms are not included;
- SNMP is not considered secure and are not included;
- Intrusion Prevention System (IPS) functions are not included; and
- The Juniper MIST for the management of the TOE is not included.

1.4 TOE Description

1.4.1 Physical Scope of the TOE

The physical scope of the TOE includes all hardware and software parts and the security guidance of the TOE. The parts of the TOE included in the physical scope are detailed in Table 1.

Table 1 Parts Included in the Physical Scope of the TOE

Part of the TOE	Identification	Description
TOE Hardware	SSR120, SSR130, SSR1200, SSR1300, SSR1400, SSR1500 and HPE ProLiant DL345 Gen 11	The hardware platform and the casing of the TOE. Includes the processor, the memories, and the persistent storage.
TOE Software	Juniper SSR Software v7.0.1	The SSR-OS included in the TOE is Juniper SSR Software v7.0.1. TOE software is distributed as the following installation package: – SSR-7.0.1-1.r1.el9.x86_64.ibu-v1.iso
Security Guidance	SSR v7.0.1 Common Criteria Install and Configuration v1.0	The Common Criteria Guidance supplement for the TOE. The security guidance is distributed as a document in PDF format.

TOE Hardware is the platform on which the TOE Software is executed. Only the platforms identified in Table 1 are included in the evaluation. The technical characteristics of the hardware platforms are summarized in Table 2.

Table 2 TOE Hardware Variants

Platform	CPU	Microprocessor	Networking
Juniper SSR120	4-Core Intel Atom C3558	Denverton	1 x RJ45 Console Port 4 x 1 GbE Ethernet Ports 2 x 1 GbE RJ-45/SFP Combo Ports
Juniper SSR130	8-Core Intel Atom C3758	Denverton	1 x RJ45 Console Port 6 x 1 GbE Ethernet Ports 2 x 1 GbE RJ-45/SFP Combo Ports

Juniper SSR1200	8-Core AMD EPYC 3251	Snowy Owl	1 x RJ45 Console Port 1 x 1 GbE Management port 4 x 1/10 GbE SFP+ ports 7 x 1 GbE Ethernet ports
Juniper SSR1300	16-Core Intel Xeon Gold 6208U	Cascade Lake	1 x RJ45 Console Port 1 x 1 GbE Management port 4 x 10 GbE SFP+ ports 4 x 1/10 GbE SFP+ ports 4 x 1 GbE Ethernet ports
Juniper SSR1400	24-Core Intel Xeon Gold 6212U	Cascade Lake	1 x RJ45 Console Port 1 x 1 GbE Management port 4 x 10 GbE SFP+ ports 4 x 1/10/25 GbE SFP28 ports 4 x 1 GbE Ethernet ports
Juniper SSR1500	64-Core AMD EPYC 7713P	Milan	1 x RJ45 Console Port 1 x 1 GbE Management port 12 x 1/10/25 GbE SFP28 ports 4 x 1 GbE Ethernet ports
HPE ProLiant DL345 Gen11	AMD EPYC 9575F	EPYC	None. Choice of OCP or stand-up card, supporting NIC adapters BTO models pre-selected with a primary networking card.

TOE Software is the software which runs on the TOE Hardware. The software image is the same for each TOE Hardware variant. The software image is also the same for both TOE configurations. TOE Software is configured differently depending on whether the TOE is configured a Router or a Conductor.

TOE Security Guidance is delivered to all users of the TOE. The TOE must be deployed and operated in accordance with it. TOE Security Guidance extends the existing TOE manuals and other product literature.

1.4.2 Logical Scope of the TOE

The logical scope of the TOE includes all security functions and mechanisms required for a distributed, non-virtual TOE defined by the Base-PP and the PP-Module. The logical scope of the TOE is summarized in Table 3. Where relevant, the different functionality available in the instances of the TOE configured differently are highlighted.

Table 3 Major Security Features of the TOE

Security Feature	Description
Security Audit	The TOE implements an audit function to collect audit records of all critical security operations. The audit records allow the administrators to analyze the state of the TOE and to investigate potential security breaches. Each audit record is a log entry

	which includes all the necessary data about the event to allow detailed analysis of the audit records. The audit records are protected against unauthorized modification and may be transferred to an external syslog server for storage and further analysis. The transfer is protected by TLS.
Cryptography	The TOE implements cryptographic functions for protection of the TSF and user data, and for secure communication between the TOE and external devices. The TOE implements a random bit generator used for generating cryptographic keys and other random quantities. The TOE also implements key agreement mechanisms as well all public key cryptographic functions, symmetric cryptographic functions, secure hash functions and keyed hash-based MAC functions. Cryptographic keys and Critical Security Parameters (CSP) are destroyed by the TOE when no longer required. Each cryptographic algorithm implemented by the TOE is CAVP-validated. This fulfills the requirements of the NIAP Policy Letter #5: Applicability and Relationship of NIST Cryptographic Algorithm Validation Program (CAVP) and Cryptographic Module Validation Program (CMVP) to NIAP's Common Criteria Evaluation and Validation Scheme (CCEVS).
SSH	The TOE implements a SSH Client and a SSH Server for secure communication between instances of the TOE, and between the TOE and a remote management station. The TOE implements SSH with public-key based authentication using pre-shared keys. The public keys are stored in key containers which can only be accessed by authorized software processes. The TOE also implements SSH with password-based authentication. Multiple authentication mechanisms are not mandatory. Upon successful authentication as remote administrator, the CLI used for administering the TOE may be accessed by the Administrators from the remote management station over SSH.
HTTPS and TLS	The TOE implements a TLS Client and TLS Server using TLS v1.2. A broad range of ciphersuites is implemented to allow secure connection between the TOE and other IT devices. The IT devices the TOE may connect with over TLS include a remote syslog server, a RADIUS server, and a web management station. The web management station is accessed with HTTPS over TLS either using the WebUI or the REST API. Public Keys used for TLS session establishment are authenticated with X.509 public key certificates.
Identification, Authentication, Authorization and Access	The TOE does not implement general purpose computing facilities. The TOE implements an authentication window for each attempted connection and displays an access banner on that window. The administrator may configure the content of the banner to inform unauthorized users of the restricted nature of access and the consequences of attempted unauthorized access. Each user is identified with a username and authenticated with a password. Only upon successful identification and authentication is the user granted access to the TOE. The TOE implements protective measures against attempted password guessing. Each user is assigned a retry counter which keeps track of the number of consecutive failed authentication attempts on that user account. If the number exceeds the administrator-configurable number of consecutive failed authentication attempts, the account is locked for a period of time. Each user may

	terminate their own session and the TOE also implements an inactivity timer for each account. If the inactivity timer reaches the maximum allowed time of inactivity, the TOE terminates the session, and the user is required to re-authenticate to re-establish access.
Security Management	The TOE implements a CLI, a WebUI and a REST API for all management functions. Administrators may access the TOE and perform all management tasks. Both components of the TOE may be administered directly. Additionally, instances of the TOE configured as Routers may also be administered through the instances of the TOE configured as Conductors. This allows the administrators to remotely manage several instances of TOE through a single TOE. The CLI may be accessed locally from console or remotely over a SSH connection. The WebUI may be accessed remotely over HTTPS. The REST API of the TOE may be accessed over HTTPS. All management functions are implemented with the CLI, WebUI or REST API. There are no administrative accesses to the TOE through other means. For example, the administrators are not granted a root access to the Operating System of the TOE which would allow them to bypass the CLI or the WebUI.
Protection	The TOE protects itself by passive and active means. Passive protection is achieved through the construction of the TOE. The TOE is a dedicated appliance with restricted interfaces. It does not provide general computing capabilities. Access is restricted to authorized administrators and all administrator accesses are through a CLI, WebUI and REST API. Administrators have no root access to the underlying Linux operating system. The network interfaces are physically separate from the management ports and may not be used for administering the TOE except when used for connecting to the TOE from a remote management station. The TOE implements a set of security measures for protecting the functions it implements and the configuration parameters. The TOE also maintains a clock which is used for generating time stamps and implementing various times used in the enforcement of security functions. The TOE implements self-tests at the start-up and takes protective measures in case of a failure of self-tests. Further, the TOE also allows upgrading of the software in case of vulnerabilities being discovered in the implementation.
Firewall	The TOE implements a stateful packet filtering firewall. Administrators may define rules which are applied for filtering all traffic that passes through the TOE from one network to another. The rules may be expressed for IPv4, IPv6, ICMP, TCP and UDP traffic. Only the traffic which is explicitly declared allowed shall be forwarded by the TOE. All other traffic is dropped. Firewall rules may be applied to each network interface separately, and the administrators may define the order of the rules. The TOE traverses the rule base for each network connection and implements the first rule that matches the traffic.
Trusted Paths and Channels	The TOE implements SSH, TLS and HTTPS for secure communication with external parties. The protocols ensure that the communication between the TOE and the remote administrators is secure, that the communication between instances of the TOE is secure, and that the communication between the TOE and external IT devices is secure.

2 Conformance Claims

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

2.1 Statement of Conformance Claims

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

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

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

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

The ST claims conformance to the following Protection Profile:

- collaborative Protection Profile for Network Devices, Version: 2.2e, 23-March-2020 (CPP_ND_V2.2E).

The ST claims conformance to the following Protection Profile Modules:

- PP-Module for Stateful Traffic Filter Firewalls, Version 1.4 + Errata 20200625, 25-June-2020 (MOD_CPP_FW_V1.4E).

The ST claims conformance to the following Protection Profile Configurations:

- PP-Configuration for Network Device and Stateful Traffic Filter Firewalls, Version 1.4 +Errata20200625, 25 June 2020 (CFG_NDcPP-FW_V1.4E).

The ST claims no conformance to any Evaluation Assurance Level or any other security assurance requirement package. Security assurance requirements applicable to the TOE are those drawn from the CPP_ND_V2.2E.

The ST claims conformance to CPP_ND_V2.2E as PP-conformant.

The ST claims conformance to CFG_NDcPP-FW_V1.4E as PP-conformant.

The ST claims exact conformance to CPP_ND_V2.2E and CFG_NDcPP-FW_V1.4E Exact conformance is defined in CC and CEM addenda for Exact Conformance, Selection-Based SFRs, and Optional SFRs, CCDB-013-v2.0 Final, 2021-Sep-30.

2.2 Conformance Claim Rationale

2.2.1 TOE Type Consistency Rationale

The TOE is a non-virtual, distributed network appliance. It implements a set of security features required for exact conformance with the Base-PP and the PP-Module. No security features which require conformance to a Protection Profile Module are claimed. This ensures that the TOE Type is consistent with the TOE Type in the Base-PP and the PP-Module when applied in accordance with the PP-Configuration.

2.2.2 Security Problem Definition Consistency

The statement of the Security Problem Definition in this ST is reproduced exactly from the Base-PP and the PP-Module. Those Security Problem Definition elements which are only applicable to virtual network devices are omitted. There are no additional Security Problem Definition elements included in the statement of the Security Problem Definition. This ensures that the statement of the Security Problem Definition is consistent with PP-Configuration.

2.2.3 Security Objective Consistency

The statement of the Security Objectives in this ST is reproduced exactly from the Base-PP and the PP-Module. There are no additional Security Objectives included in the statement of the Security Objectives. Those security objectives which are only applicable to virtual network devices are omitted. This ensures that the statement of the Security Objectives is consistent with the PP-Configuration.

2.2.4 Security Requirements Consistency

The security functional requirements are drawn exactly from the Base-PP and the PP-Module. The statement of the security functional requirements includes all mandatory security requirements and those selection-based security functional requirements applicable to the TOE. The developer claims no optional requirements and does not include additional components in the statement of the security functional requirements. As such, the security functional requirements are consistently drawn from the Base-PP and the PP-Module, and the ST ensures the consistency of the security functional requirements.

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

2.3 Technical Decisions

The Technical Decisions (TD) applicable to the Base-PP and the PP-Module are given in Table 4. For each TD which is not applicable, a brief justification for the exclusion is given.

Table 4 Technical Decisions applicable to the Base-PP and the PP-Module

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 0924	NIT Technical Decision: FFW_RUL_EXT.1.2 Expected Rule Granularity Level	Yes	
TD 0827	Aligning MOD_CPP_FW_v1.4E with CPP_ND_V3.0E	Yes	
TD 0800	Updated NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	No	The ST does not claim IPsec.
TD 0792	NIT Technical Decision: FIA_PMG_EXT.1 - TSS EA not in line with SFR	Yes	
TD 0790	NIT Technical Decision: Clarification Required for testing IPv6	Yes	
TD 0738	NIT Technical Decision for Link to Allowed-With List	Yes	
TD 0670	NIT Technical Decision for Mutual and Non-Mutual Auth TLSC Testing	Yes	

TD 0639	NIT Technical Decision for Clarification for NTP MAC Keys	Yes	
TD 0638	NIT Technical Decision for Key Pair Generation for Authentication	Yes	
TD 0636	NIT Technical Decision for Clarification of Public Key User Authentication for SSH	Yes	
TD 0635	NIT Technical Decision for TLS Server and Key Agreement Parameters	Yes	
TD 0632	NIT Technical Decision for Consistency with Time Data for vNDs	No	The TOE is not a virtual TOE.
TD 0631	NIT Technical Decision for Clarification of public key authentication for SSH Server	Yes	
TD 0592	NIT Technical Decision for Local storage of Audit Records	Yes	
TD 0591	NIT Technical Decision for Virtual TOEs and hypervisors	No	The TOE is not a virtual TOE.
TD 0581	NIT Technical Decision for Elliptic curve-based key establishment and NIST SP 800-56Arev3	Yes	
TD 0580	NIT Technical Decision for clarification about use of DH14 in NDcPPv2.2e	Yes	
TD 0572	NiT Technical Decision for Restricting FTP_ITC.1 to only IP address identifiers	Yes	
TD 0571	NiT Technical Decision for Guidance on how to handle FIA_AFL.1	Yes	
TD 0570	NiT Technical Decision for Clarification about FIA_AFL.1	Yes	
TD 0569	NIT Technical Decision for Session ID Usage Conflict in FCS_DTLSS_EXT.1.7	No	The ST does not claim DTLS.
TD 0564	NiT Technical Decision for Vulnerability Analysis Search Criteria	Yes	
TD 0563	NiT Technical Decision for Clarification of audit date information	Yes	
TD 0556	NIT Technical Decision on RFC 5077 question	Yes	
TD 0555	NIT Technical Decision for RFC Reference incorrect in TLSS Test	Yes	
TD 0551	NIT Technical Decision for Incomplete Mappings of OEs in FW Module v1.4+Errata	Yes	
TD 0547	NIT Technical Decision for Clarification on developer disclosure of AVA_VAN	Yes	
TD 0546	NIT Technical Decision for DTLS - clarification of Application Note 63	No	The ST does not claim DTLS.

TD 0545	NIT Technical Decision for Conflicting FW rules cannot be configured (extension of RfI#201837)	Yes	
TD 0537	NIT Technical Decision for Incorrect reference to FCS_TLSC_EXT.2.3	Yes	
TD 0536	NIT Technical Decision for Update Verification Inconsistency	Yes	
TD 0528	NIT Technical Decision for Missing EAs for FCS_NTP_EXT.1.4	Yes	
TD 0527	Updates to Certificate Revocation Testing (FIA_X509_EXT.1)	Yes	

3 Security Problem Definition

The Security Problem Definition includes a statement of the Threats, Assumptions and OSPs applicable to the TOE. Each is stated in this section.

3.1 Threats

The threats applicable to the TOE are drawn from the Base-PP and the PP-Module. There are no additions or omissions, and the wording of each threat statement is taken verbatim. Each threat is applicable to each type of a network device. There are no threats which are only applicable to virtual or distributed network devices.

The statement of threats drawn from the Base-PP is given in Table 5.

Table 5 Threats Drawn from the Base-PP

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

	validated updates or updates validated using non-secure or weak cryptography leave the update firmware vulnerable to surreptitious alteration.
T.UNDETECTED_ACTIVITY	Threat agents may attempt to access, change, and/or modify the security functionality of the Network Device without Administrator awareness. This could result in the attacker finding an avenue (e.g., misconfiguration, flaw in the product) to compromise the device and the Administrator would have no knowledge that the device has been compromised.
T.SECURITY_FUNCTIONALITY_COMPROMISE	Threat agents may compromise credentials and device data enabling continued access to the Network Device and its critical data. The compromise of credentials includes replacing existing credentials with an attacker's credentials, modifying existing credentials, or obtaining the Administrator or device credentials for use by the attacker.
T.PASSWORD_CRACKING	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	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.

The statement of threats drawn from the PP-Module is given in Table 6.

Table 6 Threats Drawn from the PP-Module

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

3.2 Assumptions

The assumptions applicable to the TOE are drawn from the Base-PP. There are no additions or omissions, and the wording of each assumption statement is taken verbatim. Assumptions only applicable to virtual network devices are omitted. There are no additional assumptions defined in the PP-Module. The assumptions are stated in Table 7.

Table 7 Assumptions

Assumption ID	Assumption Statement
A.PHYSICAL_PROTECTION	The Network Device is assumed to be physically protected in its operational environment and not subject to physical attacks that compromise the security or interfere with the device's physical interconnections and correct operation. This protection is assumed to be sufficient to protect the device and the data it contains. As a result, the cPP does not include any requirements on physical tamper protection or other physical attack mitigations. The cPP does not expect the product to defend against physical access to the device that allows unauthorized entities to extract data, bypass other controls, or otherwise manipulate the device.
A.LIMITED_ FUNCTIONALITY	The device is assumed to provide networking functionality as its core function and not provide functionality/services that could be deemed as general purpose computing. For example, the device should not provide a computing platform for general purpose applications (unrelated to networking functionality).
A.NO_THRU_TRAFFIC_ PROTECTION	A standard/generic Network Device does not provide any assurance regarding the protection of traffic that traverses it. The intent is for the Network Device to protect data that originates on or is destined to the device itself, to include administrative data and audit data. Traffic that is traversing the Network Device, destined for another network entity, is not covered by the ND cPP. It is assumed that this protection will be covered by cPPs and PP-Modules for particular types of Network Devices (e.g., firewall).
A.TRUSTED_ ADMINISTRATOR	The Security Administrator(s) for the Network Device are assumed to be trusted and to act in the best interest of security for the organization. This includes appropriately trained, following policy, and adhering to guidance documentation. Administrators are trusted to ensure passwords/credentials have sufficient strength and entropy and to lack malicious intent when administering the device. The Network Device is not expected to be capable of defending against a malicious Administrator that actively works to bypass or compromise the security of the device.
A.REGULAR_UPDATES	The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.ADMIN_CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.
A.COMPONENTS_ RUNNING	For distributed TOEs it is assumed that the availability of all TOE components is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. It is also assumed that in addition to the availability of all components it is also checked as appropriate that the audit functionality is running properly on all TOE components.

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

3.3 Organizational Security Policies

The Organizational Security Policies (OSP) applicable to the TOE are drawn from the Base-PP. There are no additions or omissions, and the wording of each OSP statement is taken verbatim. Each OSP is applicable to each network device type. There are none which are only applicable to virtual or distributed network devices. There are no additional OSPs defined in the PP-Module

Table 8 OSPs

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

4 Security Objectives

The security objectives are stated for the TOE Sect. 4.1 and for the operational environment of the TOE in Sect. 4.2. The security objectives rationale is given in Sect. 4.3.

4.1 Security Objectives for the TOE

There are no security objectives for the TOE explicitly stated in the Base-PP. The only security objectives for the TOE are those drawn from the PP-Module. They are stated in Table 9.

Table 9 Security Objectives for the TOE

Security Objective ID	Security Objective Statement
O.RESIDUAL_ INFORMATION	The TOE shall implement measures to ensure that any previous information content of network packets sent through the TOE is made unavailable either upon deallocation of the memory area containing the network packet or upon allocation of a memory area for a newly arriving network packet or both.
O.STATEFUL_ TRAFFIC_FILTERING	The TOE shall perform stateful traffic filtering on network packets that it 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).

4.2 Security Objectives for the Operational Environment

The security objectives for the operational environment are drawn in verbatim from the Base-PP and are stated in Table 10. Those security objectives for the environment which are only applicable to virtual network devices are omitted.

There are no additional security objectives for the environment stated in the PP-Module but a clarification that OE.NO_THRU_TRAFFIC_PROTECTION only applies for the interfaces in the TOE that are defined by the Base-PP and not by the PP-Module.

Table 10 Security Objective for the Operational Environment

Security Objective ID	Security Objective Statement
OE.PHYSICAL	Physical security, commensurate with the value of the TOE and the data it contains, is provided by the environment.

OE.NO_GENERAL_ PURPOSE	There are no general-purpose computing capabilities (e.g., compilers or user applications) available on the TOE, other than those services necessary for the operation, administration and support of the TOE.
OE.NO_THRU_TRAFFIC_ PROTECTION	The TOE does not provide any protection of traffic that traverses it. It is assumed that protection of this traffic will be covered by other security and assurance measures in the operational environment.
OE.TRUSTED_ADMIN	Security Administrators are trusted to follow and apply all guidance documentation in a trusted manner.
OE.UPDATES	The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.
OE.COMPONENTS_ RUNNING	For distributed TOEs, the Security Administrator ensures that the availability of every TOE component is checked as appropriate to reduce the risk of an undetected attack on (or failure of) one or more TOE components. The Security Administrator also ensures that it is checked as appropriate for every TOE component that the audit functionality is running properly.
OE.RESDUAL_ INFORMATION	The Security Administrator ensures that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

4.3 Security Objectives Rationale

The statement of the security problem definition and the statements of the security objectives are drawn verbatim from the Base-PP and the PP-Module. Therefore, the security objectives rationales are directly applicable to the ST. They are not repeated here.

5 Security Requirements

This section states the security requirements applicable to the TOE. The statement commences with the extended components definition in Sect. 5.1. The statement of the extended components is followed by the statement of the notations and conventions used in the expression of the security requirements. The security functional requirements are stated on a per functional class basis. The mandatory security functional requirements are stated first, followed by the statement of the selection-based and optional requirements. The security assurance requirements are given in Sect. 5.4. The security requirements rationale is given in Sect. 5.5.

5.1 Extended Components Definition

The ST references several extended components. Each one is taken verbatim from the Base-PP and the PP-Module. The definitions of the extended components are not repeated here.

5.2 Notation and Conventions

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

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

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

There are no operations defined for the Security Assurance Requirements. The notation for expressing the Security Assurance Requirements is taken verbatim from the Base-PP. That ensures consistency with the PP-Configuration.

5.3 Security Functional Requirements

5.3.1 Security Audit (FAU)

FAU_GEN.1 Audit Data Generation

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

a) Start-up and shut-down of the audit functions;

b) All auditable events for the not specified level of audit; and

c) All administrative actions comprising:

- *Administrative login and logout (name of user account shall be logged if individual user accounts are required for Administrators).*
- *Changes to TSF data related to configuration changes (in addition to the information that a change occurred it shall be logged what has been changed).*
- *Generating/import of, changing, or deleting of cryptographic keys (in addition to the action itself a unique key name or key reference shall be logged).*
- *Resetting passwords (name of related user account shall be logged).*
- *[no other actions]];*

d) Specifically defined auditable events listed in Table 11.

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 11.

Table 11 Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None	None
FAU_GEN_EXT.1	None	None
FAU_GEN.2	None	None
FAU_STG_EXT.1	None	None
FAU_STG_EXT.4	None	None
FAU_STG_EXT.5	None	None
FCO_CPC_EXT.1	• Enabling communications between a pair of components. • Disabling communications between a pair of components.	Identities of the endpoint pairs enabled or disabled.
FCS_CKM.1	None	None
FCS_CKM.2	None	None

FCS_CKM.4	None	None
FCS_COP.1/DataEncryption	None	None
FCS_COP.1/SigGen	None	None
FCS_COP.1/Hash	None	None
FCS_COP.1/KeyedHash	None	None
FCS_HTTPS_EXT.1	Failure to establish a HTTPS Session	Reason for Failure
FCS_NTP_EXT.1	• Configuration of a new time server • Removal of configured time server	Identity if new/removed time server
FCS_RBG_EXT.1	None	None
FCS_SSHC_EXT.1	Failure to establish an SSH session	Reason for failure
FCS_SSHS_EXT.1	Failure to establish an SSH session	Reason for failure
FCS_SSHS_EXT.1/Peer	Failure to establish an SSH session	Reason for failure
FCS_TLSC_EXT.1	Failure the establish a TLS Session	Reason for failure
FCS_TLSC_EXT.2	None	None
FCS_TLSS_EXT.1	Failure the establish a TLS Session	Reason for failure
FIA_AFL.1	Unsuccessful login attempts limit is met or exceeded.	Origin of the attempt (e.g., IP address).
FIA_PMG_EXT.1	None	None
FIA_UIA_EXT.1	All use of identification and authentication mechanism.	Origin of the attempt (e.g., IP address)
FIA_UAU_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	• Reason for failure of certificate validation

	• Any addition, replacement or removal of trust anchors in the TOE's trust store	• Identification of certificates added, replaced or removed as trust anchor in the TOE's trust store
FIA_X509_EXT.2	None	None
FIA_X509_EXT.3	None.	None.
FMT_MOF.1/Functions	None.	None.
FMT_MOF.1/ManualUpdate	Any attempt to initiate a manual update	None
FMT_MTD.1/CoreData	None	None
FMT_MTD.1/CryptoKeys	None.	None.
FMT_SMF.1	All management activities of TSF data	None
FMT_SMR.2	None	None
FPT_SKP_EXT.1	None	None
FPT_APW_EXT.1	None	None
FPT_ITT.1	• Initiation of the trusted channel. • Termination of the trusted channel. • Failure of the trusted channel functions.	Identification of the initiator and target of failed trusted channels establishment attempt.
FPT_TST_EXT.1	None	None
FPT_TUD_EXT.1	Initiation of update; result of the update attempt (success or failure)	None.
FPT_STM_EXT.1	Discontinuous changes to time - either Administrator actuated or changed via an automated process. (Note that no continuous changes to time need to be logged. See also application note on FPT_STM_EXT.1)	For discontinuous changes to time: The old and new values for the time. Origin of the attempt to change time for success and failure (e.g., IP address).

FTA_SSL_EXT.1 (if “terminate the session" is selected)	The termination of a local interactive session by the session locking mechanism.	None.
FTA_SSL.3	The termination of a remote session by the session locking mechanism.	None
FTA_SSL.4	The termination of an interactive session.	None
FTA_TAB.1	None	None
FTP_ITC.1	Initiation of the trusted channel. 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.
FMT_MOF.1/Services	None.	None
FDP_RIP.2[1]	None.	None.
FFW_RUL_EXT.1[2]	Application of rules configured with the ‘log’ operation	Source and destination addresses Source and destination ports Transport Layer Protocol TOE Interface
FMT_SMF.1/FFW[3]	All management activities of TSF data (including creation, modification and deletion of firewall rules).	None.

FAU_GEN_EXT.1 Security Audit Data Generation for Distributed TOE Components

FAU_GEN_EXT.1.1 The TSF shall be able to generate audit records for each TOE component. The audit records generated by the TSF of each TOE component shall include the subset of security relevant audit events which can occur on the TOE component.

[1] From [MOD_CPP_FW_V1.4e]
[2] From [MOD_CPP_FW_V1.4e]
[3] From [MOD_CPP_FW_V1.4e]

FAU_GEN.2 User Identity Association

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

FAU_STG_EXT.1 Protected Audit Event Storage

FAU_STG_EXT.1.1 The TSF shall be able to transmit the generated audit data to an external IT entity using a trusted channel according to FTP_ITC.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 [*drop new audit data*] when the local storage space for audit data is full.

FAU_STG_EXT.4 Protected Local Audit Event Storage for Distributed TOEs

FAU_STG_EXT.4.1 The TSF of each TOE component which stores security audit data locally shall perform the following actions when the local storage space for audit data is full: *[*

COMPONENT	ACTIONS
Conductor	*[drop new audit data, [stop audit function]]*
Router	*[drop new audit data, [stop audit function]]*

].

FAU_STG_EXT.5 Protected Remote Audit Event Storage for Distributed TOEs

FAU_STG_EXT.5.1 Each TOE component which does not store security audit data locally shall be able to buffer security audit data locally until it has been transferred to another TOE component that stores or forwards it. All transfer of audit records between TOE components shall use a protected channel according to [*FPT_ITT.1*].

5.3.2 Class FCO: Communication

FCO_CPC_EXT.1 Component Registration Channel Definition

FCO_CPC_EXT.1.1 The TSF shall require a Security Administrator to enable communications between any pair of TOE components before such communication can take place.

FCO_CPC_EXT.1.2 The TSF shall implement a registration process in which components establish and use a communications channel that uses [

- *A channel that meets the secure channel requirements in [FPT_ITT.1]*,

for at least TSF data.

FCO_CPC_EXT.1.3 The TSF shall enable a Security Administrator to disable communications between any pair of TOE components.

5.3.3 Class FCS: Cryptographic Support

FCS_CKM.1 Cryptographic Key Generation

FCS_CKM.1.1 The TSF shall generate **asymmetric** cryptographic keys in accordance with a specified cryptographic key generation algorithm: [

- *RSA schemes using cryptographic key sizes of 2048-bit or greater that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.3;*
- *ECC schemes using 'NIST curves' [P-256, P-384, P-521] that meet the following: FIPS PUB 186-4: "Digital Signature Standard (DSS)", Appendix B.4;*
- *FFC Schemes using 'safe-prime' groups that meet the following: "NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography" and [RFC 3526].*

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

FCS_CKM.2 Cryptographic Key Establishment (Refinement)

FCS_CKM.2.1[4] 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 2, "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 [RFC 3526]*

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

FCS_CKM.4 Cryptographic Key Destruction

FCS_CKM.4.1 The TSF shall destroy cryptographic keys in accordance with a specified cryptographic key destruction method

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

  ]

that meets the following: *No Standard*.

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

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

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

FCS_COP.1.1/SigGen The TSF shall perform *cryptographic signature services (generation and verification)* in accordance with a specified cryptographic algorithm [

[4] Modified as per TD0580 and TD0581

- *RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [2048 bits, 3076 bits, 4096 bits],*
- *Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [256 bits, 384 bits, 521 bits]*

]

that meet the following: [

- *For RSA schemes: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5.5, using PKCS #1 v2.1 Signature Schemes RSASSA-PSS and/or 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*

].

FCS_COP.1/Hash Cryptographic Operation (Hash Algorithm)

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

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

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

FCS_HTTPS_EXT.1 HTTPS Protocol

FCS_HTTPS_EXT.1.1 The TSF shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2 The TSF shall implement HTTPS using TLS.

FCS_HTTPS_EXT.1.3 If a peer certificate is presented, the TSF shall [*not establish the connection*] if the peer certificate is deemed invalid.

FCS_NTP_EXT.1 NTP Protocol

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

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

- *Authentication using [SHA1] 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.

FCS_RBG_EXT.1 Random Bit Generation

FCS_RBG_EXT.1.1 The TSF shall perform all deterministic random bit generation services in accordance with ISO/IEC 18031:2011 using [*CTR_DRBG (AES)*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [*[1] software-based noise source, [2] platform-based noise source*] with a minimum of [*256 bits*] of entropy at least equal to the greatest security strength, according to ISO/IEC 18031:2011 Table C.1 “Security Strength Table for Hash Functions”, of the keys and hashes that it will generate.

FCS_SSHC_EXT.1 SSH Client Protocol

FCS_SSHC_EXT.1.1 The TSF shall implement the SSH protocol in accordance with: RFCs 4251, 4252, 4253, 4254, [*4256, 4344, 6668, 8268, 8332*].

FCS_SSHC_EXT.1.2[5] The TSF shall ensure that the SSH protocol implementation supports the following user authentication methods as described in RFC 4252: public key-based, [*no other method*].

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

FCS_SSHC_EXT.1.4 The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: [*aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, aes256-gcm@openssh.com*].

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [*rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, ecdsa-sha2-nistp521*] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHC_EXT.1.6 The TSF shall ensure that the SSH transport implementation uses [*hmac-sha2-256, hmac-sha2-512, implicit*] as its data integrity MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHC_EXT.1.7 The TSF shall ensure that [*ecdh-sha2-nistp256*] and *[diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp384, ecdh-sha2-nistp521*] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHC_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.

FCS_SSHC_EXT.1.9 The TSF shall ensure that the SSH client authenticates the identity of the SSH server using a local database associating each host name with its corresponding public key and [*no other methods*] as described in RFC 4251 section 4.1.

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, [*4256, 4344, 6668, 8268, 8332*].

FCS_SSHS_EXT.1.2[6] 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*].

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

FCS_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-ctr, aes256-ctr*].

[5] As per TD0636
[6] As per TD0631

FCS_SSHS_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [*ssh-rsa, rsa-sha2-256, rsa-sha2-512*] 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-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 [*ecdh-sha2-nistp256*] and *[diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, 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.

FCS_SSHS_EXT.1/Peer SSH Server Protocol - Inter-TOE Peer Connection

FCS_SSHS_EXT.1.1/Peer The TSF shall implement the SSH protocol in accordance with: RFCs 4251, 4252, 4253, 4254, [*4256, 4344, 6668, 8268, 8332*].

FCS_SSHS_EXT.1.2/Peer[7] The TSF shall ensure that the SSH protocol implementation supports the following user authentication methods as described in RFC 4252: public key-based, [*no other method*].

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

FCS_SSHS_EXT.1.4/Peer The TSF shall ensure that the SSH transport implementation uses the following encryption algorithms and rejects all other encryption algorithms: [*aes128-ctr, aes256-ctr*].

FCS_SSHS_EXT.1.5/Peer The TSF shall ensure that the SSH public-key based authentication implementation uses [ssh-rsa, *rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp384*] as its public key algorithm(s) and rejects all other public key algorithms.

FCS_SSHS_EXT.1.6/Peer The TSF shall ensure that the SSH transport implementation uses [*hmac-sha2-256, hmac-sha2-512*] as its MAC algorithm(s) and rejects all other MAC algorithm(s).

FCS_SSHS_EXT.1.7/Peer The TSF shall ensure that [*diffie-hellman-group14-sha1*] and *[diffie-hellman-group14-sha256*] are the only allowed key exchange methods used for the SSH protocol.

FCS_SSHS_EXT.1.8/Peer 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.

FCS_TLSC_EXT.1 TLS Client Protocol Without Mutual Authentication

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

- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384*
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*
- *TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5288*
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*
- *TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5288*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*

[7] As per TD0631

- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*
- *TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 as defined in RFC 5246*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*
- *TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4492*
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4492*
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492*

] and no other ciphersuites.

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches [*the reference identifier per RFC 6125 section 6*].

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

- *Not implement any administrator override mechanism*

].

FCS_TLSC_EXT.1.4 The TSF shall [*present the Supported Elliptic Curves/Supported Group Extensions with the following curves/groups: [secp256r1, secp384r1, secp521r1], and no other curves/groups*] in the Client Hello.

FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication

FCS_TLSC_EXT.2.1 The TSF shall support TLS communication with mutual authentication using X.509v3 certificates.

FCS_TLSS_EXT.1 TLS Client Server Without Mutual Authentication

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

- *TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5246*
- *TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA as defined in RFC 4496*
- *TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA as defined in RFC 4496*
- *TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289*
- *TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA as defined in RFC 4492*
- *TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA as defined in RFC 4492*

] and no other ciphersuites.

FCS_TLSS_EXT.1.2 The TSF shall deny connections from clients requesting SSL 2.0, SSL 3.0, TLS 1.0, and [*TLS 1.1*].

FCS_TLSS_EXT.1.3 The TSF shall perform key establishment for TLS using [*RSA with key sizes [2048 bits, 3072 bits, 409 bits], Diffie-Hellman parameters with size [2048 bits, 3072 bits, 4096 bits] Diffie-Hellman groups [no other groups], ECDHE curves [secp256r1, secp384r1, secp521r1] and no other curves*]].

FCS_TLSS_EXT.1.4 The TSF shall support [*session resumption based on session IDs according to RFC 4346 (TLS1.1) or RFC 5246 (TLS1.2), session resumption based on session tickets according to RFC 5077]*.

5.3.4 Class FDP: User Data Protection

FDP_RIP.2 Full Residual Information Protection

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

5.3.5 Class FFW: Firewall

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, [Echo identifier]]'].*

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 [counting] packets which are invalid fragments;

b) The TSF shall drop and be capable of [counting] 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 [counted, logged].*

5.3.6 Class FIA: Identification and Authentication

FIA_AFL.1 Authentication Failure Management

FIA_AFL.1.1 The TSF shall detect when an Administrator configurable positive integer within [*1 to 65535*] 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]*.

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 [*8*] and [*15*] characters.

FIA_UAU.7 Protected Authentication Feedback

FIA_UAU.7.1 The TSF shall provide only *obscured feedback* to the administrative user while the authentication is in progress **at the local console**.

FIA_UAU_EXT.2 Password-based Authentication Mechanism

FIA_UAU_EXT.2.1 The TSF shall provide a local [*password-based*] authentication mechanism to perform local administrative user authentication.

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;
- [[*Response to an ICMP Echo, Establishment of a SSH connection on Port 22 between the TOE and a remote management station, Establishment of a HTTPS connection between the TOE and 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.

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 [*the Online Certificate Status Protocol (OCSP) as specified in RFC 6960*].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
    - *Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.*
    - *Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.*
    - *Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.*
    - *OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.*

FIA_X509_EXT.1.2/Rev *The TSF shall only treat a certificate as a CA certificate if the basicConstraints extension is present and the CA flag is set to TRUE.*

FIA_X509_EXT.2 X.509 Certificate Authentication

FIA_X509_EXT.2.1 The TSF shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [*HTTPS, TLS*] and [*no additional uses*].

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

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 [*device-specific information, Common Name, Organization, Organizational Unit, Country*].

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

5.3.7 Class FMT: Security management

FMT_MTD.1/CoreData Management of TSF Data

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

FMT_MTD.1/CryptoKeys Management of TSF Data

FMT_MTD.1.1/CryptoKeys The TSF shall restrict the ability to *manage the cryptographic keys to Security Administrators*.

FMT_MOF.1/Functions Management of Security Functions Behaviour

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

FMT_MOF.1/ManualUpdate Management of Security Functions Behaviour

FMT_MOF.1.1/ManualUpdate The TSF shall restrict the ability to enable the functions *to perform manual updates to Security Administrators*.

FMT_MOF.1/Services Management of Security Functions Behaviour

FMT_MOF.1.1/Services The TSF shall restrict the ability to **start and stop** ~~the functions~~ **services** to Security Administrators.

FMT_SMF.1 Specification of Management Functions

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

- *Ability to administer the TOE locally and remotely;*
- *Ability to configure the access banner;*
- *Ability to configure the session inactivity time before session termination or locking;*
- *Ability to update the TOE, and to verify the updates using [digital signature] capability prior to installing those updates;*
- *Ability to configure the authentication failure parameters for FIA_AFL.1;*
- [
    - *Ability to start and stop services;*
    - *Ability to modify the behaviour of the transmission of audit data to an external IT entity;*
    - *Ability to manage cryptographic keys;*
    - *Ability to configure the cryptographic functionality;*
    - *Ability to configure thresholds for SSH rekeying;*
    - *Ability to configure the interaction between TOE components;*
    - *Ability to configure NTP;*
    - *Ability to manage the TOEs trust store and designate X509.v3 certificates as trust anchors;*
    - *Ability to import X.509v3 certificates to the TOEs trust store;*
    - *Ability to manage the trusted public keys database*[8]].

FMT_SMF.1/FFW Specification of Management Functions

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

- *Ability to configure firewall rules;*

FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

[8] As per TD0631

- *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.

5.3.8 Class FPT: Protection of the TSF

FPT_APW_EXT.1 Protection of Administrator Passwords

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

FPT_APW_EXT.1.2 The TSF shall prevent the reading of plaintext administrative passwords.

FPT_ITT.1 Basic internal TSF data transfer protection (Refinement)

FPT_ITT.1.1 The TSF shall protect TSF data from disclosure and **detect its** modification when it is transmitted between separate parts of the TOE **through the use of [*SSH*]**.

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 [*synchronize time with an NTP Server*].

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.

FPT_TST_EXT.1 TSF Testing (Extended)

FPT_TST_EXT.1.1 The TSF shall run a suite of the following self-tests [*during initial start-up (on power on), at the conditions [key generation, random number generation]*] to demonstrate the correct operation of the TSF: [

1. *At start-up:*
    a. *Software integrity and authenticity tests,*
    b. *Known Answer Tests for symmetric cryptographic algorithms,*
    c. *Known Answer Tests for Hash functions and HMAC functions,*
    d. *Known Answer tests for RSA signature computation and verification;*
    e. *Known Answer Tests for DRBGs; and*
    f. *Pair-wise consistency tests for DSA, ECDSA and RSA.*
2. *At the key generation:*
    a. *Pair-wise consistency tests for DSA, RSA and ECDSA;*
3. *At the random number generation:*
    a. *Continuous RNG tests on all DRBGs.*

]

FPT_TUD_EXT.1 Trusted Update

FPT_TUD_EXT.1.1 The TSF shall provide *Security Administrators* the ability to query the currently executing version of the TOE firmware/software and [*no other TOE firmware/software version*].

FPT_TUD_EXT.1.2 The TSF shall provide *Security Administrators* the ability to manually initiate updates to TOE firmware/software and [*no other update mechanism*].

FPT_TUD_EXT.1.3 The TSF shall provide means to authenticate firmware/software updates to the TOE using a [*digital signature*] prior to installing those updates.

5.3.9 Class FTA: TOE Access

FTA_SSL.3 TSF-initiated Termination (Refinement)

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

FTA_SSL.4 User-Initiated Termination (Refinement)

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

FTA_SSL_EXT.1 TSF-initiated Session Locking (Extended - FTA_SSL_EXT)

FTA_SSL_EXT.1.1 The TSF shall, for local interactive sessions, [

- *terminate the session*]

after a Security Administrator-specified time period of inactivity.

FTA_TAB.1 Default TOE Access Banners (Refinement)

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

5.3.10 Class FTP: Trusted Path/Channels

FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1 The TSF shall **be capable of using [*SSH, TLS*] to** provide a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server, [*authentication server*]** 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 [*Communication with an external Syslog Server, Communication with another instance of the TOE acting as an audit server, Communicating with an external RADIUS Server*].

FTP_TRP.1/Admin Trusted Path

FTP_TRP.1.1/Admin The TSF shall be **capable of using [*SSH, HTTPS*] to** provide a communication path between itself and **authorized** remote **Administrators** 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*.

5.4 Security Assurance Requirements

This section states the Security Assurance Requirements. The applicable Security Assurance Requirements are stated in Table 12.

Table 12 Security Assurance Requirements

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

The refinement in ASE_TSS.1 as defined in the Base-PP is as follows:

ASE_TSS.1.1C Refinement: The TOE summary specification shall describe how the TOE meets each SFR. **In the case of entropy analysis, the TSS is used in conjunction with required supplementary information on Entropy.**

5.5 Security Requirements Rationale

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

The Security Assurance Requirements are drawn from the Base-PP only. That is consistent with the PP-Configuration. No security assurance components are added or removed. Therefore, the Security Assurance Requirements Rationale of the Base-PP is directly applicable to the ST and is not repeated.

6 TOE Summary Specification

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

6.1 Fulfillment of the Security Functional Requirements

The fulfillment of the mandatory security functional requirements is given in Table 13.

Table 13 Fulfillment of the Mandatory Security Functional Requirements

Security Functional Component	Fulfillment
FAU_GEN.1 **FAU_GEN_EXT.1** **FAU_GEN.2**	The TOE implements an audit function using syslog. Audit records are generated and stored for the following events and for each event related to specific SFRs as enumerated in Table 11: – Start-up and shut-down of the audit functions, – 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 event itself and a unique key name or key reference of the affected keys, and – Resetting passwords, including the identification of the user account. 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, identity of the subject if applicable, the outcome of the event if applicable, and all the additional SFR-specific data enumerated in Table 6. All cryptographic keys are obscured when stored in audit logs to ensure that they shall not be disclosed. For the ephemeral SSH session keys the PID is used as the key reference to relate the audit events on key generation and key destruction. Key destruction is recorded as a session termination event. The TOE implements a clock which is used for a time source for time stamps. The clock is synchronized using NTP. Each audit record includes a time stamp which states the exact time on which the auditable event occurred. Each distributed component of the TOE (Conductor and Router) implements an identical audit function. Therefore, each instance of the TOE generates identical sets of audit records.
FAU_STG_EXT.1 **FAU_STG_EXT.4** **FAU_STG_EXT.5**	Each instance of a TOE is a standalone appliance which generates and stores the audit log entries and stores them locally. Both the instances of the TOE configured as Conductor and the instances of the TOE configured as Router behave identically. The log files are not protected by cryptographic means but there are no user interface functions or other means for modifying them.

<table>
<tr><td></td><td>Audit records on the TOE are stored locally as syslog entries. The TOE may also accept a TLS connection from an external audit server and forward the syslog entries to the audit server protected by TLS.

If the TOE is configured to forward the log entries to an external syslog server, the entries are not stored locally. They are buffered for transport to the syslog server and the entries in the buffer are forwarded to the syslog server in real time.

The TOE stores the syslog entries locally in a disc partition of a fixed size. The size is defined at the provisioning of the TOE and may not be modified by the Administrator. If that partition becomes full, the TOE shall stop the auditing service. The stopping of the service shall generate an audit record which shall be stored in the log file. Each subsequent new audit record shall be dropped.</td></tr>
<tr><td>FCS_CKM.1

FCS_CKM.2</td><td>The TOE uses RSA as an asymmetric algorithm for SSH and TLS authentication keys and key establishment keys.

The RSA key size used with TLS and SSH are 2048, 3072, and 4096 bits. ECC with NIST curves P-256, P-384 and P-521 is also used with TLS. ECC keys for TLS are exchanged in accordance with NIST SP 800-56A Revision 2.

SSH host keys used for host authentication are generated at the first start-up of the TOE (if not already configured) using the RSA scheme as defined in FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3.

SSH key establishment keys are generated and exchanged in accordance with Diffie-Hellman on group 14, specifically using diffie-hellman-group14-sha1 as defined in RFC 4253 and diffie-hellman-group14-sha256 in RFC 8268. Diffie-hellman-group16-sha512 and diffie-hellman-group18-sha512 are also used as defined in RFC 8268.

The TLS keys are generated for RSA and ECC. DH Groups 14, 15 and 16 are used for RSA. DH Groups 17, 18 and 19.

The exact method key generation and exchange is defined in NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography".</td></tr>
<tr><td>FCS_CKM.4</td><td>The TOE implements functions for secure erasure of cryptographic keys and Critical Security Parameters (CSK). The keys and CSPs stored in the volatile memory are typically erased by the TOE software at the termination of a session. Keys and Critical security parameters, including long-term cryptographic keys, stored in the non-volatile memory are erased when the administrator uninstalls the TOE. The details are given in Sect. 6.3.1.</td></tr>
<tr><td>FCS_COP.1/DataEncryption

FCS_COP.1/SigGen

FCS_COP.1/Hash

FCS_COP.1/KeyedHash</td><td>The TOE implements AES, asymmetric cryptography, and cryptographic hashing, and HMAC using CAVP validated cryptographic functions. The key sizes, modes of operation and other parameters used, and the CAVP certificate references are given in Sect. 6.3.2.

The HMAC algorithms used by the TOE are detailed in the following:

<table>
<tr><td></td><td>HMAC-SHA-256</td><td>HMAC-SHA-384</td><td>HMAC-SHA-512</td></tr>
<tr><td>Key length</td><td>256 bits</td><td>512 bits</td><td>512 bits</td></tr>
<tr><td>Hash function</td><td>SHA-256</td><td>SHA-384</td><td>SHA-512</td></tr>
</table>
</td></tr>
</table>

	<table><tr><td>Block size</td><td>512 bits</td><td>512 bits</td><td>1024 bits</td></tr><tr><td>Output size</td><td>256 bits</td><td>384 bits</td><td>512b its</td></tr></table>
FCS_NTP_EXT.1	The TOE implements NTPv4 as defined in RFC5905 for synchronizing the clock of the TOE with a NTP server. Time synchronization messages are protected with SHA-1 which the TOE verifies prior to accepting the time. No timestamps are accepted from multicast or broadcast addressed. The TOE is capable of connecting to three or more NTP Servers as configured by the Administrator
FCS_RBG_EXT.1	The TOE generates random bits in accordance with NIST Special Publication 800-90 using CTR_DRBG (AES). The RBG does not require any configuration and is seeded from two platform-based entropy sources and one software-based entropy source. The entropy source provides 59.28 bits per 64-bit block of output of entropy rate as its output. The platform-based entropy sources are used by the Linux kernel to seed the internal DRBG and produce random bits which are written to the /dev/urandom. The entropy sources are the Intel CPU RDRAND instruction (on those variations of the TOE where available) and the network packet jitter are used at the boot time to seed the Linux random number generator which then produces randomness and writes it to /dev/urandom. The Linux kernel may also write randomness to other files but only that written to /dev/urandom is used by the TOE. The randomness from /dev/urandom is read and used as a seed by the OpenSSL library DRBG. 256 bits of entropy is read from /dev/urandom and used for seeding the OpenSSL DRBG. The OpenSSL DRBG is then used by the TOE for the run-time generation of random bits as required.
FCS_SSHC_EXT.1 **FCS_SSHS_EXT.1** **FCS_SSHS_EXT.1/Peer**	The TOE implements a SSH Client and a SSH Server. SSH may be used for SSH connections between the TOE and the remote management station, and to secure the transfers between the distributed components of the TOE. The requirements fulfilled by the SSH Server when used for remote management connections are expressed in FCS_SSHS_EXT.1. The requirements fulfilled by the TOE for the protection of transfers between the distributed components of the TOE are expressed in FCS_SSHS_EXT.1/Peer and FCS_SSHC_EXT.1. The TOE uses a 2048-bit RSA Host Key for SSHv2. The key is generated randomly at the initial setup of the TOE and is with an overwhelming probability unique to each host. The key cannot be managed using the CLI. The client presents the TOE with its public key which the TOE matches against its authorized_keys 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. When using SSH, different algorithms are used depending on whether SSH is used for securing a connection between a TOE and a remote management station, and when SSH is used for protecting the internal transfers between components of the TOE. The SSH Client for protecting the transfer between the distributed components of the TOE implements the following cryptographic primitives:

	– The supported SSH payload protection algorithms are aes128-ctr, aes256-ctr, aes128-gcm@openssh.com, and aes256-gcm@openssh.com. Cipher "none" is not accepted. – The authentication algorithms supported for the protection of inter-TOE peer connections are rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and ecdsa-sha2-nistp521. Authentication method "none" is not allowed. The TOE responds to it with a list of permitted authentication methods. – Integrity of the messages is protected with hmac-sha2-256, hmac-sha2-256, and implicit method. – Key establishment methods which are supported are ecdh-sha2-nistp256, diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp384, and ecdh-sha2-nistp521. The SSH Server for protecting the transfer between the distributed components of the TOE implements the following cryptographic primitives: – The supported SSH payload protection algorithms are aes128-ctr and aes256-ctr. Cipher "none" is not accepted. – The authentication algorithms supported for the protection of inter-TOE peer connections are ssh-rsa, rsa-sha2-256, rsa-sha2-512, and ecdsa-sha2-nistp384. Authentication method "none" is not allowed. The TOE responds to it with a list of permitted authentication methods. – Integrity of the messages is protected with hmac-sha2-256 and hmac-sha2-256. – Key establishment methods which are supported are diffie-hellman-group14-sha1 and diffie-hellman-group14-sha256. The SSH Server for remote management connection supports the following cryptographic primitives: – The supported SSH payload protection algorithms are aes128-ctr and aes256-ctr. Cipher "none" is not accepted. – The supported authentication algorithms are ssh-rsa, rsa-sha2-256, and rsa-sha2-512. Authentication method "none" is not allowed. The TOE responds to it with a list of permitted authentication methods. – Integrity of the messages may be protected with hmac-sha2-256 and hmac-sha2-256. – The supported key establishment methods are diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp256, ecdh-sha2-nistp384, and ecdh-sha2-nistp521. For connections between the TOE and a remote management station, SSH server listens to port 22. The inter-TOE communication ports may be configured by the administrator. The TOE rekeys every 1 gigabyte of data or after a session life-time reaches sixty minutes, whichever occurs first. 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.

	When a connection is brought down, the TOE does not attempt to re-establish it. The TOE sshd server does not support debug messages via the CLI. The TOE implements a timeout period for authentication of the SSHv2 protocol and enforces a limit of three failed authentication attempts before sending a disconnect to the client. Public key authentication method implicit is not supported. SSH Server for remote administration supports password-based authentication. SSH Server for peer to peer communication does not support password-based authentication. The SSH Client does not implement password-based authentication. Authentication of administrative sessions succeeds if a correct administrative password is presented to the TOE. The TOE does not require multiple authentications (public key and password) for users. The TOE does not support the configuration of host-based authentication methods. The TOE does not accept authentication if the requested service does not exist. Authentication requests for non-existent usernames will not succeed. The TOE returns a disconnect message as it would for failed authentications. This prevents attackers from enumerating valid usernames. The TOE reads the packet payload size in the TCP packet to determine the packet length. Packets greater than 262144 bytes are dropped and the connection is terminated. Negotiation of HMAC-SHA1 in each direction for SSH transport is not allowed. Diffie-hellman-group14-sha1 is not supported by the SSH Client. Key re-exchange is performed when SSH_MSG_KEXINIT is received. Port forwarding and sessions to clients are allowed. X11 forwarding is prohibited. The TOE does not implement the recommended modes AES192-ctr or 3des-ctr. None of the optional modes are implemented. The recommended and optional algorithms hmac-sha2-256 and hmac-sha2-512 are implemented for SSH transport.
FCS_TLSC_EXT.1 **FCS_TLSC_EXT.2** **FCS_TLSS_EXT.1**	The TOE implements a TLS Client and a TLS Server for trusted channels. The TLS Client supports mutual authentication using X.509 certificates. Only TLS v1.2 as defined in RFC 5246 is supported. Every other TLS or SSL version is rejected. Specifically, the TLS Server shall reject any client requesting a connection using SSL 2.0, SSL 3.0, TLS 1.0 or TLS 1.1. The TOE implements the following cipher suites for a TLS Server for use with the administration of the TOE over WebUI: – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256

	– TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA The TOE implements the following ciphersuites in the TLS Client for the export of log files and RADsec: – TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 – TLS_DHE_RSA_WITH_AES_256_GCM_SHA384 – TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256 – TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 – TLS_DHE_RSA_WITH_AES_128_GCM_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 – TLS_DHE_RSA_WITH_AES_256_CBC_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 – TLS_DHE_RSA_WITH_AES_128_CBC_SHA256 – TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA – TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA – TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA For those cipher suites which use RSA, the Diffle-Hellman Groups 14 (for 2048-bit RSA keys), 15 (for 3072-bit keys) and 16 (for 4096-bit RSA keys) are used in the key exchange. For those cipher suites which use ECDSA, Diffie-Hellman Groups 17 (for curve secp256r1), 18 (for curve secp384r1) and 19 (for curve secp521r1) are used. When establishing a trusted channel, the TOE verifies that the presented identifier of the peer entity matches the reference identifier per RFC 6125 section 6. Public key certificates of the peer entities are validated with X.509 certificates. Mutual authentication with X.509 certificates is supported. If the certificate is not valid, the trusted channel is not established. That is the default behavior, and the TOE does not implement any mechanisms which would allow the Administrator to select an alternative behavior. TLS server only supports ECDHE for key establishment. TLS client supports ECDHA and DHE key establishment. Elliptic curve extensions with the curves secp256r1, secp384r1 and secp521r1 are supported in the Client Hello message. The required behavior of the supported group and curve extensions is performed by default without having to be explicitly configured. Session resumption based on session IDs according to RFC 5246 and session tickets according to RFC 5077 are supported. The TOE is only used in FIPS mode which only supports TLS1.2. RFC 4346 is applicable to TLS 1.1 which is not supported by the TOE. Therefore, RFC 4346 is not supported by the TOE. Session tickets are encrypted using AES with 256bit and 128bit keys. AES 256 GCM, AES 128 GCM, AES 256 CBC and AES 128 CBC are implemented. The structure of the session tickets adheres to AES CBC mode as defined in RFC 5077.

FDP_RIP.2	If a session exists, the TOE associates each packet to the session. The forwarding decision made for the packets associated to a session is cached. The cached decision is applied to each packet associated to a session. In case of stateless protocols (i.e. UDP), a timer is set, and the forwarding decisions are cached until the timer expires. When a stateful session is terminated or a stateless session timer expires, the TOE zeroizes both the data structure holding the session data and the cached forwarding decision(s). This forces the TOE to ensure that a fresh forwarding decisions are made on each new session, and that the forwarding decision is based on the filtering rules. The previous forwarding decisions cannot influence the forwarding decisions made for packets associated to new sessions.
FFW_RUL_EXT.1	The TOE implements commands for the Administrator to configure the stateful packet filtering rules. The rules are constructed based on header fields in IPv4, IPv6, ICMPv4, ICMPv6, TCP and UDP headers, and applied in the sequence in which they are stored in the rule base to all network traffic processed by the TOE. The TOE is configured to associate network interfaces to IP subnets and source IP addresses are associated with 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 when a session inactivity timer expires, or when the session is terminated. When the TOE boots up, it executes a suite of self-tests. For the boot sequence to proceed, each self-test must pass. Network interfaces of the TOE are only activated when all functions required for processing the datagrams are verified and loaded. This ensures that the only when the TOE is fully operational, and all rules enforced before receiving any traffic through the physical interfaces. Datagram processing is controlled by a flow daemon. If the flow daemon fails, packet processing will stop and no traffic will be forwarded. This ensures that a failure in other daemons will not prevent the flow daemon from enforcing the TOE security policies. Also, stopping of the packet forwarding in case of a failure in the flow daemon ensures that the default deny policy is enforced by the TOE. The rules in the rule base are examined in sequence. If a rule is found to match a datagram, the action of that rule is performed. Ergo, only if an explicit rule exists to allow traffic, the traffic shall be forwarded. If no rule in the rule bases allows specific traffic, datagrams are dropped and not logged. Sessions are not created for denied traffic, only for traffic which is allowed. The TOE supports FTP (RFC 959) to dynamically establish sessions allowing network traffic according to Administrator rules. Session events will be logged in accordance with 'log' operations defined in the rules. Source and

	destination addresses, source and destination ports, transport layer protocol, and TOE Interface are recorded in each log record. The TOE implements an application Layer gateway which inspects FTP traffic to determine the port number used for data sessions. The gateway permits data traffic for the duration of the session, closing the port when the session ends. In the FTP context, session refers to the TCP data transfer connection, not the duration of the FTP control session. The following as packets that will be automatically dropped and are counted but not logged: a) Packets which are invalid fragments, i.e. any fragment that deviates from the followed standard, b) Fragments that cannot be completely re-assembled, c) Packets where the source address is defined as being on a broadcast or multicast network, d) Packets where the source address is defined as being a loopback address, e) Packets where the source or destination address of the network packet is defined as being unspecified (i.e. 0.0.0.0) or an address "reserved for future use" (i.e. 240.0.0.0/4) as specified in RFC 5735 for IPv4, f) 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, and g) Packets with the IP options Loose Source Routing, Strict Source Routing, or Record Route specified. Additionally, the TOE checks packets for validity. This is implemented by the following traffic being dropped and counted: a) Overlapping fragments, b) The total fragments in one packet are more than 62 pieces, c) The total length of merged fragments is larger than 64k, d) Any fragment in one packet not arriving in 2 seconds, e) The total number of the queued fragments has reached platform-specific limitations, f) The total number of concurrent fragment processing for different packet has reached platform-specific limitations, g) Packets where the source address is equal to the address of the network interface where the network packet was received, h) Packets where the source or destination address of the network packet is a link-local address, and i) Packets where the source address does not belong to the networks associated with the network interface where the network packet was received. Stateful sessions, specifically TCP, are established using the protocol-specific handshake mechanism. The TOE establishes the session in accordance with the protocol minus the options of implemented for protecting the TOE from flooding attempts. A TCP connection is allowed if the source address, destination addresses, source port and destination port are allowed by the

	firewall rules. The TOE establishes a connection and stores the connection status to a dedicated data structure. The filtering decisions for that connection are cached and any decision is made by first checking the cache. If a cached decision does not exist, the decisions is made based on the firewall rules. The session and the cached are maintained until the session is terminated as per the protocol rules. When a session is terminated, the session data and the filtering decision cached are cleared. This ensures that the stateful session is removed and any new session must be completely established, and the filtering rule decisions made for the new session. The TOE can be configured to deviate from the protocol rules to protect from the flooding attempts. The TOE may be configured to drop connection attempts after a defined number of half-open TCP connections using the screen 'tcp syn-flood'. The screen provides both source and destination thresholds on the number of uncompleted TCP connections, as well as a timeout period. The source threshold option allows administrators to specify the number of SYN segments received per second from a single source IP address—regardless of the destination IP address—before the TOE begins dropping connection requests from that source. The TOE maintains a data structure holding each half-open TCP connection and uses that data structure to limit the number of allowed half-open connections. For each incoming SYN request, the number of received requests to/from that IP address is examined and if the number has reached the defined threshold, each new SYN request is dropped. Similarly, the destination threshold option allows the Administrator to specify the number of SYN segments received per second for a single destination IP address before the TOE begins dropping connection requests to that destination. The timeout option allows administrators to set the maximum length of time before an uncompleted connection is dropped from the queue. For UDP traffic, the TOE maintains a decision threshold timer which is used for determining whether the packets belong to the same flow.
FIA_AFL.1	The Administrator always identifies and authenticates to the TOE using a username and password. The authentication may be locally from a console or remotely from a remote management station but the same method of dealing with authentication failures applies. For each username, the TOE starts a counter for the failed, consecutive authentication attempts. If the authentication attempt fails, the counter value is incremented. If the counter reaches the Administrator-configured maximum value for authentication failures, the offending account is locked for a period of time set by the Administrator. While locked, no authentication attempts are allowed on that account. When an account is locked, other Administrator accounts will remain active, and the locked account shall be unlocked once the locking period expires.
FIA_PMG_EXT.1	Authentication data for fixed password authentication is a case-sensitive, alphanumeric value. The password may be of a minimum length between 8

	and 15 characters. The minimum length value may be configured by the Administrator. Each password must contain characters from at least two different character sets (upper, lower, numeric, punctuation). Any standard ASCII, extended ASCII and Unicode characters can be selected when choosing a password. Specifically, the following special characters are allowed: [“!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(”, “)”.
FIA_UAU.7	When authenticating from a local management station (i.e. the console), the TOE does not echo the characters entered by the user. This prevents unauthorized users from learning the passwords or any password information (e.g. the number of characters) by monitoring the display of the remote management station.
FIA_UAU_EXT.2 **FIA_UIA_EXT.1**	Each user is associated with a username and password. When a user authenticates, the user identifies himself/herself with a username and enters a password for authentication. For each user, the TOE stores a digest of a reference password computed when the user selects the password. The entered password is hashed, and the two hashes are compared. If they match, the authentication is considered successful, and the user is granted access to the TOE. If the user has been assigned to a role with sufficient credentials, the user interface shall be made available to the user. Predominantly, functions of the TOE are only made available to successfully authenticated users assigned to a role with sufficient credentials to access the function. However, there are the following exceptions: 1. As part of the authentication exchange, the TOE displays an access banner to all users. This is part of the authentication window and is displayed to each user even if not yet authenticated. 2. The TOE does respond to ICMP Echo messages to allow basic diagnostics even if there is no authenticated user session active. 3. The TOE allows establishment of SSH connections between itself and a remote management station. The connection must be requested by the remote management station. 4. The TOE allows a TLS session for HTTPS used for the authentication of the user by WebUI. Users can connect to the TOE by the following means: From console, when the user authenticates to the TOE from a console directly connected to the TOE. In this case, the authentication exchange is carried out directly between the TOE and the user. 1. From a remote management workstation using the CLI. The user authenticates to the TOE from a remote (i.e. connected over a network) management station. The management station first establishes a SSH connection between itself and the TOE and upon establishment of the connection, carries out the authentication exchange with the user. 2. From a remote management workstation using the WebUI. The user authenticates to the TOE from a remote (i.e. connected over a network) management station. The management station first establishes a TLS connection between itself and the TOE and upon

	establishment of the connection, carries out the authentication exchange with the user using HTTPS. 3. Authentication of the remote IT products is through SSH and pre-distributed public keys. The TOE does not implement X.509 certificate-based authentication methods. The Administrator connects locally or remotely to a TOE configured Conductor. From the Conductor, the Administrator issues command to the TOE configured as a Conductor to which the Administrator connected to, or to the instances of a TOE connected as Routers.
FIA_X509_EXT.1/Rev **FIA_X509_EXT.2** **FIA_X509_EXT.3**	To validate certificates, the TOE extracts the subject, issuer, subjects public key, signature, basicConstraints and validity period fields. If any of those fields is not present, the validation fails. The issuer is looked up in the PKI database. If the issuer is not present, or if the issuer certificate does not have the CA:true flag in the basicConstraints section, the validation fails. The TOE verifies the validity of the signature. If the signature is not valid, the validation fails. It then confirms that the current date and time is within the valid time period specified in the certificate. The TOE also extracts the extendedKeyUsage field and verifies the value represents that for the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3). The TOE is configured to perform a revocation check using the Online Certificate Status Protocol (OCSP) as specified in RFC 6960. If the status check fails because of a connection failure, there are two possible outcomes, configurable by the Administrator: 1. If the TOE is configured with the option to skip the certificate revocation, the certificate shall be considered as having passed the validation. 2. If the TOE is configured with the option to skip certificate revocation checking on connection failure disabled, then the certificate is considered to have failed validation. The TOE validates a certificate path by building a chain of (at least 3) certificates based upon issuer and subject linkage, validating each according to the certificate validation procedure described above. If any certificate in the chain fails validation, the validation fails as a whole. A self-signed certificate is not required to be at the root of the certificate chain. The TOE determines if a certificate is a CA certificate by requiring the CA:true flag to be present in the basicConstraints section. The TOE generates Certificate Request Messages as specified in RFC 2986. Device-specific information, Common Name, Organization, Organizational Unit, Country and public key details are provided in the CSR. The SSR-OS validates the chain of certificates from the Root CA when the CA Certificate Response is received.
FMT_MOF.1/Functions **FMT_MOF.1/ManualUpdate** **FMT_MOF.1/Services**	For each management command, the Administrator connects locally or remotely to a TOE configured into a Conductor. From the Conductor, the Administrator issues command to the TOE configured as a Conductor to which the Administrator connected to, or to the instances of a TOE connected as Routers.

	The TOE allows the administrator to modify the TLS Server behavior to control how the syslog server may connect to the TOE. The syslog behaviour is configurable under "configure authority router system syslog". Any instance of the TOE may be connected to a syslog server. The TOE notifies the Administrators accessing the TOE automatically of the availability of software updates. Once an update is available, an Administrator (i.e. a user with sufficient credentials to enter the Administrator role) may use the user interface to manually update the TOE software. Users with no Administrative privileges are not granted access to the user interface and can, therefore, not update the TOE software. The updating of the TOE software is done as described under FPT_TUD_EXT.1. The software is identical at each instance of the TOE. Only the configuration of the software is different depending on whether an instance the TOE acts as Router or Conductor. The software may be upgraded on any instance of a TOE. The Administrator may turn the auditing service on and off. That is done through the user interface by setting the audit administration enabled flag to be of value true. Each turning on or off of the auditing service generates a log entry into the audit logs. The auditing service may be turned on or off on any instance of a TOE.
FMT_MTD.1/CoreData	The TOE only allows access to the management functions of the TOE to the users assigned to an Administrator role. Non-management functions may be made available to the successfully identified and authenticated users assigned to the user roles. The only TOE functions available to the users prior to a successful identification and authentication are the following: 1. Displaying the access banner. The administrators may configure an access banned which is displayed to the users when attempting to use the TOE locally or from a remote management station. The access banned is only displayed and does not allow any means of entering data or manipulating the TOE. The Administrator may only connect to the instances of TOE configured as Conductor. Therefore, the access banner is only displayed by those instances of the TOE which are configured as Conductor. 2. Responding to ICMP Echo. The Echo protocol is a simple IP layer Request-Reply protocol for other IT devices to query the status of the TOE. ICMP echo datagrams do not require session establishment and do not carry payload. They cannot be used for accessing the TSF data or TOE functions other than responding to an ICMP Echo request. Each instance of a TOE, whether configured as Conductor or Router, responds to the ICMP Echo. 3. SSH connection between the TOE and a remote management station. SSH over Port 22 will make available to the remote user an authentication window in which the remote user may authenticate as a legitimate Administrator. The CLI shall only be made available upon successful identification and authentication. SSH itself cannot be used for issuing CLI or other commands to the TOE. The Administrator may only connect to the instances of TOE configured

	as Conductor. Therefore, SSH connections are only accepted by those instances of the TOE which are configured as Conductors. 4. HTTPS over TLS Connection between the TOE and the management station for managing the TOE using the WebUI or the REST API. The WebUI and the REST API shall only be made available upon successful identification and authentication. HTTPS or TLS itself cannot be used for issuing management commands to the TOE. The Administrator may only connect to the instances of TOE configured as Conductor. Therefore, HTTPS over TLS connections are only accepted by those instances of the TOE which are configured as Conductors.
FMT_MTD.1/CryptoKeys	Most functions on the cryptographic keys are implemented as part of the cryptographic protocols of the TOE. The protocols are executed by the TOE software without Administrator intervention. Nevertheless, the Administrator may configure the cryptographic keys used by the cryptographic Protocols by two means: 1. The Administrator may modify the file authorized_keys located in the .ssh folder in the home folder of the account used. The file contains the public keys of Clients allowed to connect to the TOE over SSH. 2. The Administrator may configure the rekeying thresholds for SSH through the sshd_config knob. Both means of configuring SSH may be carried out on each component of the TOE
FMT_SMF.1	The TOE implements a user interface where a command exists for each management and configuration function of the TOE. There are no other methods of management of the TOE. The user interface commands may be issued by three different means: 1. From a local console where the Administrator is in the same physical space than the TOE and uses a management workstation connected directly to the console port of the TOE. 2. From a remote management station where the Administrator is not in the immediate proximity of the TOE and uses a management station to connect to the TOE over a TCP/IP network. The connection between the TOE and the remote management station is protected by SSH over Port 22. 3. From another instance of a TOE configured into a Conductor. The Administrator connects to the TOE locally or from a remote management workstation, and issues life-cycle and configuration management commands to a range of instances of TOE configured as Routers. The communication between the two instances of the TOE (one configured as a Router and the other configured as a Conductor) is protected by SSH. 4. From a WebUI client protected by HTTPS over TLS. The Administrator establishes a HTTPS connection to the TOE and authenticates to the TOE. When successfully authenticated, the WebUI is made available to the Administrator. 5. As RPC calls over HTTPS using the REST API. each REST API call includes a username and password in the GET request and the RPC

	call is executed of the username and the password are correct Administrator credentials. The CLI, the WebUI and the REST API are made available to the successfully authenticated administrators in entirety. There are no CLI, WebUI or REST API subsets made available to roles other than Administrator.
FMT_SMR.2	The TOE implements a role Security Administrator. Security Administrator is the only role to which the user interface is made available, i.e. which is authorized to administer the TOE. If the identification and authentication of a user is successful and the user is authorized to administer the TOE, the user is assigned a role Security Administrator. Users assigned to non-administrative roles are not granted access to the administration functions of the TOE. The role assignment remains until the session is terminated. The TOE does not implement a role hierarchy. Each user successfully authenticated and assigned to the role Security Administrator is granted access to the entire user interface. When accessing the TOE from console, human users are identified and authenticated with a username and password. The administrator may also configure the TOE to accept SSH public-key based authentication for the users accessing the TOE from the remote management station.
FPT_APW_EXT.1	The TOE protects authentication data by two means: 1. Reference passwords are not stored in plain text but as SHA-512 digests of the reference passwords. When a password entered by a user is compared to a reference password, a SHA-512 digest is computed from the entered password and the two digests are compared. 2. The TOE is only administered or otherwise accessed through the CLI or the WebUI. Neither the CLI nor the WebUI implements any functions for reading or exporting the passwords.
FPT_ITT.1	Communication between Routers and Conductors is protected with SSH. There is no separate registration channel but the administrator configures the instances of TOE with pre-shared keys. The pre-shared keys are used by the two instances of the TOE to secure the communication. Each instance of the TOE maintains a local audit database. Instances of the TOE configured as Routers may be configured to forward the log entries to a Conductor, or directly to an external Syslog server. If the syslog entries are forwarded to a Conductor, they are protected with SSH. If they are sent directly to an external Syslog server, the Router may be configured to use either SSH or TLS to protect the traffic.
FPT_SKP_EXT.1	The user interfaces implemented by the TOE do not include commands for viewing the cryptographic keys. The TOE enforces kernel-level file access rights to the key containers. The access rights granted by the TOE limit access to the contents of cryptographic key containers only to the processes with cryptographic rights and to the shell users with root permission. As security administrators do not have root permission, the measures restrict access to the contents of the key containers to authorized processes only.

FPT_STM_EXT.1	The TOE implements a real time system clock which may be used for time stamps and clock cycles when reliable time is required. There is no user interface command for setting the time, but the TOE clock is synchronized with a NTP Server. The time is used by the TOE in the following functions: 1. Time stamps for the audit records. Each entry in the audit logs is stamped with a time stamp stating the date and time of the event. 2. Inactivity timers for the active user sessions. The TOE maintains an inactivity timer for each user session and if the idle time of the session reaches the set threshold, the TOE shall terminate the session. 3. Lockdown timers. If the number of consecutive failed authentication attempts on any used account exceeds the Administrator-defined threshold, the account shall be locked for 1800 seconds. The TOE sets a timer and once the timer expires, the account is unlocked. 4. SSH host authentication request expiration timer. 5. Rekeying timer for SSH connections on Port 22. SSH shall rekey when either an administrator-defined maximum amount of data per connection is reached or an administrator-defined maximum lifetime of the connection is reached. The default values are 1Gb of data or one hour connection lifetime. 6. Life-time counters on the stateless connections by the firewall component of the TOE. For stateful connections, the packet filtering decisions are cached for the duration of the session. For stateless connections, the cached decisions are cleared on a periodical basis when a timer expires. 7. The timeout option for TCP SYN fragment allows administrators to set the maximum length of time before an uncompleted TCP connection is dropped from the session queue.
FPT_TST_EXT.1	The TOE is to be used with the FIPS mode enabled. This ensures that the suite of self-tests on the TOE is executed at the start-up and when generating cryptographic keys of random numbers. The self-tests at the start-up commence with the TOE software integrity and authenticity test. The integrity and authenticity of the software is verified against a digital signature of the software computed at the development environment with RSA and SHA2-256. 2048-bit or 4096-bit RSA key is used. If the signature verification succeeds, the TOE software is considered unaltered and authentic, and the TOE shall proceed with the boot sequence. The next step in the boot sequence are the power-up self-tests. The power-up self-tests consist of the algorithm-specific Pairwise Consistency and Known Answer tests. If any component of the power-up self-test fails, an internal global error flag is set to prevent subsequent invocation of any cryptographic function calls. Any such power-up self-test failure is a hard error that can only be recovered by reinstalling the module. The power-up self-tests may be performed at any time by reloading the module. Additional pair-wise consistency tests are executed for asymmetric ciphers when generating cryptographic keys are generated, and the continuous RNG tests when random numbers are generated. The RNG test are in accordance with the requirements of SP800-90A.

	No operator intervention is required during the running of the self-tests.
FPT_TUD_EXT.1	The currently active version of the TOE can be queried by a legitimate Administrator by the CLI command show system version or the corresponding WebUI command or REST API call. Availability of software upgrades can be queried with the show assets software command which displays all available software upgrades. The downloading of the upgrade is done manually as is the commencement of the actual upgrade. TOE software is distributed as a package-based ISO image file. The package-based ISO format allows upgrading of the packages constituting the ISO file when distributed in RPM (Red Hat Package Manager) format. Upgrades to the TOE software are distributed in RPM format. Each instance of the TOE runs identical software. The differentiation between a Conductor and Router is through the configuration of the software. The same upgrade can be applied to each component of the TOE. The RPM format allows embedding of a digital signature into the package. Into each upgrade package is embedded a GPG digital signature produced using RSA with 2048 or 4096 bit key and SHA-256. The Package Manager of the Linux operating system which is part of the TOE automatically verifies the digital signature on the downloaded package. Only if the signature verification is successful shall the package manager install the upgrade.
FTA_SSL.3 **FTA_SSL.4** **FTA_SSL_EXT.1**	The TOE implements command quit which terminates the current session. An Administrator may issue that at any time to terminate the administrative session. For each administrative session the TOE maintains an inactivity timer which tracks the time the administrator is idle, i.e. not issuing any commands. If the inactivity timer reaches a configured maximum time, the TOE shall terminate the administrative session.
FTA_TAB.1	The administrator may configure an access banner to be displayed at each authentication exchange. The banner is displayed on each authentication window, whether local from console, remotely over SSH, remotely with the WebUI over HTTPS. The banner may provide warnings against unauthorized access to the TOE and any other information that the administrator wishes to communicate. When accessing the TOE using the REST API, the banner is not displayer. REST API is an RPC interface without an interactive login window. Each REST API call includes the Administrator username and password in the procedure call.
FTP_ITC.1	The TOE implements a SSH Client which is, together with the SSH Server, used to protect the communication between the instances of the TOE configured as Routers and Conductors. Both the Router and the Conductor may initiate the SSH connection. The SSH Server listens to the SSH connections from another instance of a TOE on port 930. SSH connection for between two instances of a TOE only implements public key cryptography based authentication. When a SSH connection is established, one instance of a TOE may send audit log entries to a connected instance configured as audit server.

	The TOE implements TLS and TLS Client. TLS may be used for connecting the TOE to an external syslog and to an external RADIUS server. Either party may establish the connection.
FTP_TRP.1/Admin	The TOE implements a SSH Server to protect confidentiality and integrity of communication between itself and a remote management station. When the TOE and a remote management station establish a SSH connection, the connection is initiated by the remote management station. SSH Server listens to port 22 for the connection requests from a remote management station. The TOE may be configured to authenticate the administrator using a username and password or public key cryptography. If the authentication is successful, the TOE establishes a SSH connection with the management workstation. Multiple authentication methods are not required. Once successfully established, the administrator accesses the CLI of the TOE from the remote management workstation and each command and response thereof is protected by SSH. The SSH connection is rekeyed if the rekeying threshold is reached, and the session is terminated if the idle timer maximum value is reached. Otherwise, the SSH connection remains until the remote administrator chooses to terminate the session. The Administrator may also connect to the TOE from a web management station or use the REST API to administer the TOE. In both cases, the TLS Client of the TOE is used for a HTTPS connection requested by the remote entity.

6.2 Fulfillment of the Security Assurance Requirements

To fulfill the Security Assurance Requirements, the developer implements a set of security assurance measures. Some assurance classes are fulfilled by the evaluator of the TOE. The security assurance measures implemented by the developer and evaluator of the TOE are described in Table 14.

Table 14 Fulfillment of the Security Assurance Requirements

Security Assurance Requirement	Fulfilment
Security Target	The developer authors a Common Criteria Security target for the Target of Evaluation. The Security Target implements all assurance components required by the Base-PP. The Security Target includes: – A ST Introduction which provides a ST Reference, a TOE Reference, a TOE Overview, and a TOE Description. – Conformance Claims stating exactly the conformance to the Common Criteria and the Protection Profiles, Protection Profile Modules and Protection Profile Configurations the Security Target and the Target of Evaluation claim conformance to. – A Security Problem Definition which is a statement of Threats, Assumptions and Organizational Security Policies applicable to the TOE.

	– A statement of the security objectives for the TOE. The Base-PP only defines security requirements for the operational environment of the TOE, but the Security Target also states the security requirements for the TOE drawn from the PP-Module. – Extended Components Definition and the statement of the security requirements state exactly the Security Functional Requirements and the Security Assurance Requirements the TOE fulfills. – TOE Summary Specification which describes for each Security Functional Requirement how the TOE fulfills that Security Functional Requirement. Additionally, the refinement in ASE_TSS.1.1C is fulfilled by the developer providing a separate, proprietary entropy assessment report.
Functional Specification	Included in the TOE Summary Specification, the developer provides all information required for a basic functional specification of the TOE.
Security Guidance	Attached to the TOE and included in the physical scope of the TOE is a Common Criteria Guidance Supplement for the TOE. The Guidance Supplement gives guidance to the user of the TOE in the secure installation and preparation of the TOE so that the TOE is in an initial secure state. The Guidance Supplement also provides guidance to the user of the TOE so that the TOE always remains in a secure state when the guidance is followed.
Life Cycle Support	The developer labels the TOE with the unique identifier. The label may be examined by the user of the TOE to ensure that the correct version of the TOE is used. When the TOE software is updated, the label of the TOE is updated accordingly. The TOE label is included in the configuration list of the TOE to ensure that the evaluator can be assured of evaluating the intended version of the TOE.
Independent Testing	The evaluator carries out a set of independent tests on the TOE. The independent tests complement the functional testing carried out by the developer and ensure that the TOE passes each applicable test required for conformance with the Base-PP and the PP-Module. The evaluator documents the testing in accordance with the requirements stated in the Base-PP, the PP-Module, and the Common Criteria evaluation and certification scheme followed.
Vulnerability Assessment	The evaluator carries out a vulnerability survey to determine that there are no obvious vulnerabilities in the TOE which could be practically exploited by the threat agents. The evaluator documents the vulnerability survey in accordance with the requirements stated in the Base-PP, the PP-Module, and the Common Criteria evaluation and certification scheme followed.

6.3 Cryptographic Details and CAVP References

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

6.3.1 Zeroization of Cryptographic Keys and Critical Security Parameters

The timing and method of the zeroization of the cryptographic keys and critical security parameters (CSP) used by the TOE is given in Table 15.

Table 15 Timing and Method of the Zeroization of Cryptographic Keys and Critical Security Parameters

Key/CSP	Storage Format	Storage Location	Zeroization Method
SSH Private Host Key	Plaintext	Non-volatile memory	When the TOE is uninstalled, the config files (including SSH host keys stored in the non-volatile memory) are erased alongside with the TOE software and configuration.
	Plaintext	Volatile memory	`free()` performed by the TOE software at session termination.
SSH Session Key	Plaintext	Volatile memory	`free()` performed by the TOE software at session termination.
User Password	Plaintext when entered	Volatile memory	`free()` performed by the TOE software at the completion of the user authentication.
	Hashed when stored	Non-volatile memory	When the TOE is uninstalled, the config files (including user passwords stored in the non-volatile memory) are erased alongside with the TOE software and configuration.
RNG State	Plaintext	Volatile memory	Overwritten by the kernel of the TOE with zeros at reboot.
System Master Password	Plaintext	Non-volatile memory	When the TOE is uninstalled, the config files (including the system master password stored in the non-volatile memory) are erased alongside with the TOE software and configuration.
ecdh private keys	Plaintext	Memory	Memory free() operation is performed by SSR-OS upon session termination.

6.3.2 CAVP Certificate References

The TOE implements a rich set of cryptographic functions used to protect communications and the integrity of the security functions. Each cryptographic function of the TOE is CAVP validated. The CAVP certificate references, organized by the applicable Security Functional Component, are given in Table 16.

–

Each CAVP Certificate is applicable to each variant of the TOE.

Table 16 CAVP Certificate References

FCS_CKM.1			
Applicable SFR(s)	**Claimed Algorithms and Parameters**	**Module/Library**	**CAVP Reference**
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	RSA Probable Random Prime (2048, 3072, 4096)	OpenSSL v3.2	A7733

FCS_TLSC_EXT.1, FCS_TLSS_EXT.1	(KeyGen)		
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1 FCS_TLSC_EXT.1 FCS_TLSS_EXT.1	ECDSA (P-256, P-384, P-521) (KeyGen, KeyVer)	OpenSSL v3.2	A7733
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	FFC Safe-Prime (DH Groups 14, 16, 18)	OpenSSL v3.2	Tested by FCS_CKM.2
FCS_CKM.2			
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	DH (Groups 14, 16, 18)	OpenSSL v3.2	Tested against known-good implementation
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	KAS-ECC-SSC (ECDH) (P-256, P-384, P-521)	OpenSSL v3.2	A7733
FCS_COP.1/DataEncryption			
FCS_SSHS_EXT.1	AES-CTR (128, 256) (Encryption, Decryption)	OpenSSL v3.2	A7733
FCS_SSHC_EXT.1	AES-CTR (128, 256) AES-GCM (128, 256) (Encryption, Decryption)	OpenSSL v3.2	A7733
FCS_TLSC_EXT.1 FCS_TLSS_EXT.1	AES-GCM (128, 256) (Encryption, Decryption)	OpenSSL v3.2	A7733
FCS_COP.1/SigGen			
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1 FCS_TLSC_EXT.1 FCS_TLSS_EXT.1 FPT_TUD_EXT.1	RSA PKCS1v1_5 (n=2048 w/ SHA-256, SHA-384, and SHA-512; n=3072 w/ SHA-256, SHA-384, and SHA-512; n=4096 w/ SHA-256, SHA-384, and SHA-512) (SigGen, SigVer)	OpenSSL v3.2	A7733
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1 FCS_TLSC_EXT.1 FCS_TLSS_EXT.1	ECDSA (P-256 w/ SHA-256, SHA-284, SHA-512; P-384 w/ SHA-256, SHA-384, SHA-512; P-521 w/ SHA-256, SHA-384, SHA-512) (SigGen, SigVer)	OpenSSL v3.2	A7733
FCS_COP.1/Hash			
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	SHA-1, SHA-256, SHA-384, SHA-512	OpenSSL v3.2	A7733

FPT_TUD_EXT.1			
FCS_NTP_EXT.1	SHA-1	GnuTLS v3.8.3	A7734
FCS_COP.1/KeyedHash			
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1 FCS_TLSC_EXT.1 FCS_TLSS_EXT.1	HMAC-SHA1, HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	OpenSSL v3.2	A7733
FCS_RBG_EXT.1			
FCS_SSHC_EXT.1 FCS_SSHS_EXT.1	CTR_DRBG (AES) (SHA-256)	OpenSSL v3.2	A7733

7 Acronyms

AES	Advanced Encryption Standard
ANSI	American National Standards Institute
API	Application Programming Interface
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CFP	C Form-factor Pluggable
cPP	collaborative Protection profile
CSP	Critical Security Parameter
DH	Diffie-Hellman
EAL	Evaluation Assurance Level
ECC	Elliptic Curve Cryptosystem
ECDSA	Elliptic Curve Digital Signature
EP	Extended Package
ESP	Encapsulating Security Payload
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standard
HPE	Hewlett Packard Enterprise
HMAC	Keyed-Hash Message Authentication Code
IA	Identification and Authentication
ICMP	Internet Control Message Protocol
ID	Identity
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
ISO	International Organization for Standardization

IT	Information Technology
MAC	Message Authentication Code or Media Access Code
MIC	Modular Interface Card
MPC	Modular Port Concentrator
MS-MPC	MultiServices Modular Port Concentrator
NAT	Network Address Translation
NTP	Network Time Protocol
OSI	Open Systems Interconnect
PAM	Pluggable Authentication module
PFE	Packet Forwarding Engine
PIC/PIM	Physical Interface Card / Physical Interface Module
PKI	Public Key Infrastructure
PoE	Power over Ethernet
PRNG	Pseudo Random Number Generator
RADIUS	Remote Authentication Dial-In User Service
RADSEC	RADIUS over TLS
REST	Representational State Transfer
RE	Routing Engine
RFC	Request For Comments
RNG	Random Number Generator
RPM	Red Had Package Manager
RSA	Rivest-Shamir-Adleman
SA	Security Association
SFP	Small Form-factor Pluggable
SHA	Secure Hash Algorithm
SNMP	Simple Network Management Protocol

SSH	Secure Shell
SSL	Secure Socket Layer
SSR	Session Smart Routing
TCP	Transport Control Protocol
TLS	Transport Layer Security
UDP	User Datagram Protocol
WAN	Wide Area Network