Security Target

EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5 with Endace OSm v7.2

ST Version: 1.0

Date: June 26, 2025

Prepared for:

endace

https://www.endace.com/

Prepared by:

TERON LABS

www.teronlabs.com

Contents

List of Tables

1 Security Target Introduction

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

The target audience is the users and the potential users of the TOE wishing to gain a precise understanding of the TOE and the security features provided. The readers are assumed to possess a good understanding of the computer networking terms and practices. The readers are also expected to have a good understanding of network and computer security. Finally, the readers are assumed to be proficient in Common Criteria and the terminology thereof. Some familiarity with the networking products of Endace 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, Date: 23-March-2020 (CPP_ND_V2.2E). The ST does not claim conformance to any PP-Modules or PP-Configurations.

1.1 Security Target Reference

Security Target Title	Security Target EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5 with Endace OSm v7.2
Security Target Version	1.0
Security Target Date	June 26, 2025

1.2 TOE Reference

TOE Identification	EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5 with Endace OSm v7.2
TOE Developer	Endace
Evaluation Sponsor	Endace

1.3 TOE Overview

1.3.1 Intended Method of Use

The TOE is a suite of network appliances designed to capture an entire history of network traffic for offline analysis and investigation. It is a family of non-virtual and non-distributed network devices. Each device is a complete network appliance which includes all software, hardware, and security guidance constituting the TOE.

Distributed applications, web applications, mobile applications, cloud services and ubiquitous Internet access have all delivered unparalleled flexibility and power. They have also resulted in complex network and application architectures. Complex network and application architectures, in turn, have made it difficult for the risk owners to ensure the security, reliability and performance of networks and network applications. The TOE allows recording and storage of the entire access history of a network for detailed monitoring and forensic analysis.

As illustrated in Figure 1, the TOE is connected to a network of monitoring points from which the packets are recorded. The TOE is also connected, via a separate network interface, to a management network which provides access to a set of services used in the administration of the TOE. The recorded packets are stored locally in the drives of the TOE. The storage and recording capacities depend on the model variant of the TOE.

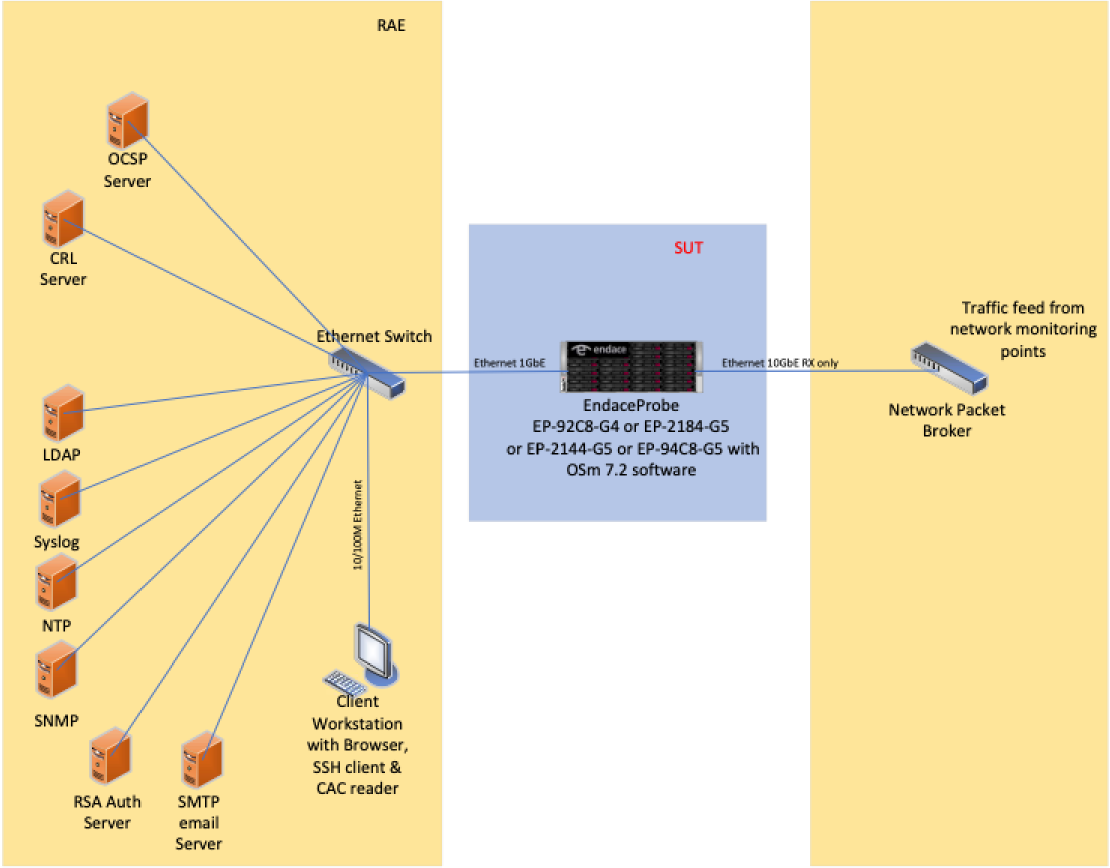
RAE
OCSP Server
CRL Server
Ethernet Switch
LDAP
Syslog
NTP
SNMP
RSA Auth Server
SMTP email Server
10/100M Ethernet
Client Workstation with Browser, SSH client & CAC reader
SUT
Ethernet 1GbE
EndaceProbe EP-92C8-G4 or EP-2184-G5 or EP-2144-G5 or EP-94C8-G5 with OSm 7.2 software
Ethernet 10GbE RX only
Traffic feed from network monitoring points
Network Packet Broker

Figure 1 Connectivity of the TOE

When a security breach occurs, it may be difficult or impossible to quickly determine the exact sequence of events, how the breach was carried out, and what data or services were compromised. Organizations also often struggle to investigate the root causes of performance problems and network outages. That, in turn, reduces productivity and negatively impacts the reputation and customer experience.

The TOE is a tool to record and store the entire network history. Endace technology allows the risk owner to record copies of every packet that traverses a network. When a network problem or a security breach occurs, exact copies of the original packets may be examined to determine the sequence and cause of the events.

EndaceProbe EP Series appliances are tools which assist the users in ensuring the security and performance of their networks as well as the integrity of their confidential data by enabling them to record an accurate history of exactly what has happened on their networks. Using this network history, Security Operations (SecOps), Network Operations (NetOps) and IT teams can quickly and accurately reconstruct the events and respond appropriately.

A typical deployment of the TOE is illustrated in Figure 2. The TOE is associated to sources of network packets which give it access to the packets that the TOE captures and stores. The captured packets can be analyzed by various third-party analytics tools hosted on the TOE, accessed from external analytics tools that have been integrated with the TOE via API, or exported for use with offline analytics tools the risk owner has deployed.

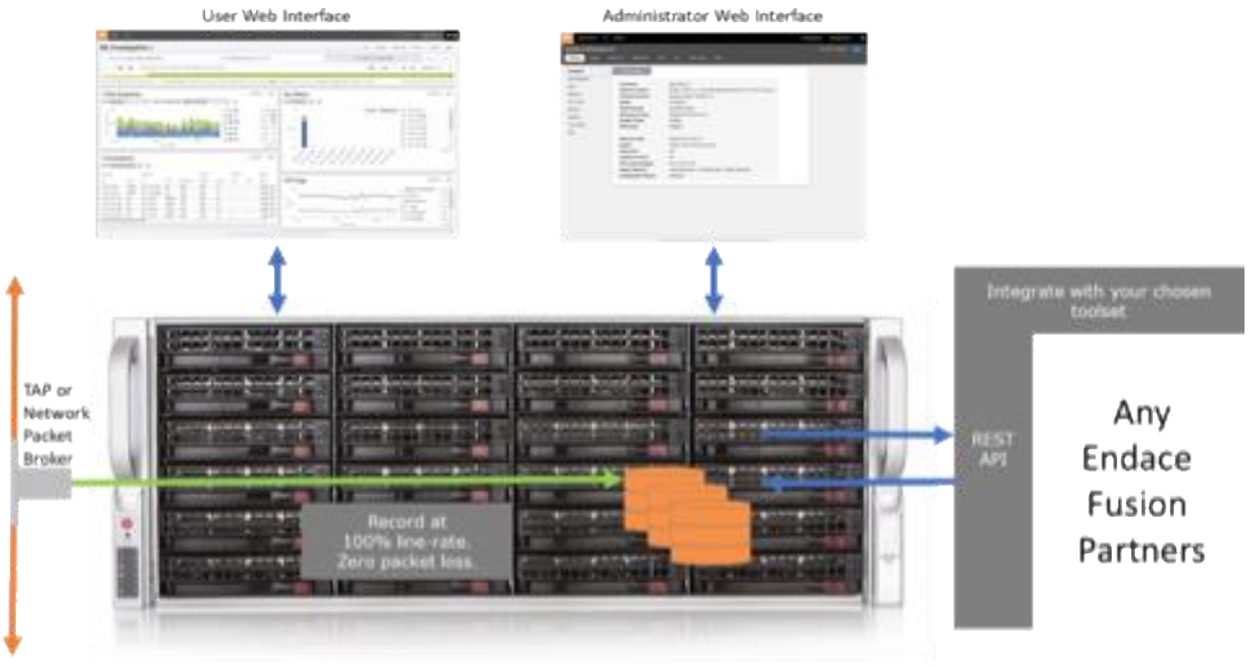
User Web Interface
Administrator Web Interface
TAP or Network Packet Broker
Record at 100% line-rate. Zero packet loss.
Integrate with your chosen toolset
REST API
Any Endace Fusion Partners

Figure 2 Representative Deployment of the TOE

The value proposition of the TOE functions is summarized as follows:

100% Accurate Recording. Endace technology provides 100% lossless packet recording with nanosecond accurate timestamping of each packet. Using a common time signal such as GPS, timestamps attached to the captured packets can be synchronized across networks. Precision and completeness are essential to accurate event reconstruction after the fact.

Analytics Workflow Integration. The TOE includes a powerful API for integrating recorded packet data with the risk owner's existing analytics tools to streamline investigation workflows and enable rapid response. Endace's Pivot-to-Vision and Pivot-to-Packets API integration lets analysts use their analytics tools of choice directly on the recorded packet data. They can analyze recorded traffic using EndaceVision™, analyze decoded packets directly using a hosted instance of Wireshark™ or download pcap files for analysis and archiving. The API also enables automation with solutions such as Security Orchestration, Automation, and Response (SOAR) tools for automated packet search and retrieval. Files and logs can be reconstructed from recorded packets using built-in file extraction and log generation.

Network History Playback. Playback lets SecOps and NetOps analysts replay recorded packet data to their analytics applications to analyze past events. This allows detailed back-in-time investigations and automated analysis that is simply not possible using conventional investigative techniques.

Provenance Enriched History. For recorded packet data to be useful, analysts need to know where it came from, how it was recorded, and what the state of the environment was at the time of recording. Provenance™ enhances recorded Network History with rich contextual data, embedding it into the packet history every second. Provenance data lives with the packets, so at any time it can be examined with decode tools like Wireshark alongside the packet data itself. With detailed information about how and where the packets were recorded there's never any doubt about the veracity of the recorded evidence.

Application Dock. Each variant of the TOE also includes Application Dock. Application Dock is EndaceProbe's built in virtual hosting environment. It builds on the concepts of Software Defined Networking (SDN) and Network Function Virtualization (NFV) and enables the virtualization of network security and performance monitoring analytics. Hosted applications can access a stream of live traffic for real time analysis. Using Playback™, the analytics tools can be fed a stream of historical traffic to investigate past events using a complete and accurate recording of the exact network events.

Once provisioned and installed, the TOE operates independently. However, the TOE does implement an administrative interface and may be managed by an authorized administrator. The administrator may manage the TOE locally from a console through a Command Line Interface (CLI) or from a remote management station over SSH using the same CLI.

If a suitable Web client is deployed in the operational environment of the TOE, the TOE also allows a non-administrative operator to connect to the TOE over a web interface. The web interface implements a Graphical User Interface (GUI) of the TOE. The connection is protected with HTTPS over TLS. No administrative functions are available over the Web interface or the GUI.

Management stations are not part of the TOE, but the TOE implements the CLI. Secure protocols are used to connect the management stations to the TOE and for the TOE to connect to other remote services. The TOE implements SSH Client, SSH Server, TLS Client, TLS Server and HTTPS for secure communication. Network peers for TLS are authenticated using X.509 certificates.

Access to the TOE is only granted to authorized administrators upon successful identification and authentication. The TOE may be configured to connect to a remote authentication server. Users can terminate their sessions, and the TOE will terminate inactive sessions after a specified period of inactivity. Audit records are generated from every auditable event and stored in log files. The log files may be exported to a remote syslog server over a secure network connection.

The platforms of the TOE are Endace branded platforms EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5. Different variants of the TOE have different capabilities as summarized in Table 1. TOE software is Endace OSm v7.2. The TOE software is deployed in an ISO package file and includes CentOS v7.9 Linux operating system with kernel version 5.14.0.

Included in the TOE distribution are also the following third-party software components:

- Apache Server 2.4.34,
- wolfSSL version 5.6.4 with WolfCrypt 5.2.1, and
- OpenSSH 7.4 libraries.

Each cryptographic algorithm on the WolfSSL v5.6.4 with WolfCrypt 5.2.1 is CAVP validated. The WolfSSL and WolfCrypt are included in the Endace Crypto Firmware v2.1. The CAVP Certificate references are given in Sect. 6.3.2.

The TOE implements TLS listeners on port 443 and on port 5558. The TLS listener on Port 443 is the TLS listener for HTTPS. The TLS listener on Port 5558 is for the Endace proprietary "eventd" used to connect to EndaceCMS for status reporting. Each TLS listener is implemented using the CAVP validated Endace Crypto Firmware v2.1.

Table 1 Performance Characteristics of TOE Variants

	EP-2144-G5	EP-2184-G5	EP-92C8-G4	EP-92C8-G5
Write to disk	10 Gbps sustained 10 Million pps >2s microburst @ 40 Gb	40 Gbps sustained 20 Million pps Unlimited microburst @ 40 Gbps	40Gbps sustained >40Gbps compressed 40 million pps >2s microburst @80Gbps	60 Gps sustained > 60Gbps compressed 40 million pps >2s microburst @ 100 Gbps
Max flow creation rate	200k flows/sec	400k flows/sec	400k flows/sec	400k flows/sec
Max concurrent flows	2 Million	6 Million	6 Million	6 Million
No. of application dock instances	Up to 4	Up to 8	Up to 12	Up to 12
Storage depth	7.6 Terabytes SSD Packets up to 20 Terabytes	46 Terabytes SSD Packets up to 120 Terabytes	Native 432 Terabytes Packets up to 1 Petabyte	Native 1320 Terabytes Packets up to 3.1 Petabytes
Physical size	1U Compact Rack Mounted	1U Compact Rack Mounted	4U Rack Mounted	4U Rack Mounted
Monitoring ports	Up to 4x10 MbE/100 MbE/1 GbE/10 GbE	Up to 4x10 MbE /100 MbE/1 GbE/10 GbE or 1x 40 GbE	Up to 8x1GbE/10GbE or up to 2 x 40GbE	Up to 8x 1GbE/10GbE/25GbE or up to 2 x 40GbE/100G
Management interfaces	4 x 100 MbE/1 GbE, 4 x 10 GbE 1x IPMI	4 x 100 MbE/1 GbE, 4 x 10 GbE 1x IPMI	2x 1GbE/10GbE, 1x IPMI	2x 1GbE, 2x 10GbE 1x IPMI

The software of the TOE only allows limited access. Administration of the TOE can only be carried out via the Endace CLI and is only granted to authenticated and authorized users. The TOE implements a number of security mechanisms to protect itself, and the critical data, and ensure that attempts to tamper with the TOE or the data contained on it 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 CPP_ND_V2.2E. 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 and the essential network security features.

3. The TOE implements trusted paths and trusted channels to allow remote IT systems - specifically, an audit server and the remote management station - to connect to the TOE in a secure manner. The trusted paths and trusted channels are implemented with SSH and TLS.
4. SSH Client and Server. The TOE implements a SSH server to allow secure connection with a syslog server and with a remote management station. SSH server and client are used for protecting the Netconf traffic used configuring the TOE.
5. HTTPS and TLS. The TOE uses HTTPS over TLS for connecting to a CRL Server, to a OCSP Server, for allowing the Web client to connect to the TOE in a secure manner for non-administrative functions, and for the Client Certificate authentication. The TOE additionally uses TLS for a secure connection to a remote syslog server, to a LDAP server, and to an RSA SecurID Authentication Manager.
6. 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.
7. Security Management. The TOE implements a Command Line Interface made available to the administrators. The CLI may be accessed locally from console or remotely over a SSH.
8. 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 network appliance implementing the security features required for exact conformance with CPP_ND_V2.2E. The TOE is neither a distributed nor a virtual network device.

1.3.4 Non-TOE Hardware, Software and Firmware

The TOE is the entire network appliance. Yet, it does require external IT devices to be properly operated. Specifically, the TOE requires the following items in the network environment:

- Syslog server including a SSHv2 client for connecting to the TOE for the TOE to send audit logs.
- A management station with a SSHv2 client for remote administration of the TOE.
- A management station with a serial connection client for local administration of the TOE.
- The administrator may configure the TOE to acquire time from an NTP Server, but the NTP Server is not part of the TOE.
- The TOE may connect to an RSA SecurID, but the SecurID is not part of the TOE.
- The user of the TOE may deploy a set of analytics tools and there are no restrictions of which tools may be used. None of those tools are part of the TOE.
- Application Dock, InvestigationManagerTM, EndaceCMSTM, and an external installation of WiresharkTM may be utilized at the user's discretion but are not part of the TOE.
- The TOE implements X.509 public key certificate validation and verification and may use Certificate Revocation Lists (CRL) or Online Certificate Status Protocol (OCSP) to verify the revocation status of the certificates. The TOE implements secure connectivity to a CRL Server or OCSP Responder, but neither is part of the TOE.

The user of the TOE may employ a suite of recording and analytics tools, with no limitations imposed on the selection of such tools. These tools are external to and not considered components of the TOE.

1.3.5 Disallowed Protocols and Services

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

- Telnet must not be used. It is not considered secure and violates the trusted path and trusted channel requirements.
- HTTP must not be used. It is not considered secure and violates the trusted path and trusted channel requirements.
- FTP and TFTP must not be used. They are not considered secure and violate the trusted path and trusted channel requirements.
- SNMP must not be used. It is not considered secure and violates the trusted path and trusted channel requirements.
- RADIUS, TACACS and TACACS+ are not considered secure and must not be used.
- IPMI/iDRAC Interface is not considered secure and must not be used.
- Web proxy function is not considered secure and must be disabled.
- The web interface, the web client, and the GUI must not be used for administrative functions on the TOE.

Address Resolution Protocol (ARP) and Internet Control Message Protocol (ICMP) may be used, but they are not covered by the evaluation of the TOE and are used without security assurance provided.

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 2.

Table 2 Parts Included in the Physical Scope of the TOE

Part of the TOE	Identification	Description
TOE Hardware	EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5	The hardware platform, the interfaces, and the casing of the TOE. Includes the processor, the memories, and the persistent storage.
TOE Software	Endace OSm v7.2	The software included in the TOE. The TOE software is distributed factory installed.
Security Guidance	Common Criteria Guidance Supplement EndaceProbe EP-92C8-G4, EP-94C8-G5, EP-2184-G5 and EP-2144-G5 with Endace OSm v7.2	The Common Criteria Guidance supplement for the TOE. The security guidance is distributed as a document in PDF format.

1.4.2 Logical Scope of the TOE

The TOE implements the security functionality required by CPP_ND_V2.2E. The major security features of the TOE are summarized in Table 3.

Table 3 Major Security Features of the TOE

Security Feature	Description
Security Audit	The TOE implements an audit function to gather a rich set of detailed audit records of all critical security operations to allow the administrators to analyze the

	state of the TOE and investigate potential security breaches. Each audit record constitutes a log entry which includes all the necessary data about the event, the outcome of it, etc. to allow detailed analysis of the audit records. Audit records are protected against unauthorized modification and may be transferred to an audit server for storage and further analysis. The transfer of the audit records to the audit server is protected by TLS.
Cryptography	The TOE implements cryptographic functions for allowing the TOE to communicate with external devices and other instances of the TOE in a secure manner. The TOE implements a random bit generator used for generating cryptographic keys. The TOE also implements key agreement mechanisms and all public key cryptographic functions, symmetric cryptographic functions, secure hash functions and keyed hash-based MAC functions for the protection of data and communication. Cryptographic keys and Critical Security Parameters (CSP) are destroyed by the TOE when no longer required. All cryptographic algorithms are implemented in a cryptographic module validated through the Cryptographic Module Validation Program (CMVP) to ensure appropriate engineering.
SSH	The TOE implements an SSH Client and an SSH Server for secure communication between the TOE and external IT devices and between the TOE and another instance of a TOE. The TOE implements public-key based and password-based authentication between itself and other IT devices, including other instances of the TOE. The public keys are stored in key containers.
HTTPS and TLS	The TOE implements a TLS Client and a TLS Server. The TOE also implements HTTPS over TLS. To support public key certificates for TLS, the TOE implements X.509 certificate validation and authentication. The TOE uses HTTPS for a secure connection to a CRL Server, to a OCSP Server, for allowing the Web client to connect to the TOE in a secure manner for non-administrative functions, and for the Client Certificate authentication. The TOE uses TLS for a secure connection to a remote syslog server, to a LDAP server, and to an RSA SecurID Authentication Manager.
Security Management	The TOE implements a CLI for the management functions. There are no alternative methods for managing the TOE. Administrators may access the CLI and perform all management tasks of the TOE. The CLI may be accessed locally from console or remotely over an SSH connection.
Identification, Authentication, Authorization and Access	The TOE ensures that only legitimate accesses to the management functions are allowed. Each user is identified with a username and password, and upon successful verification of the password the user is assigned to a role defined by the administrators of the TOE. The users are allowed to change their passwords and the TOE enforces that only good quality passwords are accepted. The password hashes and salts are stored in a secure file so that they cannot be read by any user. When a password is entered by a user of a remote management station, the characters are not echoed. This ensures that unauthorized parties may not learn passwords of legitimate Administrators. Each user may terminate their own session.

	The TOE maintains an inactivity timer for each user and if the administrator defined limit is reached, the TOE terminates that session. The TOE also maintains a counter for unsuccessful consecutive authentication attempts for remote users (i.e., Administrators accessing the TOE from a remote management station) and takes protective action if the administrative defined maximum value is exceeded. Each authentication window displays an administrator configurable access banner informing the users of the sensitive nature of the TOE and of the sanctions resulting from misuse or abuse of the TOE.
Protection	The TOE protects itself from tampering and unauthorized access by active and passive means. The active means are the active measures the TOE takes to ensure that TOE data and functions are not accessible to unauthorized users, and the passive means are the design characteristics of the TOE which minimize the attack surface accessible to threat agents. Minimization of the attack surface is achieved by the TOE software running on a dedicated hardware platform with a minimum set of physical ports and connections, and only implementing the necessary functions for the TOE. There are no general computing capabilities available to the users of the TOE and the TOE is only accessible through the CLI. The active measures the TOE takes to protect itself include: – self-tests at bootup to assert correct functioning of the cryptographic functions and other critical parts of the TOE; – implementation of an NTP synchronized clock which the TOE uses for creating reliable timestamps; – secure storage of passwords, cryptographic keys and CSPs in a manner which prevents them from being read by any unauthorized user or process; and – an update mechanism which allows the developer to issue upgrades to the TOE software, and protection of the upgrades so that the user of the TOE can verify the authenticity of the upgrade prior to installing it.
Trusted Paths and Channels	The TOE implements secure accesses for the administrators to manage the TOE remotely and secure protocols for connecting the TOE to external IT systems. The administrators may connect to the TOE from a remote management station using SSH. The CLI is made accessible over SSH to successfully identified and authenticated administrators. The TOE uses HTTPS for connecting to a CRL Server, to a OCSP Server, for allowing the Web client to connect to the TOE in a secure manner for non-administrative functions, and for the Client Certificate authentication. The TOE uses TLS for a secure connection to a remote syslog server, to a LDAP server, and to an RSA SecurID Authentication Manager.

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, Date: 23-March-2020 (CPP_ND_V2.2E).

The ST claims conformance to the following Protection Profile Modules:

- None.

The ST claims conformance to the following Protection Profile Configurations:

- None.

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

2.2 Conformance Claim Rationale

2.2.1 TOE Type Consistency Rationale

The TOE is a non-virtual and non-distributed network appliance. It implements a set of security features required for exact conformance with CPP_ND_V2.2E. 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 CPP_ND_V2.2E.

2.2.2 Security Problem Definition Consistency

The statement of the Security Problem Definition in this ST is reproduced exactly from the CPP_ND_V2.2E. Those Security Problem Definition elements which are only applicable to virtual and distributed Targets of Evaluation 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 the CPP_ND_V2.2E.

2.2.3 Security Objective Consistency

The statement of the Security Objectives in this ST is reproduced exactly from the CPP_ND_V2.2E. There are no additional Security Objectives included in the statement of the Security Objectives. Those security objectives which are only applicable to virtual or distributed Targets of Evaluation 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 CPP_ND_V2.2E. 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 CPP_ND_V2.2E, and the ST ensures the consistency of the security functional requirements.

The security assurance requirements are drawn from the CPP_ND_V2.2E only. This ensures the consistency of the security assurance requirements.

2.3 Technical Decisions

The Technical Decisions (TD) applicable to the CPP_ND_V2.2E 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 CPP_ND_V2.2E

TD	Description	Applicable	Exclusion Rationale (if applicable)
TD 0800	Updated NIT Technical Decision for IPsec IKE/SA Lifetimes Tolerance	No	The TOE 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 network device.
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 network device.
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 TOE 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 for RFC 5077 question	Yes	
TD 0555	NIT Technical Decision for RFC Reference incorrect in TLSS Test	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 TOE does not claim DTLS.
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 CPP_ND_V2.2E. There are no additions or omissions, and the wording of each threat statement is taken verbatim. The statement of threats is given in Table 5.

Table 5 Threats

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.

3.2 Assumptions

The assumptions applicable to the TOE are drawn from the CPP_ND_V2.2E. There are no additions or omissions, and the wording of each assumption statement is taken verbatim. The assumptions are stated in Table 6.

Table 6 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. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are expected to fully validate (e.g. offline verification) any CA certificate (root CA certificate or intermediate CA certificate) loaded into the TOE's trust store (aka 'root store', ' trusted CA Key Store', or similar) as a trust anchor prior to use (e.g. offline verification).
A.REGULAR_UPDATES	The Network Device firmware and software is assumed to be updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
A.ADMIN_CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the Network Device are protected by the platform on which they reside.
A.RESIDUAL_INFORMATION	The Administrator must ensure that there is no unauthorized access possible for sensitive residual information (e.g. cryptographic keys, keying material, PINs, passwords etc.) on networking equipment when the equipment is discarded or removed from its operational environment.

3.3 Organizational Security Policies

The Organizational Security Policies (OSP) applicable to the TOE are drawn from the CPP_ND_V2.2E. There are no additions or omissions, and the wording of each OSP statement is taken verbatim.

Table 7 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 CPP_ND_V2.2E. Therefore, there are none stated in the ST, either.

4.2 Security Objectives for the Operational Environment

The security objectives for the operational environment are drawn from the CPP_ND_V2.2E. The security objectives for the operational environment are drawn in verbatim and are stated in Table 8.

Table 8 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. For TOEs supporting X.509v3 certificate-based authentication, the Security Administrator(s) are assumed to monitor the revocation status of all certificates in the TOE's trust store and to remove any certificate from the TOE's trust store in case such certificate can no longer be trusted.
OE.UPDATES	The TOE firmware and software is updated by an Administrator on a regular basis in response to the release of product updates due to known vulnerabilities.
OE.ADMIN_CREDENTIALS_ SECURE	The Administrator's credentials (private key) used to access the TOE must be protected on any other platform on which they reside.
OE.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 CPP_ND_V2.2E. Therefore, the security objectives rationale is directly applicable to the ST. It is 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.

5.1 Extended Components Definition

The ST references several extended components. Each one is taken verbatim from the CPP_ND_V2.2E. The statement of the extended components is exactly as in Appendix C of the CPP_ND_V2.2E. it is not repeated here.

5.2 Notation and Conventions

This ST follows the conventions of the CPP_ND_V2.2E 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 CPP_ND_V2.2E uses an alternative notation or expression in the statement of a Security Functional Requirements, that notation or expression is followed in the ST even if deviating from the above conventions. For example, the capitalization of the component names is followed in verbatim even if sometimes inconsistent.

The notation for expressing the Security Assurance Requirements is taken verbatim from the CPP_ND_V2.2E.

5.3 Security Functional Requirements

Class FAU: Security Audit

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 9*.*

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

a) Date and time of the event, type of event, subject identity, and the outcome (success or failure) of the event; and

b) For each audit event type, based on the auditable event definitions of the functional components included in the cPP/ST, *information specified in column three of* Table 9.

Table 9 Security Functional Requirements and Auditable Events

Requirement	Auditable Events	Additional Audit Record Contents
FAU_GEN.1	None	None.
FAU_GEN.2	None	None
FAU_STG_EXT.1	None	None
FCS_CKM.1	None	None
FCS_CKM.2	None	None
FCS_CKM.4	None	None
FCS_COP.1/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 of 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.2	Failure to establish an SSH session	Reason for failure
FCS_TLSC_EXT.1	Failure to establish an TLS session	Reason for failure

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

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.

5.3.1 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;*

] ~~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[1] The TSF shall **perform** cryptographic **key establishment** in accordance with a specified cryptographic key **establishment** method: [

- *Elliptic curve-based key establishment schemes that meet the following: NIST Special Publication 800-56A Revision 2, “Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography”;*
- *Finite field-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, RFC 7919].*

] ~~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 [[<u>4</u>]-pass] overwrite consisting of [a pseudo-random pattern using the TSF’s RBG, 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 [

- *RSA Digital Signature Algorithm and cryptographic key sizes (modulus) [<u>2048 bits, 3072 bits, 4096 bits</u>],*
- *Elliptic Curve Digital Signature Algorithm and cryptographic key sizes [<u>256 bits, 384 bits, 521 bits</u>]*

]

that meet the following: [

[1] Modified as per TD0580 and TD0581

- *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-256, SHA-384, SHA-512*] ~~and cryptographic key sizes [assignment: cryptographic key sizes]~~ and **message digest sizes [*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 bits, 512 bits] **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 the HTTPS protocol 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 v3 (RFC 1305), NTP v4 (RFC 5905)*].

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

- *Authentication using [SHA256, SHA384, SHA512] 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 [*Hash_DRBG (any)*].

FCS_RBG_EXT.1.2 The deterministic RBG shall be seeded by at least one entropy source that accumulates entropy from [*[0] software-based noise source, [4] platform-based noise source*] with a minimum of [2*56 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, 5647, 5656, 6187, 6668, 8268, 8308 section 3.1, 8332*].

FCS_SSHC_EXT.1.2[2] The TSF shall ensure that the SSH protocol implementation supports the following user authentication methods as described in RFC 4252: public key-based, [*password-based*].

FCS_SSHC_EXT.1.3 The TSF shall ensure that, as described in RFC 4253, packets greater than [*262,144*] 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: [*aes256-ctr, aes256-gcm@openssh.com*].

FCS_SSHC_EXT.1.5 The TSF shall ensure that the SSH public-key based authentication implementation uses [*ssh-rsa, 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 [*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, [*4344, 5656, 6187, 6668, 8268, 8308 section 3.1, 8332*].

FCS_SSHS_EXT.1.2[3] 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 [*262,144*] 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: [*aes256-ctr, aes256-gcm@openssh.com*].

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

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

[2] Modified as per TD0636

[3] Modified as per TD0631

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_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 cipher suites:

- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 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_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 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_SHA384 as defined in RFC 5289

and no other cipher suites.

FCS_TLSC_EXT.1.2 The TSF shall verify that the presented identifier matches [the reference identifier per RFC 6125 section 6, IPv4 address in CN or SAN, IPv6 address in the CN or SAN].

FCS_TLSC_EXT.1.3 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the server 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 Groups Extension with the following curves/groups: [secp256r1, secp384r1, secp521r1] and no other curves/groups] in the Client Hello.

FCS_TLSS_EXT.1 TLS Server Protocol 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_128_CBC_SHA256 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_256_GCM_SHA384 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 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_SHA384 as defined in RFC 5289] 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 size [2048 bits, 3072 bits, 4096 bits*], ECDHE curves [*secp256r1, secp384r1, secp521r1*]]].

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].

FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication

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

FCS_TLSS_EXT.2.2 When establishing a trusted channel, by default the TSF shall not establish a trusted channel if the client certificate is invalid. The TSF shall also [

- *require administrator authorization of establish the connection if the TSF fails to [match the reference identifier] of the presented client certificate*

].

FCS_TLSS_EXT.2.3 The TSF shall not establish a trusted channel if the identifier contained in a certificate does not match an expected identifier for the client. If the identifier is a Fully Qualified Domain Name (FQDN), then the TSF shall match the identifiers according to RFC 6125, otherwise the TSF shall parse the identifier from the certificate and match the identifier against the expected identifier of the client as described in the TSS.

5.3.2 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 10*] unsuccessful authentication attempts occur related to *Administrators attempting to authenticate remotely using a password*.

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

FIA_PMG_EXT.1 Password Management

FIA_PMG_EXT.1.1 The TSF shall provide the following password management capabilities for administrative passwords:

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

b) Minimum password length shall be configurable to between [*6*] and [*64*] 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, [RSA SecurID]*] 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;
- [*automated generation of cryptographic keys, [display the End User License Agreement (EULA), establish SSH connection with the TOE, establish TLS connection with the TOE, respond to ICMP echo request]*].

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, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, Certificate Revocation List (CRL) as specified in RFC 5759 Section 5*].
- The TSF shall validate the extendedKeyUsage field according to the following rules:
  - *Certificates used for trusted updates and executable code integrity verification shall have the Code Signing purpose (id-kp 3 with OID 1.3.6.1.5.5.7.3.3) in the extendedKeyUsage field.*
  - *Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.*
  - *Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.*
  - *OCSP certificates presented for OCSP responses shall have the OCSP Signing purpose (id-kp 9 with OID 1.3.6.1.5.5.7.3.9) in the extendedKeyUsage field.*

FIA_X509_EXT.1.2/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 [*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.

5.3.3 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_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_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 configure audit behaviour (e.g. changes to storage locations for audit; changes to behaviour when local audit storage space is full);
    - Ability to modify the behaviour of the transmission of audit data to an external IT entity;
    - Ability to manage the cryptographic keys;
    - Ability to re-enable an Administrator account;
    - Ability to set the time which is used for time-stamps;
    - Ability to configure NTP;
    - Ability to manage the TOE's trust store and designate X509.v3 certificates as trust anchors;
    - Ability to import X.509v3 certificates to the TOE's trust store;
    - Ability to manage the trusted public keys database[4]].

FMT_SMR.2 Restrictions on Security Roles

FMT_SMR.2.1 The TSF shall maintain the roles:

- *Security Administrator.*

FMT_SMR.2.2 The TSF shall be able to associate users with roles.

FMT_SMR.2.3 The TSF shall ensure that the conditions

[4] As per TD0631

- *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.

5.3.4 Class FPT: Protection of the TSF

FPT_STM_EXT.1 Reliable Time Stamps

FPT_STM_EXT.1.1 The TSF shall be able to provide reliable time stamps for its own use.

FPT_STM_EXT.1.2 The TSF shall [*allow the Security Administrator to set the time, synchronise time with an NTP server*].

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_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)] to demonstrate the correct operation of the TSF: [WolfSSL Library functions, WolfSSL software integrity].

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 [*the most recently installed version of the TOE firmware/software*].

FPT_TUD_EXT.1.2 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.

5.3.5 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.

5.3.6 Class FTP: Trusted Path/Channels

FTP_TRP.1/Admin Trusted Path (refinement)

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

FTP_ITC.1 Inter-TSF Trusted Channel

FTP_ITC.1.1 The TSF shall **be capable of using [*SSH, TLS, HTTPS*] to** provide a trusted communication channel between itself and **authorized IT entities supporting the following capabilities: audit server, [*authentication server, [CRL Server, OCSP Server, Web client, another instance of a TOE]]*** 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 [*storing audit data outside of the TOE, authentication of a user with an authentication server, requesting a CRL, Requesting Certificate status from an OCSP Responder, Operation of the TOE remotely*].

Application note: The authentication server may be a LDAP Server or an RSA SecurID Authentication Manager. When the RSA SecurID is used for authentication, the LDAP Server is also used for authorization.

5.4 Security Assurance Requirements

This section states the Security Assurance Requirements. The applicable Security Assurance Requirements are stated in Table 10.

Table 10 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 CPP_ND_V2.2E 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 CPP_ND_V2.2E 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 CPP_ND_V2.2E are implemented. Therefore, the Security Functional Rationale of the CPP_ND_V2.2E is directly applicable to the ST and is not repeated.

The Security Assurance Requirements are drawn from the CPP_ND_V2.2E. None. are added or removed. Therefore, the Security Assurance Requirements Rationale of the CPP_ND_V2.2E 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 assurance requirements, and how the developer and the evaluator fulfill the security functional 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 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 11.

Table 11 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 CPP_ND_V2.2E. 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. CPP_ND_V2.2E only defines security requirements for the operational environment of the TOE. – Extended Components Definition and the statement of the security requirements state exactly the Security Functional Requirements and the Security Assurance Requirements the TOE fulfills. – TOE Summary Specification which describes for each Security 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 CPP_ND_V2.2E. The evaluator documents the testing in accordance with the requirements stated in CPP_ND_V2.2E 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 CPP_ND_V2.2E and the Common Criteria evaluation and certification scheme followed.

6.2 Fulfillment of the Security Functional Requirements

The fulfillment of the security functional requirements is given in Table 12.

Table 12 Fulfillment of the Security Functional Requirements

Security Functional Component	**Fulfillment**
FAU_GEN.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 9: – 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; – 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 9.

	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.
FAU_STG_EXT.1	The TOE implements a TLS client and server. Each instance of a TOE is a standalone appliance which generates and stores the audit log entries and stores them locally. The log files are not protected by cryptographic means but there are no management functions or other means for modifying them. Audit records on the TOE are stored as syslog entries and the TOE is capable of establishing a TLS connection with an external audit server and to forward the syslog entries to the audit server over the TLS connection. The TOE stores the syslog entries in a disc partition of a fixed size defined at the provisioning of the TOE. 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. While operating, the TOE will attempt to send audit logs to an audit server in real time. The log entries are added into a buffer which is consumed by the process forwarding the entries to the audit server over TLS. If for any reason the buffer fills up without being consumed, any new entry shall replace the oldest entry until the buffer consumption commences.
FCS_CKM.1 **FCS_CKM.2**	The TOE uses RSA and ECC as asymmetric algorithms for SSH and TLS authentication keys and key establishment keys. The RSA keys are at least 2048 bits in size. The ECC keys are generated on P-256, P-384 and P-521. RSA keys are generated using the RSA scheme as defined in FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.3. ECC keys are generated in accordance with FIPS PUB 186-4, “Digital Signature Standard (DSS)”, Appendix B.4. The keys are established in accordance with the following standards: – 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”, – Finite field-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", and – FFC Schemes using 'safe-prime' groups that meet the following: "NIST Special Publication 800-56A Revision 3, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography", RFC 3526 and RFC 7919. SSH key establishment keys are generated and exchanged in accordance with Diffie-Hellman on group 14, specifically using diffie-hellman-group14-

<table>
<tr><td></td><td>sha1 as defined in RFC 4253 and diffie-hellman-group-exchange-sha256 on Diffie-Hellman Group 14 as defined in RFC 4419.
Diffie-Hellman Group 14 is defined in RFC 3526. 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><b>FCS_CKM.4</b></td><td>The TOE implements a method to destroy all cryptographic keys which are no longer used:
– Plaintext keys stored in the volatile storage are destroyed by the TOE executing a single overwrite by zeroes.
– Plaintext keys stored in the non-volatile storage are destroyed by the TOE executing a 4-pass overwrite of the storage area in which the key resides. The overwrite consists of three rounds of overwriting with random quantities generated by the DRBG of the TOE followed by a single overwrite of zeroes.
Each cryptographic key and CSP the TOE uses is identified in Table 13. The table also describes for each cryptographic key and CSP the purpose of the key or CSP, and how the key or CSP is stored by the TOE. For each cryptographic key and CSP that is destroyed throughout the life-cycle of the TOE, the table states how the key or CSP is destroyed.</td></tr>
<tr><td><b>FCS_COP.1/DataEncryption</b></td><td>The TOE implements AES for the symmetric encryption and decryption of data on SSH and TLS connections. AES is used in CBC, CTR and GCM modes. The CBC mode is in accordance with ISO 10116, the CTR mode is in accordance with ISO 10116, and the GCM mode is in accordance with ISO 19772. Key sizes supported are 128 bits and 256 bits. Each key size is used for each mode.</td></tr>
<tr><td><b>FCS_COP.1/SigGen</b></td><td>The TOE uses RSA and ECDSA for the generation of digital signatures. RSA digital signatures are generated with a 2048-bit, 3072-bit and 4096-bit keys (modulus). ECDSA signatures are generated with 256-bit, 384-bit and 512-bit keys.</td></tr>
<tr><td><b>FCS_COP.1/Hash</b></td><td>Hash functions are used by the TOE for NTP time stamp authentication, protection of the passwords, SSH and TLS, and for the verification of the authenticity of TOE software upgrades:
The hash functions are used by the TOE for the following purposes:
<table>
<tr><th></th><th>SHA-256</th><th>SHA-384</th><th>SHA-512</th></tr>
<tr><td>NTP</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>Protection of passwords when stored on the TOE</td><td></td><td></td><td>X</td></tr>
<tr><td>SSH RSA Key Agreement</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>SSH ECC Key Agreement</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>RSA Signature generation and verification</td><td>X</td><td>X</td><td>X</td></tr>
</table>
</td></tr>
</table>

<table>
<tr>
<td></td>
<td>
<table>
<tr><td>ECDSA Signature generation and verification</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>TLS</td><td>X</td><td>X</td><td>X</td></tr>
<tr><td>File system integrity self-tests</td><td>X</td><td></td><td></td></tr>
<tr><td>Firmware integrity self-test</td><td>X</td><td></td><td></td></tr>
</table>
</td>
</tr>
<tr>
<td><b>FCS_COP.1/KeyedHash</b></td>
<td>The HMAC algorithms used by the TOE are detailed in the following:
<table>
<tr><th></th><th>HMAC-SHA-256</th><th>HMAC-SHA-384</th><th>HMAC-SHA-512</th></tr>
<tr><td>Key length</td><td>256 bits</td><td>512 bits</td><td>512 bits</td></tr>
<tr><td>Hash function</td><td>SHA-256</td><td>SHA-384</td><td>SHA-512</td></tr>
<tr><td>Block size</td><td>512 bits</td><td>1024 bits</td><td>1024 bits</td></tr>
<tr><td>Output size</td><td>256 bits</td><td>384 bits</td><td>512b its</td></tr>
</table>
</td>
</tr>
<tr>
<td><b>FCS_RBG_EXT.1</b></td>
<td>The TOE generates random bits in accordance with NIST Special Publication 800-90 using CTR_Hash (Any). The RBG does not require any configuration and is seeded from four platform-based entropy sources. No software-based entropy sources are used by the TOE.
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/random. The randomness from /dev/random is read and used as a seed by the wolfSSL DRBG. 256 bits of entropy is read from /dev/random and used for seeding the wolfSSL DRBG. The wolfSSL DRBG is then used by the TOE for the run-time generation of random bits as required.
Full details of the entropy sources and their use in the generation of random bits are given in a separate Entropy Assessment Report.</td>
</tr>
<tr>
<td><b>FCS_HTTPS_EXT.1</b></td>
<td>The HTTPS Server of the TOE is the Apache 2.4.34 server on Red Hat Linux. When booting up to a FIPS mode, HTTP is dropped and only HTTPS is allowed.</td>
</tr>
<tr>
<td><b>FCS_TLSC_EXT.1</b>
<b>FCS_TLSS_EXT.1</b></td>
<td>The TOE implements a TLS Client and a TLS Server for trusted channels. 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 Client and a TLS Server:
– TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256 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_256_GCM_SHA384 as defined in RFC 5289
– TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289
– TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289</td>
</tr>
</table>

	– TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289 – TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289 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), 80 (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, IPv4 address in CN or SAN, or IPv6 address in the CN or SAN. 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. Key establishment for TLS is with RSA or ECDHE. RSA key sizes 2048 bits, 3072 bits and 4096 bits are supported. 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.
FCS_TLSS_EXT.2	The TOE implements TLS with mutual authentication. When presented with an X.509 client host certificate, the TOE validates the certificate per the rules stated in FIA_X509_EXT.1/Rev. The TOE verifies that the presented identifier matches the reference identifier in the certificate. The TOE supports Common Name (CN) and Subject Alternative Name (SAN) reference identifiers that use IPv4 or IPv6 address values. Canonical formatting according to RFC 3986 is enforced. The TOE does not support URI, DNS (FQDN), service name reference identifiers, wildcards or pinned certificates. The IP address obtained from the certificate is converted from ASN.1 to the binary representation of the textual string of the IP address. The IP Address is also converted from the IP address from the established network connection to a binary representation of the textual string of the IP address. The two representations are compared to determine the action to be performed next. The check is carried out as follows:

	– If the SAN value exists: o If the two values match, revocation check is performed. o If the two values do not match, the certificate is deemed invalid. – If the SAN field is not used (non-existent), the representation of the CN value is used for comparison instead: o If the two values match, revocation check is performed. o If the two values do not match, the certificate is deemed invalid. An audit log entry is generated for the outcome of the certificate validation. The TOE may be configured to allow establishment of a temporary secure channel when the certificate is deemed invalid. The configuration is done by the external LDAP server to match one of the supported fields (UPN, UID, email, or CN). The TLS session established only reports an error message if the LDAP authorization fails.
FCS_SSHC_EXT.1 **FCS_SSHS_EXT.1**	The TOE implements a SSH Client and a SSH Server. Trusted paths between the TOE and a remote management station are protected with SSH. The SSH Client and the SSH Server are matching in that a nearly identical set of cryptographic suites are implemented. This assures of compatibility in scenarios where one instance of a TOE acts as a SSH client and another instance as a SSH server. The exception is where the SSH Server implements a restricted set of public key authentication algorithms. The TOE uses a 2048-bit or larger 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 management functions of the TOE. The Administrator may configure SSH to use public key based authentication or password based authentication. For public key based authentication, the TOE maintains a local database, the known_hosts file, of the IP addresses and public keys of known SSH Servers. Each SSH Server is identified by the SSH Client of the TOE by the IP address and the public key. The administrator may modify the file by adding or removing entries (i.e. known SSH Servers). When a SSH connection is requested by a SSH Client of the TOE, the SSH Server to which the connection was requested presents to the TOE its public key and the IP Address. The SSH Client of the TOE matches the presented public key and the IP Address of the SSH Server against its known_hosts file. If the IP address and the public key of the SSH Server exist in the known_hosts file, the TOE accepts the SSH Server and proceeds with the connection establishment. If there is no match in the known_hosts file, the TOE shall not allow the connection. The TOE implements all mandatory cryptographic algorithms and methods and only accepts public-key based authentication. Multiple authentication mechanisms for users are not required. Port forwarding and sessions to clients are allowed. X11 forwarding is prohibited. Message authenticity is achieved by hmac-sha2-256, hmac-sha2-512 and implicit] MAC algorithms. The TOE rejects any other MAC algorithms. MAC

	algorithm "implicit" is used when the encryption is in the GCM model and includes message authentication. In that case, the TOE omits the explicit MAC field in the message. The TOE does not accept the “none” cipher and implements aes256-ctr and aes256-gcm@openssh.com for the protection of data in transport. The TOE rekeys every 1Gb bytes 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 cannot be directly configured (i.e. cannot be configured using the sshd_config knob). When a connection is brought down, the TOE does not attempt to re-establish it. Key exchange is performed only using the supported key exchange algorithms ordered as follows: ecdh-sha2-nistp256, diffie-hellman-group14-sha256, diffie-hellman-group16-sha512, diffie-hellman-group18-sha512, ecdh-sha2-nistp384, and ecdh-sha2-nistp521. 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. 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. Authentication method "none" is not allowed. The TOE responds to it with a list of permitted authentication methods. The SSH Client implements public key authentication for SSHv2 session authentication using ssh-rsa, rsa-sha2-256, rsa-sha2-512, ecdsa-sha2-nistp256, ecdsa-sha2-nistp384, and ecdsa-sha2-nistp521. The SSH Server implements SSHv2 session authentication using rsa-sha2-256 and ecdsa-sha2-nistp521. Authentication succeeds if the correct private key is used. 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 reads the packet payload size in the TCP packet to determine the packet length. Packets greater than 256K (i.e. 262,144) bytes are dropped and the connection is terminated. Negotiation of HMAC-SHA1 in each direction for SSH transport is allowed. Diffie-hellman-group14-sha1 is supported. Key re-exchange is performed when SSH_MSG_KEXINIT is received. The TOE implements aes256-ctr and aes256-gcm@openssh.com for encryption. The TOE does not implement the recommended modes AES192-ctr or 3des-ctr. None of the optional modes are implemented.

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. The administrator may define the maximum value to be between one and ten. 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 or when another administrator unlocks the locked account using the relevant CLI command. The administrator may define the locking period to be between 60 and 3600 seconds. The default value is 900 seconds.
FIA_PMG_EXT.1	Authentication data for fixed password authentication is a case-sensitive, alphanumeric value. The password has a minimum length which the administrator may configure to be between 6 and 64 characters. 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: “!”, “@”, “#”, “$”, “%”, “^”, “&”, “*”, “(“, “)”, "_" and "+".
FIA_UAU.7	When authenticating from a remote management station, the TOE shall not echo the characters entered by the user: – When authenticating from console using the CLI, nothing is echoed on console. – When authenticating from a remote management station using SSH, what is displayed by the SSH Client is displayed on the remote management workstation. 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_UIA_EXT.1 **FIA_UAU_EXT.2**	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. The TOE may also be configured to use RSA SecurID tokens for additional authentication. The TOE stores a SHA-512 digest of a reference password created 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 CLI 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. 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 may display the End User License Agreement (EULA) governing the use of the TOE or parts thereof. 4. The TOE allows establishment of SSH and TLS connections between itself and remote IT products. The remote IT product may be an audit server or a remote management station. Users can connect to the TOE by two means: 1. 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. 2. From a remote management workstation over SSH, in which case 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. Authentication of the remote IT products may be through SSH and pre-distributed public keys or over TLS. TLS keys are validated using X.509 certificates.
FIA_X509_EXT.1 **FIA_X509_EXT.2** **FIA_X509_EXT.3**	The TOE uses X.509 certificates as defined in RFC 5280. X.509 certificates are used for the validation of public keys in TLS and HTTPS. The Administrator may request generation of a certificate. There are several ways to do this: – Use internally generated self-signed certificate. – Load an externally generated certificate. – Use a CSR to generate a request for an external certificate that is then loaded back in. The generation may be by a sequence of the following CLI commands. Generation of a self-signed certificate: `crypto certificate name <cert-name> generate self-signed [key-size-bits <bits>] [serial-num <serial-number>] [days-valid <days>] [common-name <common-name>] [country-code <country-code>] [state-or-prov <state-or-prov-name>] [locality <locality-name>] [organization <organization-name>] [org-unit <org-unit-name> | <org unit list>] [email-addr <email-addr>] [comment <comment>] [cert-key-type rsa|ecdsa] [curve <ecdsa curve>] [san-dns <FQDN list>]`

Loading an existing certificate:

```
crypto certificate name <cert-name> public-cert [pem
<pem-string>|file <pem-filename>|url <pem file url>]
[comment <comment>]
```

Copy and paste the private key:

```
crypto certificate name <cert-name> prompt-private-key
```

With the request, the Administrator supplies the following values:

- Certificate-id: The identifying string for this certificate
- Domain-name
- Email address
- IP address
- Subject (DC=<Domain component>,CN=<Common-Name>,OU=<Organizational-Unit-name>,O=<Organization-name>,SN=<Serial-Number>,L=<Locality>,ST=<state>,C=<Country>)
- Filename: The local file in which the certificate signing request is stored.

When validating a certificate, the TOE extracts the subject, issuer, subjects public key, signature, basicConstraints and validity period fields:

- If any of the 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.
- The TOE confirms that the current date and time is within the validity time period specified in the certificate. If the date and time are outside the validity time, the validation fails.
- The TOE 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). If the value is correct, the validation fails.

The TOE may be configured to check for the revocation of the certificates using Online Certificate Status Protocol (OCSP) as specified in RFC 6960, a Certificate Revocation List (CRL) as specified in RFC 5280 Section 6.3, or a Certificate Revocation List (CRL) as specified in RFC 5759 Section 5. The administrator may also configure the TOE to not check the revocation status of the certificates.

The following steps take place when validating the certificate revocation status using a CRL:

- The CRL is downloaded at a frequency configurable by an administrator.
- The CRL is cached immediately for use in the determination of the revocation status of the presented certificate.
- The CRL is cached until the TOE has successfully downloaded a newer CRL from the CRL Distribution Point (CDP). The newly downloaded CRL replaces the cached CRL.

	– If the TOE fails to establish a connection to the CDP, it continues automatically to attempt downloading a new CRL at regular intervals until successful. If the TOE is configured to perform a revocation check and the revocation fails because of a connection problem, the certificate is considered to have failed validation and is rejected. Yet, the TOE may be configured to allow the establishment of TLS connections even when the certificate validation fails. The TOE validates a certificate path by building a chain of at least three certificates based upon issuer and subject linkage. Each link is validated in accordance with the certificate validation procedure described above. If any certificate in the chain fails the 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 when validating certificates for TLS and HTTPS connections. Common Name, Organization, Organizational Unit, Country and public key details are provided in the CSR. The TOE validates the chain of certificates from the Root CA when the CA Certificate Response is received.
FMT_MOF.1/ManualUpdate	The TOE does not notify the Administrators of the availability of software updates. The CLI does not support the notifications. Instead, notifications are provided offline through the Endace Support Portal. The TOE does not have access to the portal but it must be accessed by the administrator using other means. Once an update is available, an Administrator (i.e. a user with sufficient credentials to gain access to the CLI) may use the CLI to manually update the TOE software. Users with no Administrative privileges are not granted access to the CLI and cannot update the TOE software. The updating of the TOE software is done as described under FPT_TUD_EXT.1.
FMT_MTD.1/CoreData	The TOE predominantly only allows access to the TOE functions to successfully authenticated users assigned to an Administrator role. There are, nevertheless, the following exceptions which are the only functions available to the users prior to a successful identification and authentication as a legitimate Administrator: 1. Displaying the access banner. The administrators may configure an access banner which is displayed to the users when attempting to use the TOE locally or remotely. The access banner is only displayed and does not allow any means of entering data or manipulating the TOE. 2. Responding to ICMP Echo. 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. 3. SSH connection between the TOE and a remote management station. SSH will make available to the remote user an authentication window in which the remote user may authenticate as a legitimate

	Administrator. Access to the CLI shall only be made available upon successful identification and authentication. SSH cannot be used for issuing commands to the TOE. 4. The EULA may also be displayed prior to administrator access being granted to the user. The TOE protects access to the trust store by two means. First, the trust store may only be accessed by privileged processes. Second, there are no CLI functions which allow the administrator direct access to the objects in the trust store. The trust store is only accessible by the privileged processes which are part of the TOE. There is no general purpose software or processes running in the TOE which restricts access to the trust store only to the trusted TOE processes.
FMT_SMF.1	The TOE implements a CLI where a command exists for each management and configuration function of the TOE. There are no other methods of management of the TOE. The CLI commands may be issued by two 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 and the Administrator uses the CLI to administer the TOE. The CLI is made available to the successfully authenticated administrators in entirety. There are no CLI subsets made available to roles other than Administrator.
FMT_SMR.2	The TOE only implements a single role, Security Administrator. Security Administrator is the only role to which the CLI is made available, i.e. which is authorized to administer the TOE. 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 CLI. Human users are identified and authenticated with a username and password. If the identification and authentication is successful, the user is assigned a role Security Administrator. The role assignment remains until the session is terminated.
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. The CLI does not implement any functions for reading or exporting the passwords or representations thereof.

FPT_SKP_EXT.1	The TOE stores the cryptographic keys and CSPs as described in Table 13. Cryptographic keys are only accessed by authorized processes. There are no methods available for users to execute unauthorized processes on the TOE. The CLI does not implement any means of reading or exporting the private or symmetric cryptographic keys.
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. The administrator may use the CLI or the GUI for setting the time but the TOE clock may also be configured for synchronization by 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 an administrator-defined time interval. 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. 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.
FPT_TST_EXT.1	The TOE stores the cryptographic keys and CSPs as described in Table 11 (Sect. 6.4). Cryptographic keys are only accessed by authorized processes. There are no methods available for users to execute unauthorized processes on the TOE. The CLI does not implement any means of reading or exporting the private or symmetric cryptographic keys. The following self-tests are executed by the TOE at the boot up: – Software Integrity: HMAC-SHA-256 with a 256-bit key – AES: Separate encryption and decryption KATs, CBC mode, 128-bit key – AES in GCM mode: Separate authenticated encryption and decryption KATs, GCM mode, 128-bit key – DRBG: KAT for the HASH_DRBG using SHA-256 – ECDSA: ECDSA PCT using the P-256 curve – HMAC: HMAC-SHA-512 (512-bit key), and HMAC-SHA3-256 (256-bit key) KATs – KAS ECC: Primitive "Z" computation KAT (per [140IG] 9.6), using P-256 – KAS FFC: Primitive "Z" computation KAT (per [140IG] 9.6), using L = 2048 N = 256 - RSA, Separate signature generation and signature verification KATs (k = 2048), inclusive of the embedded SHA-256, RSADP and RSAEP (KTS) 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 version. Availability of software upgrades can be queried from the Endace Support Portal. The downloading

	of the upgrade is done manually as is the commencement of the actual upgrade. TOE software is distributed as an .img file. The format allows upgrading of the packages constituting the image when distributed in RPM (Red Hat Package Manager) format. Upgrades to the TOE software are distributed in RPM format. 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-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 local and remote authentication exchange. The banner may provide warnings against unauthorized access to the TOE and any other information that the administrator wishes to communicate.
FTP_ITC.1 **FTP_TRP.1/Admin**	The TOE implements a SSH Client and a SSH Server. The TOE also implements TLS Client and a TLS Server. The TOE implements HTTPS over TLS. In addition to the trusted paths, the TOE allows access for non-administrative operation of the TOE using SSH. Two instances of a TOE may be connected over SSH and one may be operated through the other. TLS and HTTPS are used for secure connections to external devices as follows: – The TOE may connect to the audit server using TLS, – For authentication servers, the TOE uses TLS to connect to a LDAP Server and TLS to connect to a RSA SecurID Authentication Manager. The TOE authenticates the Client Certificates for HTTPS, and – For interactions with the CRL Server and OCSP Server, the toe uses HTTPS, Each of the above connections are established by the TOE. The TOE also allows incoming connections from Web clients for non-administrative operation of the TOE. Those connections are protected by HTTPS over TLS.
FTP_TRP.1/Admin	The TOE implements a SSH Server for the administrator to connect to the TOE remotely and administer the TOE over the CLI. The CLI implements an authentication function. Once the connection is established, the Administrator is authenticated using a username and a

	password. Once successfully authenticated, the administrator accesses the CLI from the remote management workstation. Each command and response thereof is protected by SSH.

6.3 Cryptographic Details

This section first summarizes the zeroization of cryptographic keys and Critical Security Parameters. That is followed by the CAVP References for the cryptographic algorithms used by the TOE.

6.3.1 Key and CSP Zeroization

The timing and method of the zeroization of the cryptographic keys and critical security parameters (CSP) used by the TOE is given in Table 13.

Table 13 Timing and Method of the Zeroization of Cryptographic Keys and Critical Security Parameters

Key/CSP	Storage Memory	Destruction Method	Remarks
Certificates	Volatile	Zeroize on function exit	
Certificates	Non-Volatile	Overwrite with DRBG and zeroize on factory reset	Stored on TOE system disk
Passwords	Volatile	Zeroize on function exit	
Service Keys	Volatile	Zeroize on function exit	
SSH Keys	Volatile	Zeroize on function exit	
SSH Keys	Non-Volatile	Overwrite with DRBG and zeroize on factory reset	Stored on TOE system disk. Holds public keys
User hashed passwords	Non-Volatile	Overwrite with DRBG and zeroize on factory reset	Stored on TOE system disk
web request cookies	Volatile	Free cookie value, zeroize after session ends	

6.3.2 CAVP Certificate References

All cryptography of the TOE is implemented on the Endace Crypto Firmware 2.1 included in the CAVP validation of the wolfCrypt v5.2.1 software. The validation certificate reference is A4308. The CAVP validation has been carried out on the following CPUs, some of which are included in different variants of the TOE:

- Intel® Xeon® Gold 5418N CPU @1.80GHz with AES-NI (PAA), included in the EP-94C8-G5.
- Intel® Xeon® Gold 6230N CPU @2.30GHz with AES-NI (PAA), included in the EP-92C8-G4.
- Intel® Xeon® Gold 6338N CPU @2.20GHz with AES-NI (PAA), included in the EP-2184-G5.
- Intel® Xeon® Gold 6338N CPU @2.20GHz without AES-NI (PAA)
- Intel® Xeon® Silver 4316 CPU @2.30GHz with AES-NI (PAA), included in the EP-2144-G5.
- Intel® Xeon® Silver 4316 CPU @2.30GHz without AES-NI (PAA)

7 Acronyms

AES	Advanced Encryption Standard
AES-NI	Advanced Encryption Standard New Instructions
ANSI	American National Standards Institute
API	Application Programming Interface
ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
ASN.1	Abstract Syntax Notation One
CBC	Cipher Block Chaining
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CDP	CRL Distribution Point
CFP	C Form-factor Pluggable
CLI	Command Line Interface
CMVP	Crypto Module Validation Program
CN	Common Name
cPP	collaborative Protection profile
CRL	Certificate Revocation List
CTR	Counter Mode
CSP	Critical Security Parameter
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generation
DSS	Digital Signature Standard
EAL	Evaluation Assurance Level
ECC	Elliptic Curve Cryptosystem
ECDSA	Elliptic Curve Digital Signature
EP	Extended Package OR Endace Probe
ESP	Encapsulating Security Payload
EULA	End User License Agreement
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standard

FQDN	Fully Qualified Domain Name
GbE	Giga bit Ethernet
GBPS	Giga Bits Per Second
GCM	Galois Counter Mode
GPG	GNU Privacy Guard
GPS	Global Positioning System
GUI	Graphical User Interface
HMAC	Keyed-Hash Message Authentication Code
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Security
IA	Identification and Authentication
ICMP	Internet Control Message Protocol
ID	Identity
iDRAC	intelligent Dell Remote Access Controller
IEC	International Electrotechnical Commission
IETF	Internet Engineering Task Force
IKE	Internet Key Exchange
IP	Internet Protocol
IPMI	Intelligent Platform Management Interface
IPsec	Internet Protocol Security
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
ISO	International Organization for Standardization
IT	Information Technology
KAT	Known Answer Test
LDAP	Lightweight Directory Access Protocol
MAC	Message Authentication Code or Media Access Code
MbE	Mega bit Ethernet
NAT	Network Address Translation
NetOps	Network Operations
NFV	Network Function Virtualization

NIST	National Institute of Standards and Technology
NTP	Network Time Protocol
OCSP	Online Certificate Status Protocol
OID	Object Identifier
OSI	Open Systems Interconnect
PAA	Parallel Algorithms and Architectures
PAM	Pluggable Authentication module
PCT	Public Component Test
PIC/PIM	Physical Interface Card / Physical Interface Module
PID	Process Identity
PKCS	Public Key Cryptography Standard
PKI	Public Key Infrastructure
PoE	Power over Ethernet
PPS	Packets Per Second
PRNG	Pseudo Random Number Generator
RADIUS	Remote Authentication Dial-In User Service
RBG	Random Bit Generation
RE	Routing Engine
RFC	Request For Comments
RNG	Random Number Generator
RPM	Red Had Package Manager
RSA	Rivest-Shamir-Adleman
SA	Security Association
SAN	Subject Alternative Name
SDN	Software Defined Networking
SecOps	Secure Operations
TACACS	Terminal Access Controller Access-Control System
TACACS+	TACACS Plus
TFTP	Trivial File Transfer Protocol