Tenable Security Center 6.2.0

Security Target

Version 1.1

10 October 2023

Prepared for:

Tenable, Inc.
7021 Columbia Gateway Dr.
Columbia, MD 21046

Prepared by:

Accredited Testing and Evaluation Labs
6841 Benjamin Franklin Drive
Columbia, MD 21046

Contents

Tables

1 Introduction

This section introduces the Target of Evaluation (TOE) and provides the Security Target (ST) and TOE identification, ST and TOE conformance claims, ST conventions, glossary, and list of abbreviations.

The TOE is Security Center (formerly Tenable.sc) from Tenable, Inc. Security Center is application software designed to consolidate asset discovery, network monitoring, log aggregation, and vulnerability scanning into a single location to assess an organization's security posture. The ST contains the following additional sections:

- Product and TOE Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
-
- The [TLS_PKG] does contain evaluation activities for how to evaluate its SFR claims as part of the evaluation of ASE_TSS.1, AGD_OPE.1, AGD_PRE.1, and ATE_IND.1. All Security Functional Requirements specified by [TLS_PKG] are evaluated in the manner specified in that package.
- TOE Summary Specification (Section 0)
-
- The Trusted Path/Channels security function is designed to satisfy the following security functional requirements:
- FTP_DIT_EXT.1—the TOE implements TLS and HTTPS and invokes platform-provided SSH to secure data in transit between itself and its operational environment.
- Protection Profile Claims (Section 0)
-

- Rationale (Section 0)
- TOE Usage of Third-Party Components (Appendix A)

1.1 Security Target, TOE and CC Identification

ST Title – Tenable Security Center 6.2.0 Security Target

ST Version – Version 1.1

ST Date – 10 October 2023

TOE Identification – Security Center 6.2.0, supported on RHEL 8

TOE Developer – Tenable, Inc.

Evaluation Sponsor – Tenable, Inc.

CC Identification – Common Criteria for Information Technology Security Evaluation, Version 3.1, Revision 5, April 2017

1.2 Conformance Claims

This ST and the TOE it describes are conformant to the following CC specifications:

- *Protection Profile for Application Software*, Version 1.4, 7 October 2021 ([APP_PP]) with the following optional and selection-based SFRs:
  - FCS_CKM.1/AK
  - FCS_CKM_EXT.1/PBKDF
  - FCS_CKM.2
  - FCS_COP.1/SKC
  - FCS_COP.1/Hash
  - FCS_COP.1/Sig
  - FCS_COP.1/KeyedHash
  - FCS_HTTPS_EXT.1/Client
  - FCS_HTTPS_EXT.1/Server
  - FCS_HTTPS_EXT.2
  - FCS_RBG_EXT.2
  - FIA_X509_EXT.1
  - FIA_X509_EXT.2
  - FPT_TUD_EXT.2
- *Functional Package for Transport Layer Security (TLS)*, Version 1.1, 1 March 2019 ([TLS_PKG]) with the following optional and selection-based SFRs:
  - FCS_TLSC_EXT.1
  - FCS_TLSC_EXT.2
  - FCS_TLSC_EXT.5
  - FCS_TLSS_EXT.1
  - FCS_TLSS_EXT.2
- The following NIAP Technical Decisions affecting [APP_PP] apply to the TOE and have been accounted for in the ST development and the conduct of the evaluation, or were considered to be non-applicable for the reasons stated:

TD0628: Addition of Container Image to Package Format

- The ST accounts for this TD.

TD0650: Conformance claim sections updated to allow for MOD_VPNC_V2.3 and 2.4

- No change to ST; affects only PP conformance claims and the ST does not claim conformance to the relevant PP-Module.

TD0664: Testing activity for FPT_TUD_EXT.2.2

- No change to ST; affects only evaluation activities.

TD0717: Format changes for PP_APP_V1.4

- The ST accounts for this TD.

TD0719: ECD for PP APP V1.3 and 1.4

- No change to ST; adds Extended Component Definitions to PP to satisfy APE_ECD.1.

TD0736: Number of elements for iterations of FCS_HTTPS_EXT.1

- The ST accounts for this TD.

TD0743: FTP_DIT_EXT.1.1 Selection exclusivity

- The ST accounts for this TD.

TD0747: Configuration Storage Option for Android

- No change to ST; affects only evaluation activities.

TD0756: Update for platform-provided full disk encryption

- No change to ST; affects only evaluation activities.

TD0780: FIA_X509_EXT.1 Test 4 Clarification

- No change to ST; affects only evaluation activities.

- The following NIAP Technical Decisions affecting [TLS_PKG] apply to the TOE and have been accounted for in the ST development and the conduct of the evaluation, or were considered to be non-applicable for the reasons stated:

TD0442: Updated TLS Ciphersuites for TLS Package

- No change to ST; affects selections in FCS_TLSS_EXT.1 that are not applicable to the TOE.

TD0469: Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1

- No change to ST; the TD modifies evaluation activities only.

TD0499: Testing with pinned certificates

- No change to ST; the TD modifies evaluation activities only.

TD0513: CA Certificate loading

- No change to ST; the TD modifies evaluation activities only.

TD0588: Session Resumption Support in TLS package

- The ST accounts for this TD.

TD0726: Corrections to (D)TLSS SFRs in TLS 1.1 FP

- The ST accounts for this TD.

TD0739: PKG_TLS_V1.1 has 2 different publication dates

- No change to ST; the TD modifies evaluation activities only.

TD0770: TLSS.2 connection with no client cert

- The ST accounts for this TD.

- Common Criteria for Information Technology Security Evaluation Part 2: Security functional components, Version 3.1, Revision 5, April 2017.
  - Part 2 Extended
- Common Criteria for Information Technology Security Evaluation Part 3: Security assurance components, Version 3.1 Revision 5, April 2017.
  - Part 3 Extended

1.3 Conventions

The following conventions have been applied in this document:

- Security Functional Requirements – Part 2 of the CC defines the approved set of operations that may be applied to functional requirements: iteration, assignment, selection, and refinement.
  - Iteration: allows a component to be used more than once with varying operations. This ST includes iterated requirements reproduced from [APP_PP], which uses descriptive strings to distinguish iterations of a requirement. For example, [APP_PP] identifies iterations of FCS_COP.1 as follows: FCS_COP.1/SKC, FCS_COP.1/Hash, FCS_COP.1/Sig, and FCS_COP.1/KeyedHash. ST-defined iterations (i.e., those not reproduced from [APP_PP]) use digits inside parentheses (e.g., '(1)') to distinguish between iterations of an SFR (e.g., FDP_DAR_EXT.1(1) and FDP_DAR_EXT.1(2)).
  - Assignment: allows the specification of an identified parameter. Assignments are indicated using italics and are surrounded by brackets (e.g., [*assignment item*]). Note that an assignment within a selection would be identified in both italics and underline, with the brackets themselves underlined since they are explicitly part of the selection text, unlike the brackets around the selection itself (e.g., [selection item, [*assignment item inside selection*]]).
  - Selection: allows the specification of one or more elements from a list. Selections are indicated using underlines and are surrounded by brackets (e.g., [selection item]).
  - Refinement: allows technical changes to a requirement to make it more restrictive and allows non-technical changes to grammar and formatting. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… ~~some~~ **big** things …"). Note that minor grammatical changes that do not involve the addition or removal of entire

words (e.g., for consistency of quantity such as changing “meets” to “meet”) do not have formatting applied.

- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.
- The ST does not show operations that have been completed by the PP authors, though it does preserve brackets to show where such operations have been made.

1.3.1 Terminology

The following terms and abbreviations are used in this ST:

Table 1: Terms and Definitions

Term	Definition
Log Correlation Engine	An environmental component that is responsible for collecting log data from a variety of sources and aggregating it into a single collection of results.
Nessus Agent	An environmental component that is installed on an endpoint system to collect details about that system’s configuration and behavior.
Nessus Network Monitor	An environmental component that collects and analyzes raw network traffic.
Nessus	An environmental component that conducts remote scans of systems to collect data about their configuration and behavior and is used to deploy and collect data from remote Nessus Agent instances.
Platform	A general-purpose computer on which the TOE is installed.
Scan	The process by which Nessus or Nessus Agent actively collects data from a target system.
Security Center	The TOE; a software application that functions as a centralized aggregator for data collected by Nessus, Nessus Network Monitor, and Log Correlation Engine.

1.3.2 Abbreviations and Acronyms

Table 2: Abbreviations and Acronyms

Term	Definition
API	Application Programming Interface
AES	Advanced Encryption Standard
ASLR	Address Space Layout Randomization
CA	Certificate Authority
CAVP	Cryptographic Algorithm Validation Program
CBC	Cipher Block Chaining
CC	Common Criteria for Information Technology Security Evaluation
CCECG	Common Criteria Evaluated Configuration Guidance
CEM	Common Evaluation Methodology for Information Technology Security
CN	Common Name
CTR	Counter (cryptographic mode)

Term	Definition
CVE	Common Vulnerabilities and Exposures
DRBG	Deterministic Random Bit Generator
EAR	Entropy Analysis Report
ECC	Elliptic Curve Cryptography
ECDHE	Elliptic Curve Diffie-Hellman (Ephemeral)
ECDSA	Elliptic Curve Digital Signature Algorithm
FIPS	Federal Information Processing Standard
FQDN	Fully Qualified Domain Name
GB	Gigabyte
GCM	Galois/Counter Mode
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
LCE	Log Correlation Engine
NIAP	National Information Assurance Partnership
NIST	National Institute of Standards and Technology
NNM	Nessus Network Monitor
OCSP	Online Certificate Status Protocol
OE	Operational Environment
OID	Original Issue Document
OS	Operating System
PBKDF	Password-Based Key Derivation Function
PII	Personally Identifiable Information
PKI	Public Key Infrastructure
PP	Protection Profile
RAM	Random Access Memory
RPC	Remote Procedure Call
RSA	Rivest, Shamir and Adleman (algorithm for public-key cryptography)
SAN	Subject Alternative Name
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SSH	Secure Shell
SSL	Secure Sockets Layer
ST	Security Target
TCP	Transmission Control Protocol
TLS	Transport Layer Security
TOE	Target of Evaluation

Term	Definition
TSF	TOE Security Function
XML	Extensible Markup Language

2 Product and TOE Description

2.1 Introduction

Security Center (formerly Tenable.sc) is a software product that is designed to consolidate asset discovery, network monitoring, log aggregation, and vulnerability scanning into a single location to assess an organization's security posture comprehensively using a variety of different data.

Security Center is a server application that connects with one or more instances of other Tenable products (Nessus, Nessus Network Monitor, Log Correlation Engine) over secure channels.

The TOE conforms to [APP_PP] and [TLS_PKG]. As such, the security-relevant functionality of the product is limited to the claimed requirements in those standards. The security-relevant functionality is described in sections 2.3 and 2.4. The product overview in section 2.2 below is intended to provide the reader with an overall summary of the entire product so that its intended usage is clear. The subset of the product functionality that is within the evaluation scope is subsequently described in the sections that follow it.

2.2 Product Overview

Security Center is a vulnerability management product that is designed to provide visibility into network and system assets. The product is used to discover and scan assets such as servers, endpoints, network devices, operating systems, databases, and applications. Information collected by Security Center through its various connections to its operational environment and aggregated into the product can be presented in various customizable dashboards and analyzed by the product to determine the risk levels of findings or non-compliance with organizational security policies.

The product integrates three separate capabilities:

- **Asset monitoring:** Security Center interfaces with Nessus along with locally-deployed Nessus Agents to collect the results of authenticated configuration and vulnerability scanning. It also interfaces with Nessus to collect the results of remote authenticated scanning.
- **Network monitoring:** Security Center interfaces with Nessus Network Monitor (NNM), which passively scans network traffic using deep packet inspection to perform asset discovery and to detect user and application activities that could indicate compromise or misuse. Security Center collects this data from NNM.
- **Log aggregation:** Security Center interfaces with Log Correlation Engine (LCE) to aggregate, normalize, and analyze event log data from various sources. This activity can be used to establish baseline behavior for network assets to detect abnormal usage that may be indicative of vulnerability exploitation or compliance violations.

Security Center functions as a central point where all of the collected data is aggregated, analyzed, and displayed to administrators in various customizable views. It can also be used to orchestrate the data collection performed by the various environmental components via customizable scheduling. The aggregated data can be used to provide information on vulnerabilities, misconfiguration, and malware. Security Center also provides configurable workflows and alerts to automatically take corrective action based on specific findings.

Security Center also supports plugins, which can be downloaded and added to the product or made available to the other Tenable applications that connect to the product to detect specific vulnerabilities.

2.3 TOE Overview

The Target of Evaluation (TOE) for Security Center consists of the mandatory functionality prescribed by [APP_PP] and [TLS_PKG], as well as some selection-based functionality where needed.

The logical boundary is summarized in section 2.4.2 below. In general, the following Security Center capabilities are considered to be within the scope of the TOE:

- **Protection of sensitive data at rest:** the TOE uses encryption to protect credentials and other sensitive data.
- **Protection of data in transit:** the TOE secures data in transit between itself and its operational environment using TLS and HTTPS. Note that the TOE also has one logical interface that uses SSH but it relies on the underlying OS platform to provide this.
- **Trusted updates:** the TOE provides visibility into its current running version and the vendor distributes updates to it that are digitally signed so that administrators can securely maintain up-to-date software.
- **Remote administration:** the TOE provides a web-based graphical user interface (GUI) to administer its security functions. Note however that the bulk of the product's administration functions are outside the scope of the App PP and TLS Package and are therefore not part of the TOE.
- **Cryptographic services:** the TOE includes an implementation of OpenSSL with NIST-validated algorithm services that it uses to secure data at rest and in transit.
- **Secure interaction with operating system:** the TOE is designed to interact with its underlying host operating system platform in such a way that the TOE cannot be used as an attack vector to compromise an operating system.

The TOE's scanning, data collection, vulnerability analysis, analytics, and incident response capabilities are outside the scope of the TOE, as is any other product behavior that is not described in [APP_PP] or [TLS_PKG]. The content and execution of plugins is similarly excluded from the TOE, although they are discussed in the context of network communications because the TSF must use platform network resources to acquire them and make them available to environmental applications.

2.4 TOE Architecture

The TOE consists of the Security Center application, which is a C application with a PHP web front-end running on Apache. The TOE is a Linux application.

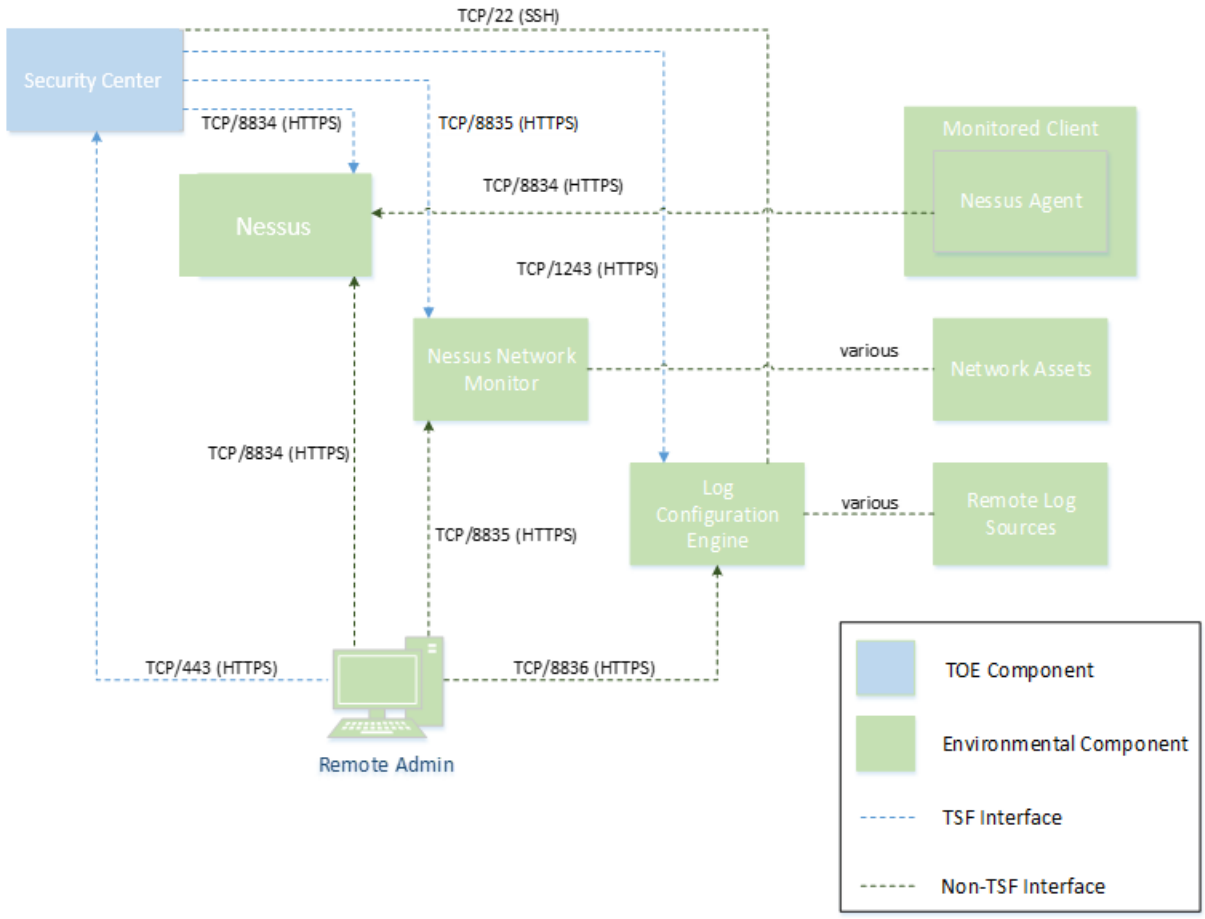
2.4.1 Physical Boundary

The TOE consists of the following component, as shown in Figure 1 below:

- Security Center 6.2.0

Figure 1 shows the TOE in a sample deployment with other Tenable applications in its operational environment.

Figure 1 - TOE Boundary



TSF-relevant remote interfaces are shown in Figure 1. Note that the TOE consists of exactly one instance of Security Center.

The TOE has the following system requirements for its host platform:

- 4 x 2GHz cores
- 8 GB RAM
- 125 GB disk storage—Tenable recommends installing the TOE on direct-attached storage (DAS) devices (or storage area networks (SANs), if necessary) with a storage latency of 10 ms or less. Tenable does not support installing the TOE on network-attached storage (NAS).
- Gigabit Ethernet.

These system requirements reflect the lightest usage scenarios for the TOE. Additional factors such as network size and storage retention requirements will affect the system requirements for a particular deployment. Refer to the relevant TOE documentation (as referenced in section 2.5) for the specific system requirements that apply to a given deployment.

The following network ports must be open for the TOE to function:

- TCP/22 (for communications with LCE)

- TCP/443 (for administrator communications)
- TCP/1243 (for communications with LCE)

Additional network ports must be open, but these are configurable if the default ports cannot be used. The connections and their default ports are as follows:

- TCP/8834 (for communications with Nessus)
- TCP/8835 (for communications with NNM).

The TOE's operational environment includes the following:

- Other Tenable components (one or more instances of Nessus, Nessus Agent, NNM, and LCE applications).
- Platform (hardware and software) on which the TOE is hosted.
    - The TOE is capable of running on a general-purpose Linux operating system on standard consumer-grade hardware on either a physical or virtual machine. For the evaluated configuration, the TOE was tested on a virtualized instance of RHEL 8.7 running on VMware ESXi 6.5 on a system using an AMD Ryzen Threadripper 1950X processor with the Zen microarchitecture.
- Full disk encryption is required for the TOE platform to ensure adequate data-at-rest protection.
- The platform on which the TOE is deployed is required to provide SSH client functionality through its host operating system.
- Web browser, used to access the web-based GUI.

2.4.2 Logical Boundary

This section summarizes the security functions provided by the TOE:

- Timely Security Updates
- Cryptographic Support
- User Data Protection
- Identification and Authentication
- Security Management
- Privacy
- Protection of the TSF
- Trusted Path/Channels

2.4.2.1 Timely Security Updates

The TOE developer has internal mechanisms for receiving reports of security flaws, tracking product vulnerabilities, and distributing software updates to customers in a timely manner.

2.4.2.2 Cryptographic Support

The TOE implements cryptography to protect data at rest and in transit.

For data at rest, the TOE stores credential data (both to log in to the TOE and to log in to remote systems for the purpose of conducting authenticated configuration scanning) as well as passphrase data used to protect PKI certificates that the TOE uses to authenticate to environmental components. This stored data is encrypted using AES or a PBKDF, depending on the data that is being stored.

For data in transit, the TOE implements TLS/HTTPS as both a client and a server. The TOE implements a TLS server for its administrative interface while it implements a TLS client to communicate with environmental components, including other Tenable products. The TOE supports mutual authentication as a TLS client.

The TOE implements all cryptography used for these functions using its own implementations of OpenSSL with NIST-approved algorithms. The TOE's DRBG is seeded using entropy from the underlying OS platform.

Some product functionality requires the use of SSH; the TOE does not claim SSH functionality as it invokes its platform to implement this.

2.4.2.3 User Data Protection

The TOE uses cryptographic mechanisms to protect sensitive data at rest. Credential data is protected through the use of a PBKDF while all other sensitive data is protected by the TOE platform's use of full disk encryption.

The TOE relies on the network connectivity and system log capabilities of its host OS platform. The TOE supports user-initiated and application-initiated uses of the network.

2.4.2.4 Identification and Authentication

The TOE supports X.509 certificate validation as part of establishing TLS and HTTPS connections. The TOE supports various certificate validity checking methods and can also check certificate revocation status using OCSP. If the validity status of a certificate cannot be determined, the certificate will be accepted. All other cases where a certificate is found to be invalid will result in rejection without an administrative override.

2.4.2.5 Security Management

The TOE itself and the configuration settings it uses are stored in locations recommended by the platform vendor.

The TOE provides a web-based GUI to administer its security functions. The GUI enforces username/password authentication using locally-stored credentials that are created using the TOE. The TOE does not include a default user account to access its management interface.

The security-relevant management functions supported by the TOE relate to the configuration of how frequently the various environmental components access network resources and for the transmission and presentation of system, network, and log data that the TOE obtains from its operational environment.

2.4.2.6 Privacy

The TOE does not handle personally identifiable information (PII) of any individuals.

2.4.2.7 Protection of the TSF

The TOE enforces various mechanisms to prevent itself from being used as an attack vector to its host OS platform. The TOE implements address space layout randomization (ASLR), does not allocate any memory with both write and execute permissions, does not write user-modifiable files to directories that contain executable files, is compiled using stack overflow protection, and is compatible with the security features of its host OS platform.

The TOE contains libraries and invokes system APIs that are well-known and explicitly identified.

The TOE has a mechanism to determine its current software version. Software updates to the TOE can be acquired by leveraging its OS platform. All updates are digitally signed to guarantee their authenticity and integrity.

2.4.2.8 Trusted Path/Channels

The TOE encrypts sensitive data in transit between itself and its operational environment using TLS and HTTPS. It facilitates the transmission of sensitive data from remote users over TLS and HTTPS.

The TOE may also invoke OS platform functionality to establish SSH communications with an instance of LCE in its operational environment.

2.5 TOE Documentation

Tenable provides the following product documentation in support of the installation and secure use of the TOE:

- Security Center 6.2.0 Common Criteria Evaluated Configuration Guide, 4 September 2023.

3 Security Problem Definition

This ST includes by reference the Security Problem Definition, composed of threats and assumptions, from [APP_PP]. The Common Criteria also provides for organizational security policies to be part of a security problem definition, but no such policies are defined in [APP_PP].

As a functional package, [TLS_PKG] does not contain a Security Problem Definition. The TOE's use of TLS is intended to mitigate the T.NETWORK_ATTACK and T.NETWORK_EAVESDROP threats defined by [APP_PP].

In general, the threat model of [APP_PP] is designed to protect against the following:

- Disclosure of sensitive data at rest or in transit that the user has a reasonable expectation of security for.
- Excessive or poorly-implemented interfaces with the underlying platform that allow an application to be used as an intrusion point to a system.

This threat model is applicable to the TOE because aggregated and analyzed vulnerability scan results could show an attacker what system weaknesses are present in the environment if they were able to obtain this data. It is also applicable because the TOE is a collection of executable binaries that an attacker could attempt to use to compromise the underlying OS platform if it was designed in such a manner that this exploitation was possible.

4 Security Objectives

As with the Security Problem Definition, this ST includes by reference the security objectives defined in [APP_PP]. This includes security objectives for the TOE (used to mitigate threats) and for its operational environment (used to satisfy assumptions).

As a functional package, [TLS_PKG] does not contain a Security Problem Definition. The TOE's use of TLS is intended to satisfy the O.PROTECTED_COMMS objective of [APP_PP] by implementing a specific method by which network communications are protected.

5 IT Security Requirements

This section defines the Security Functional Requirements (SFRs) and Security Assurance Requirements (SARs) that serve to represent the security functional claims for the Target of Evaluation (TOE) and to scope the evaluation effort.

The SFRs have all been drawn from the following Protection Profiles (PP) and Functional Packages:

- *Protection Profile for Application Software*, Version 1.4, 7 October 2021
- *Functional Package for Transport Layer Security (TLS),* Version 1.1, 1 March 2019.

As a result, any selection, assignment, or refinement operations already performed on the claimed SFRs drawn from these documents are not identified here (i.e., they are not formatted in accordance with the conventions specified in section 1.3 of this ST). Formatting conventions are only applied on SFR text that was chosen at the ST author's discretion.

5.1 Extended Requirements

All of the extended requirements in this ST have been drawn from [APP_PP] and [TLS_PKG]. These documents define the following extended SAR and extended SFRs; since they extended requirements have not been redefined in this ST, [APP_PP] and [TLS_PKG] should be consulted for more information regarding these extensions to CC Parts 2 and 3.

Defined in [APP_PP]:

- ALC_TSU_EXT.1 Timely Security Updates
- FCS_CKM_EXT.1 Cryptographic Key Generation Services
- FCS_CKM_EXT.1/PBKDF Password Conditioning
- FCS_HTTPS_EXT.1/Client HTTPS Protocol
- FCS_HTTPS_EXT.1/Server HTTPS Protocol
- FCS_HTTPS_EXT.2 HTTPS Protocol with Mutual Authentication
- FCS_RBG_EXT.1 Random Bit Generation Services
- FCS_RBG_EXT.2 Random Bit Generation from Application
- FCS_STO_EXT.1 Storage of Credentials
- FDP_DAR_EXT.1 Encryption of Sensitive Application Data
- FDP_DEC_EXT.1 Access to Platform Resources
- FDP_NET_EXT.1 Network Communications
- FIA_X509_EXT.1 X.509 Certificate Validation
- FIA_X509_EXT.2 X.509 Certificate Authentication
- FMT_CFG_EXT.1 Secure by Default Configuration
- FMT_MEC_EXT.1 Supported Configuration Mechanism
- FPR_ANO_EXT.1 User Consent for Transmission of Personally Identifiable Information
- FPT_AEX_EXT.1 Anti-Exploitation Capabilities
- FPT_API_EXT.1 Use of Supported Services and APIs
- FPT_IDV_EXT.1 Software Identification and Versions
- FPT_LIB_EXT.1 Use of Third Party Libraries
- FPT_TUD_EXT.1 Integrity for Installation and Update
- FPT_TUD_EXT.2 Integrity for Installation and Update
- FTP_DIT_EXT.1 Protection of Data in Transit.

Defined in [TLS_PKG]:

- FCS_TLS_EXT.1 TLS Protocol
- FCS_TLSC_EXT.1 TLS Client Protocol
- FCS_TLSC_EXT.2 TLS Client Support for Mutual Authentication
- FCS_TLSC_EXT.5 TLS Client Support for Supported Groups Extension
- FCS_TLSS_EXT.1 TLS Server Protocol
- FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication

5.2 TOE Security Functional Requirements

The following table identifies the SFRs that are satisfied by the TOE.

Table 3: TOE Security Functional Components

Requirement Class	Requirement Component
FCS: Cryptographic Support	FCS_CKM_EXT.1 – Cryptographic Key Generation Services
	FCS_CKM.1/AK – Cryptographic Asymmetric Key Generation
	FCS_CKM_EXT.1/PBKDF – Password Conditioning
	FCS_CKM.2 – Cryptographic Key Establishment
	FCS_COP.1/SKC – Cryptographic Operation – Encryption/Decryption
	FCS_COP.1/Hash – Cryptographic Operation – Hashing
	FCS_COP.1/Sig – Cryptographic Operation – Signing
	FCS_COP.1/KeyedHash – Cryptographic Operation – Keyed-Hash Message Authentication
	FCS_HTTPS_EXT.1/Client – HTTPS Protocol
	FCS_HTTPS_EXT.1/Server – HTTPS Protocol
	FCS_HTTPS_EXT.2 – HTTPS Protocol with Mutual Authentication
	FCS_RBG_EXT.1 – Random Bit Generation Services
	FCS_RBG_EXT.2 – Random Bit Generation from Application
	FCS_STO_EXT.1 – Storage of Credentials
	FCS_TLS_EXT.1 – TLS Protocol (TLS Package)
	FCS_TLSC_EXT.1 – TLS Client Protocol (TLS Package)
	FCS_TLSC_EXT.2 – TLS Client Support for Mutual Authentication (TLS Package)
	FCS_TLSC_EXT.5 – TLS Client Support for Supported Groups Extension (TLS Package)
	FCS_TLSS_EXT.1 – TLS Server Protocol (TLS Package)
	FCS_TLSS_EXT.2 – TLS Server Support for Mutual Authentication (TLS Package)
FDP: User Data Protection	FDP_DAR_EXT.1(1) – Encryption of Sensitive Application Data (by TOE)
	FDP_DAR_EXT.1(2) – Encryption of Sensitive Application Data (by OE)
	FDP_DEC_EXT.1 – Access to Platform Resources
	FDP_NET_EXT.1 – Network Communications

Requirement Class	Requirement Component
FIA: Identification and authentication	FIA_X509_EXT.1 – X.509 Certificate Validation
	FIA_X509_EXT.2 – X.509 Certificate Authentication
FMT: Security Management	FMT_CFG_EXT.1 – Secure by Default Configuration
	FMT_MEC_EXT.1 – Supported Configuration Mechanism
	FMT_SMF.1 – Specification of Management Functions
FPR: Privacy	FPR_ANO_EXT.1 – User Consent for Transmission of Personally Identifiable Information
FPT: Protection of the TSF	FPT_AEX_EXT.1 – Anti-Exploitation Capabilities
	FPT_API_EXT.1 – Use of Supported Services and APIs
	FPT_IDV_EXT.1 – Software Identification and Versions
	FPT_LIB_EXT.1 – Use of Third Party Libraries
	FPT_TUD_EXT.1 – Integrity for Installation and Update
	FPT_TUD_EXT.2 – Integrity for Installation and Update
FTP: Trusted Path/Channels	FTP_DIT_EXT.1 – Protection of Data in Transit

5.2.1 Cryptographic Support (FCS)

5.2.1.1 FCS_CKM_EXT.1 – Cryptographic Key Generation Services

Note: This SFR has been modified in accordance with TD0717.

FCS_CKM_EXT.1.1 The application shall [

- implement asymmetric key generation

].

Application Note: *The TOE also relies on the underlying OS platform to implement SSH functionality. However, this capability, including generation of asymmetric cryptographic keys for use within SSH, is implemented entirely outside the TOE boundary. As such, the ST does not select "invoke platform-provided functionality for asymmetric key generation".*

5.2.1.2 FCS_CKM.1/AK – Cryptographic Asymmetric Key Generation

Note: This SFR has been modified in accordance with TD0717.

FCS_CKM.1.1/AK The application shall [

- implement functionality

] to generate asymmetric cryptographic keys in accordance with a specified cryptographic key generation algorithm [

- [ECC schemes] using ["NIST curves" P-384 and [P-256]] that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4]

].

5.2.1.3 FCS_CKM_EXT.1/PBKDF – Password Conditioning

Note: This SFR has been modified in accordance with TD0717.

FCS_CKM_EXT.1.1/PBKDF A password/passphrase shall perform [Password-based Key Derivation Functions] in accordance with a specified cryptographic algorithm as specified in FCS_COP.1/KeyedHash, with [*10000*] iterations, and output cryptographic key sizes [128] that meet the following [NIST SP 800-132].

FCS_CKM_EXT.1.2/PBKDF The TSF shall generate salts using a RBG that meets FCS_RBG_EXT.1 and with entropy corresponding to the security strength selected for PBKDF in FCS_CKM.1.1/PBKDF.

5.2.1.4 FCS_CKM.2 – Cryptographic Key Establishment

FCS_CKM.2.1 The application shall [implement functionality] to perform cryptographic key establishment in accordance with a specified cryptographic key establishment method: [

- [Elliptic curve-based key establishment schemes] that meets the following: [NIST Special Publication 800-56A, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"]

].

5.2.1.5 FCS_COP.1/SKC – Cryptographic Operation – Encryption/Decryption

Note: This SFR has been modified in accordance with TD0717.

FCS_COP.1.1/SKC The application shall perform [encryption/decryption] in accordance with a specified cryptographic algorithm [

- AES-CBC (as defined in NIST SP 800-38A) mode,
- AES-GCM (as defined in NIST SP 800-38D) mode

] and cryptographic key sizes [128-bit, 256-bit].

5.2.1.6 FCS_COP.1/Hash – Cryptographic Operation – Hashing

Note: This SFR has been modified in accordance with TD0717.

FCS_COP.1.1/Hash The application shall perform [cryptographic hashing services] in accordance with a specified cryptographic algorithm [

- SHA-256,
- SHA-384,
- SHA-512

] and message digest sizes [

- 256,
- 384,
- 512

] bits that meet the following: [FIPS Pub 180-4].

5.2.1.7 FCS_COP.1/Sig – Cryptographic Operation – Signing

Note: This SFR has been modified in accordance with TD0717.

FCS_COP.1.1/Sig The application shall perform [cryptographic signature services (generation and verification)] in accordance with a specified cryptographic algorithm [

- RSA schemes using cryptographic key sizes of [2048-bit or greater] that meet the following: [FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5]

].

5.2.1.8 FCS_COP.1/KeyedHash – Cryptographic Operation – Keyed-Hash Message Authentication

Note: This SFR has been modified in accordance with TD0717.

FCS_COP.1.1/KeyedHash The application shall perform [keyed-hash message authentication] in accordance with a specified cryptographic algorithm [

- HMAC-SHA-256,
- HMAC-SHA-384,
- HMAC-SHA-512

] and [

- no other algorithms

] with key sizes [*256 bits, 384 bits, 512 bits*] and message digest sizes [256, 384, 512] and [no other size] bits that meet the following: [FIPS Pub 198-1, '*The Keyed-Hash Message Authentication Code*' and FIPS Pub 180-4 '*Secure Hash Standard*'].

5.2.1.9 FCS_HTTPS_EXT.1/Client – HTTPS Protocol

FCS_HTTPS_EXT.1.1/Client The application shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2/Client The application shall implement HTTPS using TLS as defined in the Functional Package for TLS.

FCS_HTTPS_EXT.1.3/Client The application shall [not establish the application-initiated connection] if the peer certificate is deemed invalid.

5.2.1.10 FCS_HTTPS_EXT.1/Server – HTTPS Protocol

Note: This SFR has been modified in accordance with TD0736.

FCS_HTTPS_EXT.1.1/Server The application shall implement the HTTPS protocol that complies with RFC 2818.

FCS_HTTPS_EXT.1.2/Server The application shall implement HTTPS using TLS as defined in the TLS package.

FCS_HTTPS_EXT.1.3/Server The application shall [not establish the connection] if the peer certificate is deemed invalid.

5.2.1.11 FCS_HTTPS_EXT.2 – HTTPS Protocol with Mutual Authentication

FCS_HTTPS_EXT.2.1 The application shall [not establish the connection] if the peer certificate is deemed invalid.

5.2.1.12 FCS_RBG_EXT.1 – Random Bit Generation Services

FCS_RBG_EXT.1.1 The application shall [

- implement DRBG functionality

] for its cryptographic operations.

5.2.1.13 FCS_RBG_EXT.2 – Random Bit Generation from Application

FCS_RBG_EXT.2.1 The application shall perform all deterministic random bit generation (DRBG) services in accordance with NIST Special Publication 800-90A using [CTR_DRBG (AES)].

FCS_RBG_EXT.2.2 The deterministic RBG shall be seeded by an entropy source that accumulates entropy from a platform-based DRBG and [

- no other noise source

] with a minimum of [

- 256 bits

] of entropy at least equal to the greatest security strength (according to NIST SP 800-57) of the keys and hashes that it will generate.

5.2.1.14 FCS_STO_EXT.1 – Storage of Credentials

FCS_STO_EXT.1.1 The application shall [

- implement functionality to securely store [*Web GUI authentication credentials, authenticated scanning credentials, PKI certificate passphrases*] according to [FCS_COP.1/SKC, FCS_CKM_EXT.1/PBKDF]

] to non-volatile memory.

5.2.1.15 FCS_TLS_EXT.1 – TLS Protocol (TLS Package)

FCS_TLS_EXT.1.1 The product shall implement [

- TLS as a client,
- TLS as a server

].

5.2.1.16 FCS_TLSC_EXT.1 – TLS Client Protocol (TLS Package)

FCS_TLSC_EXT.1.1[1] The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a client that supports the cipher suites [

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289]

and also supports functionality for [

- mutual authentication

].

FCS_TLSC_EXT.1.2 The product shall verify that the presented identifier matches the reference identifier according to RFC 6125.

FCS_TLSC_EXT.1.3 The product shall not establish a trusted channel if the server certificate is invalid [

- with no exceptions

].

5.2.1.17 FCS_TLSC_EXT.2 – TLS Client Support for Mutual Authentication (TLS Package)

FCS_TLSC_EXT.2.1 The product shall support mutual authentication using X.509v3 certificates.

[1] This SFR is modified by TD0442 but this ST does not claim any of the selections that were added by the TD.

5.2.1.18 FCS_TLSC_EXT.5 – TLS Client Support for Supported Groups Extension (TLS Package)

FCS_TLSC_EXT.5.1 The product shall present the Supported Groups Extension in the Client Hello with the supported groups [
- secp256r1,
- secp384r1

].

5.2.1.19 FCS_TLSS_EXT.1 – TLS Server Protocol (TLS Package)

FCS_TLSS_EXT.1.1[2] The product shall implement TLS 1.2 (RFC 5246) and [no earlier TLS versions] as a server that supports the cipher suites [
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 as defined in RFC 5289,
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 as defined in RFC 5289

] and also supports functionality for [
- mutual authentication,
- session resumption based on session IDs according to RFC 4346 (TLS1.1) or RFC 5246 (TLS1.2)

].

FCS_TLSS_EXT.1.2 The product 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 product shall perform key establishment for TLS using [
- ECDHE parameters using elliptic curves [secp256r1, secp384r1] and no other curves

].

5.2.1.20 FCS_TLSS_EXT.2 – TLS Server Support for Mutual Authentication (TLS Package)

FCS_TLSS_EXT.2.1 The product shall support authentication of TLS clients using X.509v3 certificates.

FCS_TLSS_EXT.2.2[3] The product shall [not establish a trusted channel] if the client certificate is invalid.

FCS_TLSS_EXT.2.3 The product shall not establish a trusted channel if the Distinguished Name (DN) or Subject Alternative Name (SAN) contained in a certificate does not match one of the expected identifiers for the client.

[2] This SFR is modified by the following Technical Decisions: TD0442 (although this ST does not claim any of the selections that were added by the TD); TD0588; TD0726.

[3] Modified in accordance with TD0770.

5.2.2 User Data Protection (FDP)

5.2.2.1 FDP_DAR_EXT.1(1) – Encryption of Sensitive Application Data (by TOE)

FDP_DAR_EXT.1.1(1) The application shall [
- protect sensitive data in accordance with FCS_STO_EXT.1

] in non-volatile memory.

Application Note: *"Sensitive data" includes both the credential data specified in FCS_STO_EXT.1 as well as system scan, network traffic, and log data that is collected from the Operational Environment. This data is not credential data, but it is still protected using the methods specified in FCS_STO_EXT.1. This is because all sensitive data, regardless of whether or not it is credential data, is stored in an encrypted database.*

5.2.2.2 FDP_DAR_EXT.1(2) – Encryption of Sensitive Application Data (by OE)

FDP_DAR_EXT.1.1(2) The application shall [
- leverage platform-provided functionality to encrypt sensitive data

] in non-volatile memory.

Application Note: *The database encryption referenced in FDP_DAR_EXT.1(1) requires a secret key to be stored on the platform. This is considered to be sensitive data and is therefore protected using platform-provided means.*

5.2.2.3 FDP_DEC_EXT.1 – Access to Platform Resources

FDP_DEC_EXT.1.1 The application shall restrict its access to [
- network connectivity

].

FDP_DEC_EXT.1.2 The application shall restrict its access to [
- [*system configuration*]

].

5.2.2.4 FDP_NET_EXT.1 – Network Communications

FDP_NET_EXT.1.1 The application shall restrict network communication to [
- User-initiated communication for [
  - *manual initiation of plugin download,*
  - *access to Web GUI,*
  - *retrieval of NNM data for analysis,*
  - *initiation of remote Nessus scan,*
  - *check for and download of plugin updates*]
- [*application-initiated network communication for*
  - *periodic initiation of plugin download,*
  - *retrieval of log data from LCE,*
  - *import of LCE records into vulnerability database,*
  - *retrieval of data from Nessus engine,*
  - *periodic retrieval of data from NNM,*
  - *periodic initiation of remote scan,*
  - *push plugin updates to Nessus and NNM,*

- *check for and download of plugin updates*]

].

5.2.3 Identification and Authentication (FIA)

5.2.3.1 FIA_X509_EXT.1 – X.509 Certificate Validation

FIA_X509_EXT.1.1 The application shall [implement functionality] to validate certificates in accordance with the following rules:

- RFC 5280 certificate validation and certificate path validation.
- The certificate path must terminate with a trusted CA certificate.
- The application shall validate a certificate path by ensuring the presence of the basicConstraints extension and that the CA flag is set to TRUE for all CA certificates, and that any path constraints are met.
- The application shall validate that any CA certificate includes caSigning purpose in the key usage field.
- The application shall validate the revocation status of the certificate using [OCSP as specified in RFC 6960].
- The application shall validate the extendedKeyUsage (EKU) 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 EKU 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 EKU field.
  - S/MIME certificates presented for email encryption and signature shall have the Email Protection purpose (id-kp 4 with OID 1.3.6.1.5.5.7.3.4) in the EKU 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 EKU field.
  - Server certificates presented for EST shall have the CMC Registration Authority (RA) purpose (id-kp-cmcRA with OID 1.3.6.1.5.5.7.3.28) in the EKU field.

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

5.2.3.2 FIA_X509_EXT.2 – X.509 Certificate Authentication

FIA_X509_EXT.2.1 The application shall use X.509v3 certificates as defined by RFC 5280 to support authentication for [HTTPS, TLS].

FIA_X509_EXT.2.2 When the application cannot establish a connection to determine the validity of a certificate, the application shall [not accept the certificate].

5.2.4 Security Management (FMT)

5.2.4.1 FMT_CFG_EXT.1 – Secure by Default Configuration

FMT_CFG_EXT.1.1 The application shall provide only enough functionality to set new credentials when configured with default credentials or no credentials.

FMT_CFG_EXT.1.2 The application shall be configured by default with file permissions which protect the application binaries and data files from modification by normal unprivileged users.

5.2.4.2 FMT_MEC_EXT.1 – Supported Configuration Mechanism

FMT_MEC_EXT.1.1 The application shall [invoke the mechanisms recommended by the platform vendor for storing and setting configuration options].

5.2.4.3 FMT_SMF.1 – Specification of Management Functions

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

- enable/disable the transmission of any information describing the system's hardware, software, or configuration,
- enable/disable transmission of any application state (e.g. crashdump) information,
- *[configuration of presentation (reporting/dashboards/analytics) of collected system and network data]*

].

5.2.5 Privacy (FPR)

5.2.5.1 FPR_ANO_EXT.1 – User Consent for Transmission of Personally Identifiable Information

FPR_ANO_EXT.1.1 The application shall [

- not transmit PII over a network

].

5.2.6 Protection of the TSF (FPT)

5.2.6.1 FPT_AEX_EXT.1 – Anti-Exploitation Capabilities

FPT_AEX_EXT.1.1 The application shall not request to map memory at an explicit address except for [*no exceptions*].

FPT_AEX_EXT.1.2 The application shall [

- not allocate any memory region with both write and execute permissions

].

FPT_AEX_EXT.1.3 The application shall be compatible with security features provided by the platform vendor.

FPT_AEX_EXT.1.4 The application shall not write user-modifiable files to directories that contain executable files unless explicitly directed by the user to do so.

FPT_AEX_EXT.1.5 The application shall be built with stack-based buffer overflow protection enabled.

5.2.6.2 FPT_API_EXT.1 – Use of Supported Services and APIs

FPT_API_EXT.1.1 The application shall use only documented platform APIs.

5.2.6.3 FPT_IDV_EXT.1 – Software Identification and Versions

FPT_IDV_EXT.1.1 The application shall be versioned with [[*semantic versioning (SemVer)*]].

5.2.6.4 FPT_LIB_EXT.1 – Use of Third Party Libraries

FPT_LIB_EXT.1.1 The application shall be packaged with only [*third-party libraries listed in Appendix A.2*].

Application Note: *The TOE uses a large number of third-party libraries so this information has been provided in an Appendix for readability purposes.*

5.2.6.5 FPT_TUD_EXT.1 – Integrity for Installation and Update

FPT_TUD_EXT.1.1 The application shall [provide the ability] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2 The application shall [provide the ability, leverage the platform] to query the current version of the application software.

FPT_TUD_EXT.1.3 The application shall not download, modify, replace, or update its own binary code.

FPT_TUD_EXT.1.4 Application updates shall be digitally signed such that the application platform can cryptographically verify them prior to installation.

FPT_TUD_EXT.1.5 The application is distributed [as an additional software package to the platform OS].

5.2.6.6 FPT_TUD_EXT.2 – Integrity for Installation and Update

FPT_TUD_EXT.2.1[4] The application shall be distributed using [the format of the platform-supported package manager].

FPT_TUD_EXT.2.2 The application shall be packaged such that its removal results in the deletion of all traces of the application, with the exception of configuration settings, output files, and audit/log events.

FPT_TUD_EXT.2.3 The application installation package shall be digitally signed such that its platform can cryptographically verify them prior to installation.

5.2.7 Trusted Path/Channels (FTP)

5.2.7.1 FTP_DIT_EXT.1 – Protection of Data in Transit

FTP_DIT_EXT.1.1[5] The application shall [

- encrypt all transmitted [sensitive data] with [
  - HTTPS as a client in accordance with FCS_HTTPS_EXT.1/Client for [*retrieving Nessus and Nessus Agent scan results from Nessus*],

[4] Modified in accordance with TD0628.

[5] Modified in accordance with TD0743.

- HTTPS as a server in accordance with FCS_HTTPS_EXT.1/Server for [*securing administrator interactions with the TOE*],
- HTTPS as a server using mutual authentication in accordance with FCS_HTTPS_EXT.2 for [*securing administrator interactions with the TOE*],
- TLS as a server as defined in the Functional Package for TLS and also supports functionality for [mutual authentication] for [*securing administrator interactions with the TOE*],
- TLS as a client as defined in the Functional Package for TLS] for [*collecting unaltered bulk log data aggregated by Log Correlation Engine, collecting network traffic data from Nessus Network Monitor*]

- invoke platform-provided functionality to encrypt all transmitted sensitive data with [SSH] for [*collecting parsed log data from Log Correlation Engine*]

] between itself and another trusted IT product.

5.3 TOE Security Assurance Requirements

The security assurance requirements for the TOE are included by reference to [APP_PP].

Table 4: Assurance Components

Requirement Class	Requirement Component
ADV: Development	ADV_FSP.1 Basic Functional Specification
AGD: Guidance Documentation	AGD_OPE.1 Operational User Guidance
	AGD_PRE.1 Preparative Procedures
ALC: Life-cycle Support	ALC_CMC.1 Labeling of the TOE
	ALC_CMS.1 TOE CM coverage
	ALC_TSU_EXT.1 Timely Security Updates
ATE: Tests	ATE_IND.1 Independent Testing – Conformance
AVA: Vulnerability Assessment	AVA_VAN.1 Vulnerability Survey

As a functional package, [TLS_PKG] does not define its own SARs. The expectation is that all SARs required by [APP_PP] will apply to the entire TOE, including the portions addressed by [TLS_PKG]. Consequently, the evaluation activities specified in [PP_PP] apply to the entire TOE evaluation, including any changes made to them by subsequent NIAP Technical Decisions as summarized in section 1.2 above.

The [TLS_PKG] does contain evaluation activities for how to evaluate its SFR claims as part of the evaluation of ASE_TSS.1, AGD_OPE.1, AGD_PRE.1, and ATE_IND.1. All Security Functional Requirements specified by [TLS_PKG] are evaluated in the manner specified in that package.

6 TOE Summary Specification

This chapter describes the security functions of the TOE:

- Timely Security Updates
- Cryptographic Support
- User Data Protection
- Identification and Authentication
- Security Management
- Privacy
- Protection of the TSF
- Trusted Path/Channels.

6.1 Timely Security Updates

Tenable supports a timely security update process for the TOE In addition to their own internal research, the product vendor supports disclosure of potential issues using community forums, direct engagement, and the Tenable support channel. For issues where there is a potential security concern, the support channel uses HTTPS for secure disclosure.

When an issue is reported, Tenable will determine its applicability to the product. The length of time needed to make this determination depends on the complexity of the issue and the extent to which it can be reproduced; well-documented issues such as exposure to a published CVE can be made quickly. If found to be a security issue, an update is released within 30 days. Tenable monitors the third-party components used by the TOE for potential security issues as well. However, an issue with a dependent component may not be addressed if found not to be applicable to the TOE. For example, security issues are frequently found within the PHP image library but Tenable does not install this library as part of the Security Center distribution.

Security updates to the TOE are delivered as regular update packages in the same manner as a functional update. This process is described in section 6.7 below.

6.2 Cryptographic Support

The TOE uses cryptography to secure data in transit between itself and its operational environment.

All TOE cryptographic services are implemented by the OpenSSL cryptographic library. The TOE uses OpenSSL 3.0.10. The cryptographic algorithms supplied by the TOE are NIST-validated. The following table identifies the cryptographic algorithms used by the TSF, the associated standards to which they conform, and the NIST certificates that demonstrate that the claimed conformance has been met.

Table 5: Cryptographic Functions

Functions	Standards	Certificates
FCS_CKM.1/AK Cryptographic Asymmetric Key Generation		
ECC key pair generation (NIST curves P-256, P-384)	FIPS PUB 186-4	A3617
FCS_CKM.2 Cryptographic Key Establishment		
ECDSA based key establishment	NIST SP 800-56A	A3617

Functions	Standards	Certificates
FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption		
AES-CBC and AES-GCM (128, 256 bits)	CBC as defined in NIST SP 800-38A GCM as defined in NIST SP 800-38D	A3617
FCS_COP.1/Hash Cryptographic Operation – Hashing		
SHA-256, SHA-384, and SHA-512 (digest sizes 256, 384, and 512 bits)	FIPS PUB 180-4	A3617
FCS_COP.1/Sig Cryptographic Operation – Signing		
RSA (2048-bit or greater)	FIPS PUB 186-4, Section 4	A3617
FCS_COP.1/KeyedHash Cryptographic Operation – Keyed Hash Message Authentication		
HMAC-SHA-256, HMAC-SHA-384, HMAC-SHA-512	FIPS PUB 198-1 FIPS PUB 180-4	A3617
FCS_RBG_EXT.2 Random Bit Generation from Application		
CTR_DRBG DRBG (256 bits)	NIST SP 800-90A NIST SP 800-57	A3617

The TOE generates asymmetric keys in support of trusted communications. The TSF generates ECC keys using P-256 and P-384. These keys are generated in support of the ECDHE key establishment schemes that are used for TLS/HTTPS communications. To ensure sufficient key strength, the TOE also implements DRBG functionality for key generation, using the AES-CTR_DRBG. The proprietary Entropy Analysis Report (EAR) describes how the TSF extracts random data from software-based sources to ensure that an amount of entropy that is at least equal to the strength of the generated keys is present (i.e., at least 256 bits when the largest supported keys are generated) when seeding the DRBG for key generation purposes. The TOE relies on the Linux OS platform entropy source. Specifically, random numbers are obtained from the `/dev/random` pseudo-device. The platform is assumed to provide at least 256 bits of entropy.

The TOE uses TLS 1.2 for client and server communications. In the case where the TOE acts as a TLS server, all other TLS versions are rejected. The TLS client and server implementation support the following TLS cipher suites in the TOE's evaluated configuration:

- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.

All supported ciphersuites use elliptic curves as the method of key establishment. The TSF presents secp256r1 and secp384r1 as the supported values in the Supported Groups extension and uses the same NIST curves for key establishment.

As part of certificate validation in the establishment of TLS connectivity, the TOE will validate the reference identifier of a presented server certificate. This is done through validation of the Common Name (CN) and Subject Alternative Name (SAN) certificate fields, the latter of which is expected to contain the FQDN of the external system that is presenting the certificate to the TOE. The reference identifier is established by configuration. IP addresses are not supported. Wildcards are only supported for the left-most label

immediately preceding the public suffix. Certificate pinning is not supported. All digital signatures used for the establishment of TLS communications use 2048-bit RSA.

The TOE uses TLS client functionality for communications between the TOE and other Tenable applications in the operational environment. All communications between the TOE and Nessus and between the TOE and NNM use mutually-authenticated TLS, while outbound communications to LCE and to the operational environment do not.

The TOE uses TLS server functionality for communications from remote administrators to its web-based GUI. The TOE implementation supports session resumption based on session IDs as specified in RFC 5246. Mutual authentication is supported for this interface. As with TLS server certificates, the CN and SAN of the client certificate are validated for this interface.

The TOE's implementation of HTTPS conforms to RFC 2818. Regardless of whether the TOE is acting as a client or a server for HTTPS, the connection will be rejected if certificate validation fails.

The TOE also uses OpenSSL to secure credential data at rest. Specifically, the TOE stores the following credentials:

- Web GUI authentication credentials: username and hashed password data for locally-defined users.
- Authenticated scanning credentials: operating system credential data that is stored by the TOE and used to perform authenticated scanning of remote systems.
- Passphrases for certificate encryption: used to encrypt PKI certificates that the TOE uses for communications with remote administrators and with the environmental Tenable applications when using TLS mutual authentication.

All credential data is encrypted by the TOE using AES-CBC, except for administrative credentials to the TOE, which are encrypted using PBKDF2. The TOE uses the DRBG specified in FCS_RBG_EXT.2 to generate salts that contain at least as many entropy bits as the output key length. The TOE's PBKDF2 implementation performs 10,000 iterations and outputs a 128-bit strength key. Password-based derived keys are formed using a 128-bit salt that is randomly generated by the TOE's DRBG. This is input to the PBKDF function along with the password and specified hashing algorithm, which is SHA-512.

The TOE does not maintain a key hierarchy; the TOE's usage of PBKDF is to generate a hash.

The Cryptographic Support security function is designed to satisfy the following security functional requirements:

- FCS_CKM_EXT.1, FCS_CKM.1/AK—the TOE implements its own cryptographic functionality for asymmetric key generation. The TOE uses a NIST-validated implementation to generate asymmetric keys in support of TLS communications.
- FCS_CKM_EXT.1/PBKDF—the TOE performs password-based key derivation in support of secure storage of credentials.
- FCS_CKM.2—the TOE performs NIST-validated key establishment in support of TLS communications.
- FCS_COP.1/SKC—the TOE uses a NIST-validated implementation to perform AES encryption and decryption in support of both TLS communications and secure storage of credentials.

- FCS_COP.1/Hash—the TOE uses a NIST-validated implementation to perform cryptographic hashing in support of TLS communications and password-based key derivation.
- FCS_COP.1/Sig—the TOE uses a NIST-validated implementation to generate and verify RSA digital signatures in support of TLS communications.
- FCS_COP.1/KeyedHash—the TOE uses a NIST-validated implementation to perform HMAC functions in support of TLS communications and the pseudo-random function used for password-based key derivation.
- FCS_HTTPS_EXT.1/Client—the TOE implements HTTPS as a client to secure data in transit.
- FCS_HTTPS_EXT.1/Server—the TOE implements HTTPS as a server to secure data in transit.
- FCS_HTTPS_EXT.2—the TOE implements mutual authentication when acting as an HTTPS server.
- FCS_RBG_EXT.1—the TOE implements its own random bit generation services.
- FCS_RBG_EXT.2—the TOE uses a NIST-validated implementation to generate pseudo-random bits and this implementation is seeded with sufficiently strong entropy collected from the operational environment.
- FCS_STO_EXT.1—the TOE uses its own cryptographic functions to secure credential data at rest.
- FCS_TLS_EXT.1—the TOE implements TLS to secure data in transit.
- FCS_TLSC_EXT.1—the TOE implements TLS as a client.
- FCS_TLSC_EXT.2—the TOE's TLS client implementation supports mutual authentication for some TLS functions.
- FCS_TLSC_EXT.5—the TOE's TLS client implementation presents supported elliptic curves to the server in the Supported Groups extension when an ECDHE cipher suites is negotiated.
- FCS_TLSS_EXT.1—the TOE implements TLS as a server.
- FCS_TLSS_EXT.2—the TOE's TLS server implementation supports mutual authentication for some TLS functions.

6.3 User Data Protection

The [APP_PP] defines 'sensitive data' as follows: "Sensitive data may include all user or enterprise data or may be specific application data such as emails, messaging, documents, calendar items, and contacts. Sensitive data must minimally include PII, credentials, and keys. Sensitive data shall be identified in the application's TSS by the ST author."

The table below lists the data that is considered to be 'sensitive data' for the TOE along with where that data resides and how it is transmitted to its operational environment.

Table 6: Sensitive Data

Sensitive Data	Exchange	Protection at Rest	Protection in Transit
GUI credentials	Admin's browser to TOE over browser connection	FCS_STO_EXT.1 (PBKDF)	HTTPS

Sensitive Data	Exchange	Protection at Rest	Protection in Transit
Remote system scan credentials	TOE to Nessus; Nessus to remote system	FCS_STO_EXT.1 (AES-CBC)	TLS; native protocol used by applicable OE authentication mechanism
Nessus authentication credentials	TOE to Nessus over XML RPC	FCS_STO_EXT.1 (AES-CBC)	HTTPS
NNM authentication credentials	TOE to NNM over TLS	FCS_STO_EXT.1 (AES-CBC)	TLS
LCE authentication credentials	TOE to LCE over TLS	FCS_STO_EXT.1 (AES-CBC)	TLS
Passphrase for PKI certificate encryption	None	FCS_STO_EXT.1 (PBKDF)	N/A
Collected system scan data	Nessus to TOE	FDP_DAR_EXT.1(2)	HTTPS
Collected network traffic data	NNM to TOE	FDP_DAR_EXT.1(2)	TLS
Collected log data	LCE to TOE	FDP_DAR_EXT.1(2)	TLS

All sensitive data that is not credential data is protected by the platform full disk encryption method specified in FDP_DAR_EXT.1(2).

Other than the hardware resources ordinarily used by applications (such as central processing units, memory, input/output peripherals, and persistent storage), the TOE accesses only network connectivity resources provided by its platform. The TOE uses network connectivity for remote management and connections to environmental components.

The TOE restricts its access to sensitive information repositories on the platform to system configuration, which it accesses in order to generate a diagnostic report of local system configuration for troubleshooting purposes.

The TOE uses environmental network capabilities in various ways. All communications between the TOE and environmental Tenable components are encrypted. The following table highlights the TOE's network usage.

Table 7: TOE Network Usage

Component	User-Initiated	Externally-Initiated	TOE-Initiated
Security Center	Manual initiation of plugin download	None	Periodic initiation of plugin download
	Access to web GUI		Retrieval of log data from LCE
	Retrieval of NNM data for analysis		Import of LCE records into vulnerability database
	Initiation of remote Nessus scan		Retrieval of data from Nessus engine
	Check for and download of plugin updates		Periodic retrieval of data from NNM

Component	User-Initiated	Externally-Initiated	TOE-Initiated
			Periodic initiation of remote Nessus scan
			Push plugin updates to Nessus and NNM
			Check for and download of plugin updates

The User Data Protection security function is designed to satisfy the following security functional requirements:

- FDP_DAR_EXT.1(1)—sensitive credential data at rest is protected by the TOE's implementation of PBKDF.
- FDP_DAR_EXT.1(2)—sensitive data at rest is protected in turn by the platform's use of full disk encryption.
- FDP_DEC_EXT.1—the TOE accesses only the platform hardware resources and sensitive information repositories it needs in order to perform its functions. The TOE documentation clearly identifies these platform hardware resources and sensitive information repositories and justifies its need to access them.
- FDP_NET_EXT.1—the TOE communicates over the network for well-defined purposes. Depending on the function, the use of network resources is user-initiated directly through the TSF or initiated by the TOE itself.

6.4 Identification and Authentication

The TOE uses X.509 certificates for authentication of the following trusted communications: validation of administrator TLS client certificate and validation of TLS server certificates for environmental Tenable components that it communicates with (Nessus, NNM, LCE). It validates X.509 certificates using the path validation algorithm defined in RFC 5280, which can be summarized as follows:

- Check the public key algorithm and parameters of each certificate in the path
- Check the current date/time against the validity period of each certificate in the path
- Check the revocation status of each certificate in the path
- Check the issuer name to ensure it equals the subject name of the previous certificate in the path
- Check name constraints to make sure the subject name is within the permitted subtrees list of all previous CA certificates and not within the excluded subtrees list of any previous CA certificate
- Check the asserted certificate policy OIDs against the permissible OIDs of the previous certificate, including any policy mapping equivalencies asserted by the previous certificate
- Check policy constraints to ensure any explicit policy requirements are not violated
- For all CA certificates, confirm the presence of the basicConstraints extension and that the CA flag is set to TRUE, and ensure the key usage field includes the caSigning purpose
- Check the path length does not exceed any maximum path length asserted in this or a previous certificate
- Check the key usage extension
- Process any other recognized critical extensions.

The TOE validates the extendedKeyUsage field according to the following rules:

- Server certificates presented for TLS shall have the Server Authentication purpose (id-kp 1 with OID 1.3.6.1.5.5.7.3.1) in the extendedKeyUsage field.
- Client certificates presented for TLS shall have the Client Authentication purpose (id-kp 2 with OID 1.3.6.1.5.5.7.3.2) in the extendedKeyUsage field.
- 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.

The TOE does not use any of the other values for the extended KeyUsage field listed in the requirement (i.e., Code Signing, Email Protection, or CMC Registration Authority), so this part of the requirement is trivially satisfied.

The TOE validates the certificate chain to its root to ensure it terminates with a trusted CA certificate. It performs a revocation check on each certificate in the chain (except the root certificate) using Online Certificate Status Protocol (OCSP) in accordance with RFC 6960. In the event the revocation status of a certificate cannot be verified (i.e., the OCSP responder cannot be reached), the TOE rejects the certificate.

Because the TOE's use of the certificate validation function is to validate the authenticity of remote endpoints, the TSF chooses what certificates to use based on what is presented to it as part of establishing the TLS session. The TOE is only assigned one certificate for its own use, so there is only one certificate that it will present in cases where a remote entity may need to validate it.

The Identification and Authentication security function is designed to satisfy the following security functional requirements:

- FIA_X509_EXT.1—the TOE validates X.509 certificates when establishing trusted communications.
- FIA_X509_EXT.2—X.509 certificates are used for TLS. When revocation status of a certificate cannot be determined, the TSF rejects the certificate.

6.5 Security Management

The TOE provides a web-based GUI that requires user authentication to access. As part of initial setup of the TOE, the administrator performing the install must specify an initial username/password that is used to log on to the web GUI; the TOE is not pre-loaded with "default" administrator credentials. These credentials are stored locally and protected by the TSF as per FCS_STO_EXT.1. Following the initial installation, additional accounts can be created.

During general operations, an administrator will typically interact with a deployment of Tenable applications using the TOE. The TOE is used for all aggregation, visualization, and reporting of data that is collected and analyzed by other Tenable applications. However, the TOE does not include the ability to directly modify the initial configuration settings of the environmental components.

The TOE is installed into `/opt/sc`.

All directories containing TOE software and data are configured by default in such a manner that nothing is world-writable. Configuration settings that affect the TOE's interaction with the host OS platform are stored in `/etc`.

The TOE supports the following security-relevant management functions:

- Configuration of transmission of system's hardware, software, or configuration information
  - Used to configure parameters related to the collection of system configuration, network, and log data performed by environmental components, such as identifying the collection targets and configuring periodic intervals for data collection (or manually initiating data collection)
- Configuration of transmission of application state (crashdump) information
  - Includes a diagnostics utility that is manually-initiated and is used to collect application state information that can be sent to Tenable for troubleshooting purposes
- Configuration of presentation (reporting/dashboards/analytics) of collected system and network data
  - Includes a number of tools and views that aggregate, organize, summarize, and report on collected data.

The Security Management security function is designed to satisfy the following security functional requirements:

- FMT_CFG_EXT.1—the TOE requires credentials to be defined before administrative use. The TOE is protected from direct modification by untrusted users via its host OS platform.
- FMT_MEC_EXT.1—configuration settings for the TOE are stored in an appropriate location in its host OS platform.
- FMT_SMF.1—administrators can use the TSF to configure the collection of system and network data from the TOE's operational environment and the presentation and analysis of this data, or to collect application state information that is used for troubleshooting.

6.6 Privacy

The TOE's primary function is to be part of a system that examines organizational assets for configuration or operational states that may indicate the presence of a vulnerability or misuse of organizational resources. To this end, the TOE receives transmissions of data about system configuration and network activity for aggregation, analysis, and reporting. The TOE is not responsible for the collection or transmission of PII. The TOE accepts administrative credentials as part of the GUI login process but user account information is not considered to be PII.

The Privacy security function is designed to satisfy the following security functional requirements:

- FPR_ANO_EXT.1—the TOE prevents the unnoticed/unauthorized transmission of PII across a network by not having functionality that is intended for such transmissions.

6.7 Protection of the TSF

The TOE implements several mechanisms to protect against exploitation. The TOE implements address space layout randomization (ASLR) through the use of the `-fPIC` compiler flag and relies fully on its underlying host platforms to perform memory mapping. The TOE also does not use both `PROT_WRITE` and `PROT_EXEC` on the same memory regions. There is no situation where the TSF maps memory to an explicit address. The TOE is written in C. It is compiled with stack overflow protection through the use of the `-fstack-protector-strong` GCC compiler flag. The TOE has a web-based front-end, based on PHP. This is interpreted code to which compilation instructions do not apply.

The TOE is designed to run on a host OS platform where SElinux is enabled and enforcing. The TOE uses only documented platform APIs. Appendix A.1 lists the APIs used by the TOE. The TOE also makes use of third-party libraries. Appendix A.2 lists the libraries used by the TOE. The TOE is versioned using semver (Semantic Versioning) in the format x.y(.z) where x is the major version, y is the minor version, and the optional z is the patch version; SWID is not used. The TOE is a standalone application that is not natively bundled as part of a host OS.

An administrator can identify the current running version of the TOE through both platform and TSF-mediated methods. The TOE is installed as an RPM and will identify its version in RPM itself. The TOE will also return its version information if its binary is invoked with the `-v` flag on the OS platform. Administrators can also check the version of the TOE via the About menu on the web GUI.

The TOE can check for software updates and notify the administrator of their availability by displaying a "Bell" icon in the upper right corner of the GUI. The administrator obtains updates by downloading them directly from Tenable's website (https://www.tenable.com/downloads/security-center) or through a package manager such as `yum`. The TOE will not download, modify, replace, or update its own binary code. The TOE is packaged as a `.rpm` file. This is digitally signed by Tenable using 4096-bit RSA. Removing (uninstalling) the product will remove all executable code from the host system.

The Protection of the TSF security function is designed to satisfy the following security functional requirements:

- FPT_AEX_EXT.1—the TOE interacts with its host OS platform in a manner that does not expose the system to memory-related exploitation.
- FPT_API_EXT.1—the TOE uses documented platform APIs.
- FPT_IDV_EXT. 1—the TOE is versioned using semver.
- FPT_LIB_EXT.1—the set of third-party libraries used by the TOE is well-defined.
- FPT_TUD_EXT.1—there is a well-defined method for checking what version of the TOE is currently installed and whether updates to it are available. Updates are signed by the vendor and validated by the host OS platform prior to installation.
- FPT_TUD_EXT.2—the TOE is distributed using the format of the platform-supported package manager.

6.8 Trusted Path/Channels

In the evaluated configuration, the TOE uses both its own cryptographic implementation and its host OS platform to encrypt sensitive data in transit. Listed below are the various external interfaces to the TOE that rely on trusted communications.

Between TOE and operational environment:

- Between user and TOE web GUI
  - Communications use mutually-authenticated TLS/HTTPS (TOE is server)
  - TCP port 443
  - Used to secure administrator interactions with the TOE

Between TOE and environmental Tenable components:

- Between TOE and Nessus
  - Communications use XML RPCs over mutually-authenticated TLS/HTTPS (TOE is client and Nessus is server)
  - Configurable TCP port, 8834 is default
  - Used by the TOE to retrieve Nessus/Nessus Agent scan results from Nessus
- Between TOE and Nessus Network Monitor
  - Communications use mutually-authenticated TLS (TOE is client)
  - Configurable TCP port, 8835 is default
  - Used by the TOE to collect network traffic data from NNM
- Between TOE and Log Correlation Engine
  - Communications use TLS (TOE is client)
  - TCP port 1243
  - Used by the TOE to collect unaltered bulk log data aggregated by LCE
- Between TOE and Log Correlation Engine
  - Communications use SSH (implemented by the operational environment)
  - TCP port 22
  - Used by the TOE to collect log data that has already been parsed by LCE as potential vulnerabilities

All use of SSH is accomplished through TSF invocation of the RHEL `ssh` utility.

The Trusted Path/Channels security function is designed to satisfy the following security functional requirements:

- FTP_DIT_EXT.1—the TOE implements TLS and HTTPS and invokes platform-provided SSH to secure data in transit between itself and its operational environment.

7 Protection Profile Claims

This ST is conformant to the *Protection Profile for Application Software*, Version 1.4, 7 October 2021 ([APP_PP]) and *Functional Package for Transport Layer Security (TLS)*, Version 1.1, 1 March 2019 ([TLS_PKG]) along with all applicable errata and interpretations from the certificate issuing scheme.

The TOE consists of a software application that runs on a Linux operating system as its platform.

As explained in section 3, Security Problem Definition, the Security Problem Definition of [APP_PP] has been included by reference into this ST.

As explained in section 4, Security Objectives, the Security Objectives of [APP_PP] has been included by reference into this ST.

All claimed SFRs are defined in [APP_PP] and [TLS_PKG]. All mandatory SFRs are claimed. No optional or objective SFRs are claimed. Selection-based SFR claims are consistent with the selections made in the mandatory SFRs that prompt their inclusion.

8 Rationale

This Security Target includes by reference the App PP Security Problem Definition, Security Objectives, and Security Assurance Requirements. The Security Target does not add, remove, or modify any of these items. Security Functional Requirements have been reproduced with the Protection Profile operations completed. All selections, assignments, and refinements made on the claimed Security Functional Requirements have been performed in a manner that is consistent with what is permitted by the App PP and TLS Package. The proper set of selection-based requirements have been claimed based on the selections made in the mandatory requirements. Consequently, the claims made by this Security Target are sufficient to address the TOE's security problem. Rationale for the sufficiency of the TOE Summary Specification is provided below.

8.1 TOE Summary Specification Rationale

This section in conjunction with Section 0, the

The [TLS_PKG] does contain evaluation activities for how to evaluate its SFR claims as part of the evaluation of ASE_TSS.1, AGD_OPE.1, AGD_PRE.1, and ATE_IND.1. All Security Functional Requirements specified by [TLS_PKG] are evaluated in the manner specified in that package.

TOE Summary Specification, provides evidence that the security functions meet the TOE security requirements. Each description includes rationale indicating which requirements the corresponding security functions satisfy. The combined security functions work together to satisfy all of the security requirements. The security functions described in Section 6 are necessary for the TSF to enforce the required security functionality. Table 8 demonstrates the relationship between security requirements and functions.

Table 8: Security Functions vs. Requirements Mapping

	Cryptographic Support	User Data Protection	Identification and Authentication	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FCS_CKM_EXT.1	X						
FCS_CKM.1/AK	X						
FCS_CKM_EXT.1/PBKDF	X						
FCS_CKM.2	X						
FCS_COP.1/SKC	X						
FCS_COP.1/Sig	X						
FCS_COP.1/Hash	X						
FCS_COP.1/KeyedHash	X						
FCS_HTTPS_EXT.1/Client	X						
FCS_HTTPS_EXT.1/Server	X						

	Cryptographic Support	User Data Protection	Identification and Authentication	Security Management	Privacy	Protection of the TSF	Trusted Path/Channels
FCS_HTTPS_EXT.2	X						
FCS_RBG_EXT.1	X						
FCS_RBG_EXT.2	X						
FCS_STO_EXT.1	X						
FCS_TLS_EXT.1	X						
FCS_TLSC_EXT.1	X						
FCS_TLSC_EXT.2	X						
FCS_TLSC_EXT.5	X						
FCS_TLSS_EXT.1	X						
FCS_TLSS_EXT.2	X						
FDP_DAR_EXT.1(1)		X					
FDP_DAR_EXT.1(2)		X					
FDP_DEC_EXT.1		X					
FDP_NET_EXT.1		X					
FIA_X509_EXT.1			X				
FIA_X509_EXT.2			X				
FMT_CFG_EXT.1				X			
FMT_MEC_EXT.1				X			
FMT_SMF.1				X			
FPR_ANO_EXT.1					X		
FPT_AEX_EXT.1						X	
FPT_API_EXT.1						X	
FPT_IDV_EXT.1						X	
FPT_LIB_EXT.1						X	
FPT_TUD_EXT.1						X	
FPT_TUD_EXT.2						X	
FTP_DIT_EXT.1							X

A TOE Usage of Third-Party Components

This Appendix lists the platform APIs and third-party libraries that are used by the TOE.

A.1 Platform APIs

Listed below are the platform APIs used by the TOE. Note that these APIs do not necessarily relate to the TOE functionality claimed in the Security Target; however, since they are bundled with the product itself they are disclosed since a vulnerability in outside the logical boundary of the product could still present an exploitable vulnerability.

date, ls, ln, tar, rm, cp, zip/unzip, gzip/gunzip, ssh, scp, rsync (LCE results), chown, chmod, ps (diagnostics), zcat, find, file, exec (in proc_open), cd, pwd, echo, bash (patches), cat, uname, uptime, ipcs, ip, dmessage, free, vmstat, df, du, mdstat, chkconfig, sort, ulimit, rpm, ss, iptables, iostat, lsof, mkdir, xmllint, getconf, su, env, service

A.2 Third-Party Libraries

Listed below are the third-party libraries used by the TOE. Note that these libraries do not necessarily relate to the TOE functionality claimed in the Security Target; however, since they are bundled with the product itself they are disclosed since a vulnerability in outside the logical boundary of the product could still present an exploitable vulnerability.

Library	**Version**
Apache FOP	2.8
Apache HTTP Server	2.4.57
Apache Portable Runtime	1.7.3
Apache Portable Runtime Utils	1.6.3
Backbone	1.4.1
Bootstrap	3.4.1
ChartDirector	7.0
composer	2.5.7
D3	3.3.8
fusioncharts	3.18.0
fusioncharts.charts	3.18.0
fusioncharts.gantt	3.18.0
fusioncharts.theme.fusion	3.18.0
fusioncharts.timeseries	3.18.0
GMP	6.3
Handlebars	4.7.7
jQuery	3.6.0

Library	Version
jQuery UI	1.13.2
libcurl	8.3.0
libmcrypt	2.5.8
libssh2	1.11.0
mcrypt	1.0.6
Moment Timezone	0.5.38
MomentJS	2.29.4
OpenLDAP	2.6.6
OpenSSL	3.0.10
PCRE / libpcre	8.45
PHP	8.2.8
PHP SourceGuardian Loaders	14.0.1
PHP SSH2 Extension	1.10.0
PHPMailer	6.8.0
RapidJSON	1.1.0
SimpleSAMLPHP	2.0.4
sqlite	3.40.1
SSH PECL	1.3.1
UnderscoreJS	1.13.6
zlib	1.2.13