Palo Alto Networks Cortex XSOAR Server 6.6 Security Target

Version: 1.0
Date: September 16, 2022

XSOAR

Palo Alto Networks, Inc.
www.paloaltonetworks.com

Table of Contents

LIST OF FIGURES

LIST OF TABLES

1. Security Target Introduction

This section identifies the Security Target (ST) and Target of Evaluation (TOE) identification, ST conventions, ST conformance claims, and the ST organization. The TOE is the Cortex XSOAR Server version 6.6.

The Palo Alto Networks Cortex XSOAR (eXtended Security Orchestration, Automation and Response) combines security orchestration, threat intel and incident management, and automated investigation.

The focus on this evaluation is on the TOE functionality supporting the claims in the Protection Profile for Application Software.

The Security Target contains the following additional sections:

- Product Description (Section 2)
- Security Problem Definition (Section 3)
- Security Objectives (Section 4)
- IT Security Requirements (Section 5)
- TOE Summary Specification (Section 6)
- Protection Profile Claims (Section 7)
- Rationale (Section 8).

1.1 Security Target, TOE and CC Identification

ST Title – Palo Alto Networks Cortex XSOAR Server 6.6 Security Target

ST Version – Version 1.0

ST Date – September 16, 2022

TOE Identification –

- Cortex XSOAR Server 6.6.0 running on RedHat 8[1].

TOE Developer – Palo Alto Networks, Inc.

Evaluation Sponsor – Palo Alto Networks, 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: This ST is conformant to:

- Protection Profile for Application Software, Version 1.4, October 7, 2021 [APPSW].
- Functional Package for Transport Layer Security (TLS), Version 1.1, February 12, 2019 [PKGTLS]

[1] While the TOE was tested on RHEL 8, additional Linux environments such as RHEL 7, Ubuntu (18.04, 20.04), Oracle Linux 7, and Amazon Linux 2 are supported as well. This is vendor affirmed. The minimum hardware requirements on those system are 16 CPU cores, 32 GB of memory, and 1 TB SSD hard disk.

This TOE and ST are conformant to Parts 2 (extended) and 3 (extended) of Common Criteria Version 3.1, Revision 5.

The following NIAP Technical Decisions apply to this APPSW, and have been accounted for in the ST development[2]::

- TD0624 – Addition of DataStore for Storing and Setting Configuration Options
- TD0626 – FCS_COP.1 Keyed Hash Selections
- TD0628 – Addition of Container Image to Package Format
- TD0659 - Change to Required NIST Curves for FCS_CKM.1/AK
- TD0655 - Mutual authentication in FTP_DIT_EXT.1 for SW App

The TOE and ST is package-name conformant to [PKGTLS].

The following NIAP Technical Decisions apply to this PKGTLS, and have been accounted for in the ST development:

- 0442 – Updated TLS Ciphersuites for TLS package
- 0469 – Modification of test activity for FCS_TLSS_EXT.1.1 test 4.1
- 0499 – Testing with pinned certificates
- 0513 – CA Certificate loading
- 0588 – Session Resumption Support in TLS Package

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. In the ST, iteration is indicated by a number in parentheses placed at the end of the component. For example, FDP_ACC.1 (1) and FDP_ACC.1 (2) indicate that the ST includes two iterations of the FDP_ACC.1 requirement, (1) and (2).
    - Assignment: allows the specification of an identified parameter. Assignments are indicated using italicized and are surrounded by brackets (e.g., [*assignment*]). Note that an assignment within a selection would be identified in italics and with embedded bold brackets (e.g., [***[selected-assignment]***]).
    - Selection: allows the specification of one or more elements from a list. Selections are indicated using bold and are surrounded by brackets (e.g., [**selection**]).
    - Refinement: allows the addition of details. Refinements are indicated using bold, for additions, and strike-through, for deletions (e.g., "… **all** objects …" or "… some **big** things …"). Note that 'cases' that are not applicable in a given SFR have simply been removed without any explicit identification.
- Other sections of the ST – Other sections of the ST use bolding to highlight text of special interest, such as captions.

[2] TD0650 does not apply as the TOE is not a VPN Client

- The ST does not highlight operations that have been completed by the PP and EP authors.

1.3.1 Terminology

The following terms and abbreviations are used in this ST:

Playbooks	Playbooks are at the heart of the Cortex XSOAR system. They enable you to automate many of your security processes, including, but not limited to handling your investigations and managing your tickets.

1.3.2 Acronyms

AES	Advanced Encryption Standard
CBC	Cipher-Block Chaining
CC	Common Criteria for Information Technology Security Evaluation
CEM	Common Evaluation Methodology for Information Technology Security
CM	Configuration Management
CLI	Command Line Interface
DH	Diffie-Hellman
DRBG	Deterministic Random Bit Generator
FIPS	Federal Information Processing Standard
FSP	Functional Specification
GCM	Galois/Counter Mode
GUI	Graphical User Interface
HMAC	Hashed Message Authentication Code
HTTPS	Hypertext Transfer Protocol Secure
IP	Internet Protocol
IPv4	Internet Protocol version 4
Ipv6	Internet Protocol version 6
NIST	National Institute of Standards and Technology
PP	Protection Profile
RHEL	Red Hat Enterprise Linux
RSA	Rivest, Shamir and Adleman (algorithm for public-key cryptography)
SAR	Security Assurance Requirement
SFR	Security Functional Requirement
SHA	Secure Hash Algorithm
SOAR	Security Orchestration, Automation and Response
SOC	Security Operation Center
SSH	Secure Shell
SSL	Secure Socket Layer
ST	Security Target
TLS	Transport Layer Security
TOE	Target of Evaluation
TSF	TOE Security Functions
URL	Uniform Resource Locator
VM	Virtual Machine

2. Product Description

The Target of Evaluation (TOE) is the Palo Alto Networks Cortex[3] XSOAR Server 6.6. Cortex XSOAR combines security orchestration, incident management, and interactive investigation into a seamless experience. The orchestration component is designed to automate security product tasks and weave in human analyst tasks and workflows.

2.1 TOE Overview

The TOE is a software application that runs on various Linux environments (server/workstation) to provide SOC team with detection and automatic response capabilities. The Server provides UI functionality and playbook detection/response functionality, while Engine in the operational environment is used to efficiently share the workload (e.g., load-balancing), thereby speeding up execution time. In essence, the Engine is like an extension of the Server used to offload tasks to it.

The Server provides security management functions and interface via web UI protected by HTTPS and communicates with the Engine(s) through TLSv1.2 protected channels. The Server implements TLS server functionality while the Engine, which is in the operational environment, implements TLS client functionality. The TOE and Engine can be deployed on the same workstation/server, or they can be deployed on separate devices.

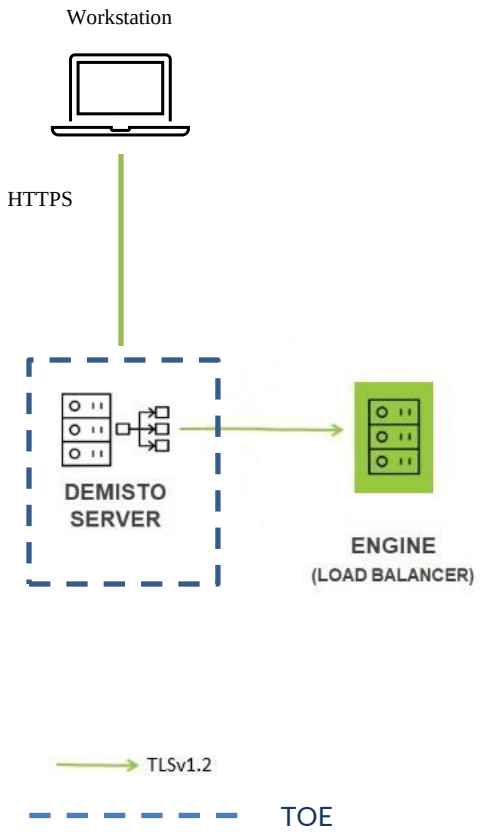

Figure 1: TOE

[3] Cortex was formerly known as Demisto.

FIPS-CC Mode of Operation

The FIPS-CC Version of Cortex XSOAR are, by default, compliant with all FIPS and CC Approved algorithms, protocols, and key sizes requirements. The FIPS-CC mode is enabled by default and cannot be disabled. Additional configurations must be performed to enable X509v3 certification validation, revocation checking, and mutual authentication. All configuration steps are fully documented in the CC Evaluated Configuration Guide (also known as the AGD).

2.2 TOE Architecture

The TOE is a software solution that is comprised of items listed in Section 2.2.1 and 2.2.2. The software is available for download from the Palo Alto Networks support site.

The following diagram depicts the software architecture of the TOE.

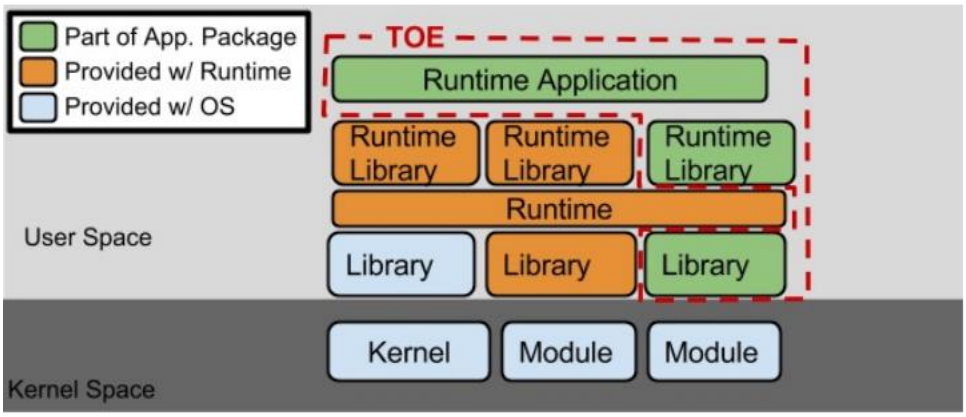

Figure 2: TOE Architecture

2.2.1 Physical Boundaries

The physical boundary of the TOE is the application installed and running on a supported platform (i.e., Linux operating systems).

2.2.1.1 Software Requirements

The TOE runs on an operating system that includes RHEL 8, RHEL 7, Ubuntu (18.04, 20.04), Oracle Linux 7, or Amazon Linux 2.

2.2.1.2 Hardware Requirements

The TOE must be installed on server system with the minimum hardware requirements specified below.

Minimum hardware: 16 CPU (minimum); 32 GB RAM (minimum); 1 TB of disk space (minimum)

The TOE was installed and tested on the following platforms:

- Redhat Enterprise Linux v8.4 - Processor: Intel Xeon Gold 6248 (Cascade Lake)

2.2.2 Logical Boundaries

This section summarizes the security functions provided by the TOE:

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

2.2.2.1 Cryptographic Support

The TOE implements CAVP validated cryptographic algorithms that provide key management, random bit generation, encryption/decryption, digital signature and cryptographic hashing and keyed-hash message authentication features in support of cryptographic protocols such as TLS.

2.2.2.2 User Data Protection

The TOE provides no access to hardware resources or sensitive information repositories. The TOE restricts network communication to application-initiated communication between the TOE and Engine(s), and remotely user-initiated communication from administrators. All sensitive application data (e.g., API keys, passphrase) are stored encrypted using AES-256 in CBC mode.

2.2.2.3 Identification and Authentication

The TOE authenticates all users using password-based or X509v3 certificate-based method. This is configurable and by default, password-based method is used.

2.2.2.4 Security Management

The TOE provides access to the security management functions via the web UI. Identification and authentication are required before accessing the UI. In addition, the operating system can provide some configuration options for the TOE. In that case, the operating system I&A method and privileges will be used and enforced.

2.2.2.5 Privacy

The TOE does not transmit PII over the network.

2.2.2.6 Protection of the TSF

The TOE implements a number of functions to ensure that it is protected against tampering and corruption. These mechanisms include utilizing platform APIs, memory mapping, and stack-based buffer overflow protection. Palo Alto Networks provides customers with a means of updating their TOE using trusted updates. These trusted updates (signed RPM package) are securely delivered over HTTPS website and verified using approved digital signature methods. All of these updates are properly signed using RSA 2048 with SHA-256 and is verified by the operating system mechanism. In addition, the TOE image is protected with FIPS Software integrity test at power-up (e.g., when the application is started or is reloaded).

2.2.2.7 Trusted Path/Channels

The TOE protects interactive communication with remote administrators using HTTP over TLS (HTTPS). TLS ensures both integrity and disclosure protection. TLS also protects communication between the Server and its' Engine(s).

2.3 TOE Documentation

Palo Alto Networks Inc. offers a series of documents that describe the installation of Palo Alto Networks Cortex XSOAR as well as guidance for subsequent use and administration of the applicable security features.

For Cortex XSOAR v6.6, these documents include:

- Palo Alto Networks Common Criteria Evaluated Configuration Guide (CCECG) for Cortex XSOAR Server and Engine 6.6, August 3, 2022

3. Security Problem Definition

This security target includes by reference the Security Problem Definition (composed of organizational policies, threat statements, and assumption) from [APPSW].

In general, the [APPSW] has presented a Security Problem Definition appropriate for software applications, and as such, is applicable to the TOE.

The following threats are directly from the [APPSW]:

T. NETWORK_ATTACK	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may engage in communications with the application software or alter communications between the application software and other endpoints in order to compromise it.
T.NETWORK_EAVESDROP	An attacker is positioned on a communications channel or elsewhere on the network infrastructure. Attackers may monitor and gain access to data exchanged between the application and other endpoints.
T.LOCAL_ATTACK	An attacker can act through unprivileged software on the same computing platform on which the application executes. Attackers may provide maliciously formatted input to the application in the form of files or other local communications.
T.PHYSICAL_ACCESS	An attacker may try to access sensitive data at rest.

The following assumptions are made as drawn directly from the [APPSW]:

A. PLATFORM	The TOE relies upon a trustworthy computing platform with a reliable time clock for its execution. This includes the underlying platform and whatever runtime environment it provides to the TOE.
A. PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software in compliance with the applied enterprise security policy.
A. PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software in compliance with the applied enterprise security policy.

.

4. Security Objectives

The sections below identify the security objectives for the TOE and for the operational environment. These security objectives identify the responsibilities of the TOE and the operational environment in meeting security needs.

4.1 Security Objectives for the TOE

O.INTEGRITY	Conformant TOEs ensure the integrity of their installation and update packages, and also leverage execution environment-based mitigations. Software is seldom, if ever, shipped without errors. The ability to deploy patches and updates to fielded software with integrity is critical to enterprise network security. Processor manufacturers, compiler developers, execution environment vendors, and operating system vendors have developed execution environment-based mitigations that increase the cost to attackers by adding complexity to the task of compromising systems. Application software can often take advantage of these mechanisms by using APIs provided by the runtime environment or by enabling the mechanism through compiler or linker options.
O.QUALITY	To ensure quality of implementation, conformant TOEs leverage services and APIs provided by the runtime environment rather than implementing their own versions of these services and APIs. This is especially important for cryptographic services and other complex operations such as file and media parsing. Leveraging this platform behavior relies upon using only documented and supported APIs.
O.MANAGEMENT	To facilitate management by users and the enterprise, conformant TOEs provide consistent and supported interfaces for their security-relevant configuration and maintenance. This includes the deployment of applications and application updates through the use of platform-supported deployment mechanisms and formats, as well as providing mechanisms for configuration. This also includes providing control to the user regarding disclosure of any PII.
O.PROTECTED_STORAGE	To address the issue of loss of confidentiality of user data in the event of loss of physical control of the storage medium, conformant TOEs will use data-at-rest protection. This involves encrypting data and keys stored by the TOE in order to prevent unauthorized access to this data. This also includes unnecessary network communications whose consequence may be the loss of data.
O.PROTECTED_COMMS	To address both passive (eavesdropping) and active (packet modification) network attack threats, conformant TOEs will use a trusted channel for sensitive data. Sensitive data includes cryptographic keys, passwords, and any other data specific to the application that should not be exposed outside of the application.

4.2 Security Objectives for the Operational Environment

OE.PLATFORM	The TOE relies upon a trustworthy computing platform for its execution. This includes the underlying operating system and any discrete execution environment provided to the TOE.
OE.PROPER_USER	The user of the application software is not willfully negligent or hostile, and uses the software within compliance of the applied enterprise security policy.
OE.PROPER_ADMIN	The administrator of the application software is not careless, willfully negligent or hostile, and administers the software within compliance of the applied enterprise security policy.

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):

- *Protection Profile for Application Software, Version 1.4, 7 October 2021* [APPSW],
- *Functional Package for Transport Layer Security (TLS), Version 1.1 [PKGTLS]*

The SARs are the set of SARs specified in [APPSW].

5.1 Extended Requirements

All extended requirements in this ST have been drawn from the [APPSW] and [PKGTLS]. The [APPSW] and [PKGTLS] define all the extended SFRs (*_EXT.1) and since they are not redefined in this ST, the [APPSW] and [PKGSTLS] should be consulted for more information in regard to those CC extensions.

5.2 TOE Security Functional Requirements

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

Table 1 TOE Security Functional Components

Requirement Class	Requirement Component
FCS: Cryptographic Support	FCS_CKM.1 Cryptographic Key Generation Services
	FCS_CKM.1/AK Cryptographic Asymmetric Key Generation
	FCS_CKM.2 Cryptographic Key Establishment
	FCS_COP.1/SKC Cryptographic Operation – Encryption/Decryption
	FCS_COP.1/Hash Cryptographic Operation – Hashing
	FCS_COP.1/KeyedHash Cryptographic Operation – Keyed-Hash Message Authentication
	FCS_COP.1/Sig Cryptographic Operation -- Signing
	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
	FCS_TLSS_EXT.1 TLS Server Protocol
	FCS_TLSS_EXT.2 TLS Server Support for Mutual Authentication
FDP: User Data Protection	FDP_DEC_EXT.1 Access to Platform Resources
	FDP_NET_EXT.1 Network Communications
	FDP_DAR_EXT.1 Encryption of Sensitive Application Data

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_MEC_EXT.1 Supported Configuration Mechanism
	FMT_CFG_EXT.1 Secure by Default Configuration
	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_API_EXT.1 Use of Supported Services and APIs
	FPT_AEX_EXT.1 Anti-Exploitation Capabilities
	FPT_TUD_EXT.1 Integrity for Installation and Update
	FPT_TUD_EXT.2 Integrity for Installation and Update
	FPT_IDV_EXT.1 Software Identification and Versions
	FPT_LIB_EXT.1 Use of Third Party Libraries
FTP: Trusted Path/Channel	FTP_DIT_EXT.1 Protection of Data in Transit

5.2.1 Cryptographic Support (FCS)

FCS_CKM.1 – Cryptographic Key Generation Services

FCS_CKM.1.1 The application shall [

- ***implement asymmetric key generation***

].

FCS_CKM.1/AK – Cryptographic Asymmetric Key Generation

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]***

].

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:

[

- ***[Elliptical 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"]***

].

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

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***]**.**

FCS_COP.1/Hash – Cryptographic Operation – Hashing

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

- ***SHA-1,***
- ***SHA-256,***
- ***SHA-384***
- ***SHA-512***

] and message digest sizes [

- ***160***
- ***256,***
- ***384***
- ***512***

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

Application Note*: Support for SHA-1 is needed for digital signature verification only. Users should use digital signature with SHA-256 or higher per NIST SP 800-131A guideline.*

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

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, 384, 512*] 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.*

FCS_COP.1/Sig – Cryptographic Operation – Signing

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 4,*
- ***ECDSA schemes*** *using "NIST curves" P-256, P-384 and [**no other curves**] that meet the following: FIPS PUB 186-4, "Digital Signature Standard (DSS)", Section 5*

].

FCS_HTTPS_EXT.1/Server – HTTPS Protocol

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 Functional Package for TLS.

FCS_HTTPS_EXT.2 – HTTPS Protocol with Mutual Authentication

FCS_HTTPS_EXT.2.1 The application shall [***establish or not establish the connection based on an administrative or user setting***] if the peer certificate is deemed invalid.

FCS_RBG_EXT.1 – Random Bit Generation Services

FCS_RBG_EXT.1.1 The application shall [

- ***implement DRBG functionality***

] for its cryptographic operations.

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.

FCS_STO_EXT.1 – Storage of Credentials

FCS_STO_EXT.1.1 The application shall [

- ***Implement functionality to securely store [API keys, passphrase] according to [FCS_COP.1/SKC]***

] to non-volatile memory.

FCS_TLS_EXT.1 – TLS Protocol

FCS_TLS_EXT.1 The product shall implement [

- ***TLS as a server***

].

FCS_TLSS_EXT.1 - TLS Server Protocol

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

- ***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***

] and no other cipher suites, and also supports functionality for [

- ***mutual authentication,***
- ***session resumption based on session tickets according to RFC 5077***

].

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, secp521r1**] and no other curves*

].

FCS_TLSS_EXT.2 - TLS Server Protocol with mutual authentication

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

FCS_TLSS_EXT.2.2 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.

***Application Note**: The TOE only verifies the SAN in the X509v3 certificate. The certificate must have a SAN field or the connection will be terminated.*

5.2.2 User Data Protection (FDP)

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 [

- ***no sensitive information repositories***

].

***Application Note**: The TOE is not designed to be installed on a mobile device (such as a phone or tablet).*

FDP_NET_EXT.1 – Network Communications

FDP_NET_EXT.1.1 The application shall restrict network communication to [

- ***respond to [web UI users-initiated HTTPS connection],***
- ***[communication with Engine]***

].

FDP_DAR_EXT.1 – Encryption of Sensitive Application Data

FDP_DAR_EXT.1.1 The application shall [

- ***protect sensitive data in accordance with FCS_STO_EXT.1***

] in non-volatile memory.

5.2.3 Identification and Authentication (FIA)

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 verification.
- 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.
- 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.

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 [***TLS***].

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

5.2.4 Security Management (FMT)

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.***

].

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.

FMT_SMF.1 – Specification of Management Functions

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

- ***[User management***
- ***Password change***
- ***Password complexity management***
- ***TLS/X509 configuration***
- ***Querying the current version of the TOE***
- ***Encrypting API keys and passphrase in the database]***

]

5.2.5 Privacy (FPR)

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)

FPT_API_EXT.1 – Use of Supported Services and APIs

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

FPT_AEX_EXT.1 – Anti-Exploitation Capabilities

FPT_AEX_EXT.1.1 The application shall not request to map memory at an explicit address expect 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 compiled with stack-based buffer overflow protection enabled.

FPT_TUD_EXT.1 – Integrity for Installation and Update

FPT_TUD_EXT.1.1 The application shall [***leverage the platform***] to check for updates and patches to the application software.

FPT_TUD_EXT.1.2 The application shall [***provide the ability***] 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***].

FPT_TUD_EXT.2 – Integrity for Installation and Update

FPT_TUD_EXT.2.1 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.

FPT_IDV_EXT.1 – Software Identification and Versions

FPT_IDV_EXT.1.1 The application shall be versioned with **[*[XSOAR software version]*].**

FPT_LIB_EXT.1 – Use of Third Party Libraries

FPT_LIB_EXT.1.1 The application shall be packaged with only [*Golang*, *BoringCrypto*]

5.2.7 Trusted Path/Channel (FTP)

FTP_DIT_EXT.1 – Protection of Data in Transit

FTP_DIT_EXT.1.1 The application shall [

- ***encrypt all transmitted [data] with [HTTPS as a server in accordance with FCS_HTTPS_EXT.1/Server, HTTPS as a server using mutual authentication in accordance with FCS_HTTPS_EXT.2, TLS as a server as defined in the Functional Package for TLS and also supports functionality for [mutual authentication]***

] 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 [APPSW].

Table 2 Assurance Components

Requirement Class	Requirement Component
ASE: Security Target	ASE_CCL.1 Conformance claims
	ASE_ECD.1 Extended components definition
	ASE_INT.1 ST introduction
	ASE_OBJ.1 Security objectives
	ASE_REQ.1 Security requirements
	ASE_SPD.1 Security problem definition
	ASE_TSS.1 TOE summary specification
ADV: Development	ADV_FSP.1 Basic functional specification
AGD: Guidance Documents	AGD_OPE.1: Operational user guidance
	AGD_PRE.1: Preparative procedures
ALC: Life-Cycle Support	ALC_CMC.1 Labelling 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

6. TOE Summary Specification

This chapter describes the security functions:

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

6.1 Cryptographic Support

FCS_CKM.1 FCS_CKM.1/AK	The TOE does not provide users with the capability to generate certificates[4]. Platform Administrators can set/load client or server certificates into the certificate store of the platform (i.e., keypair generated on the platform) that the TOE is running on. During a TLS handshake, the TOE utilizes ECC scheme to generate asymmetric keys with NIST curves that include P-256 and P-384 that adhere to the NIST Special Publication 186-4. For details regarding the algorithms supported and their CAVP certificates, see table below.
FCS_CKM.2	The TOE implements key establishment methods using elliptical curve key establishment scheme (ECDHE). The curves utilized by the TOE include P-256 and P-384 as defined in NIST SP 800-56A (revision 3).
FCS_COP.1/SKC	The TOE can encrypt/decrypt using AES-CBC mode (as defined in NIST SP 800-38A) and AES-GCM mode (as defined in NIST SP 800-38D) with key sizes 128-bits and 256-bits. Corresponding CAVP certificates for these algorithms are present in table below.
FCS_COP.1/Hash	The TOE uses hash functions that include SHA-1, SHA-256, SHA-384, and SHA-512 as defined in FIPS 180-4. The digest sizes include 160-bits, 256-bits, 384-bits, and 512-bits that are compliant with FIPS 180-4. The hashing capabilities are utilized for digital signature verification and generation and software integrity checks. SHA-1 is not used for generating digital signatures as noted in SP 800-131A but is used only for verification for legacy purpose. The TOE uses SHA-256 and SHA-384 hashing as part of generating digital signatures. SHA-512 is used as part of the software integrity power-up test. Corresponding CAVP certificates for these algorithms are present in table below.
FCS_COP.1/KeyedHash	The TOE supports the use of a Keyed-Hash Message Authentication algorithms that include HMAC-SHA-256, HMAC-SHA-384, and HMAC-SHA-512. They include key sizes of 256 bits, 384-bits, and 512-bits respectively. The HMAC-SHA functions are used as part of the TOE's integrity check (HMAC-SHA-512) to ensure that it has not been tampered and is additionally used as part of the TLS handshake (HMAC-SHA-256 and HMAC-SHA-384). Corresponding CAVP certificates for these algorithms are present in table below.

[4] The TOE does come with pre-installed self-signed certificates that it protects with a randomly generated passphrase. This passphrase can be encrypted in its database using configuration setting. This self-signed certificate is intended for first-time startup, and the customers should replace this certificate with their own.

FCS_COP.1/Sig	Both RSA and ECDSA schemes are used for TLS functions with Approved key sizes. These include RSA 2048-bits and 3072-bits. For ECDSA, they include the curve sizes P-256 and P-384. During TLS handshakes, these certificates are used for peer authentication to verify the server's identity (and/or client, if configured). These certificates are also used by the TOE to present its identity to the peer. Corresponding CAVP certificates and the relevant schemes for these algorithms are present in the table below.
FCS_HTTPS_EXT.1/Server FCS_HTTPS_EXT.2	The TOE provides a web UI protected by HTTPS (complies with RFC 2818) to the administrators. The TOE acts as an HTTPS server and waits for client connections on TCP port 443. The TOE's HTTPS server supports TLSv1.2 only and will deny connection requests from TLS clients with lower versions. By default, mutual authentication is not enabled. In the evaluated configuration, the administrator must enable mutual authentication. The TLS implementation that protects the HTTP connection is the same implementation as described below. The verification of the peer certification (server or client) is configurable.
FCS_RBG_EXT.1 FCS_RBG_EXT.2	The TOE implements DRBG functionality using the CTR_DRBG in AES-256 mode. The DRBG is seeded using the platform's source, which provides a minimum of 256 bits of entropy. The entropy design source description is described in a separate document proprietary to the platform vendor.
FCS_STO_EXT.1	The TOE uses AES-CBC (256-bits) to protect sensitive data such as API keys or passphrase in the TOE database on the non-volatile memory. API keys are used for authentication during API calls. The passphrase is used to encrypt the private key. Restrictive access and permissions to the database are, in addition, enforced by Linux.
FCS_TLSS_EXT.1 FCS_TLSS_EXT.2	All data that is transmitted between the Server and the web users and Engine in the operational environment are encrypted using TLSv1.2. When the TOE is establishing a TLS session, it checks the reference identifier that has been specified by the administrators. These reference identifiers can include FQDN or IP address (not recommended) and are checked by looking at the Subject Alternative Name (SAN). The certificate must have a SAN field, or the TLS connection will be terminated. The TOE does not support wildcards if a certificate is presented with one in it. Certificate pinning is also not supported but session tickets are supported for session resumption based on RFC 5077. CA certificates are stored in the Linux trust anchor or key store. The TOE will also only support the same four cipher suites below. TLS version 1.2 is the only version of TLS supported by the TOE. • *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* The client hello extension supports secp256r1, secp384r1, and secp521r1 NIST curves. During the negotiation of the TLS handshake, X.509v3 certificates are used to verify the client's identity (if configured for mutual authentication).

Table 3 Cryptographic Functions and CAVP Certificates

Function(s)	Standards	Certificates
Asymmetric key generation (FCS_CKM.1 and FCS_CKM.1/AK)		
ECDSA (P-256, P-384 curves)	FIPS PUB 186-4, "Digital Signature Standard (DSS)", Appendix B.4	#A2517
Cryptographic key establishment (FCS_CKM.2)		
Elliptic curve-based scheme	NIST Special Publication 800-56A Revision 3, "Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography"	#A2517
Symmetric encryption/decryption (FCS_COP.1/SKC)		
AES CBC, GCM (128, 256 bits)	FIPS PUB 197 CBC as defined in NIST SP 800-38A GCM as defined in NIST SP 800-38D	#A2517
Cryptographic hashing (FCS_COP.1/Hash)		
SHA-1, SHA-256, SHA-384, SHA-512	FIPS PUB 180-4	#A2517
Cryptographic signature services (FCS_COP.1/Sig)		
RSA with 2048-bit modulus or 3072-bit modulus	FIPS PUB 186-4	#A2517
ECDSA with NIST Curves P-256, P-384	FIPS PUB 186-4	#A2517
Keyed-hash message authentication (FCS_COP.1/KeyedHash)		
HMAC-SHA-256 HMAC-SHA-384 HMAC-SHA-512	FIPS Pub 198-1 FIPS Pub 180-4	#A2517
Deterministic random bit generation (FCS_RBG_EXT.2)		
CTR_DRBG (AES)	NIST SP 800-90A	#A2517

6.2 User Data Protection

FDP_DEC_EXT.1	The TOE accesses the network connectivity of its' platform as identified by the SFR. The TOE does not access any sensitive information repositories.
FDP_NET_EXT.1	The TOE can respond to user-initiated HTTPS connection session or TLS connection with Engine in the operational environment.
FDP_DAR_EXT.1	The TOE protects sensitive data (such as API keys) in accordance with FCS_STO_EXT.1. Please see FCS_STO_EXT.1 for more details.

6.3 Identification and Authentication

FIA_X509_EXT.1 FIA_X509_EXT.2	The TOE implements the ability to perform certificate path validation on the certificate chain that is presented to it by the peer. The certificate path validation begins with the identity certificate presented by the peer, and then proceeds in checking the intermediate CA certificate(s) until it reaches the trusted CA certificate stored in the platform OS trust store. If no trusted CA certificate is found (i.e., the TOE cannot choose to use any certificate for path validation), the session will terminate. The following steps are performed for each certificate in the path: • The public key algorithm/parameters are checked (i.e., RSA/ECDSA key sizes meet requirements of 2048 bits or greater for RSA and 256 bits or greater for ECDSA) • The certificate is checked to make sure it is not expired (i.e., validity period of the certificate must be proper) • The certificate is checked to make sure it is not revoked using OCSP • The issuer name is checked to ensure that it matches the subject name of the previous certificate in the chain • The certificate is checked that it terminates with a trusted CA certificate and that all CA certificate have the basicConstraints extension present (and set to TRUE) • The extendedKeyUsage field is checked such that OCSP certificates and server/client certificates contain the correct OID (e.g., OCSP Signing purpose and Server/Client Authentication purpose) • The key usage extension of the certificate is checked to make sure that it is allowed to sign certificates Certificates that are presented to the TOE must meet the x509v3 requirements as defined in RFC 5280 for TLS. If there are any issues with the certificate presented (as noted above), the TOE will not accept the certificate and reject the connection. A log message will be generated, and an administrator will be required to address the problem noted for the connection to succeed. The TOE also supports revocation checking of the certificate presented using OCSP as specified in RFC 2560. In the event that the certificate is revoked following a check of its status, the TOE will terminate the connection. If the OCSP responder can't be reached, the administrator can configure if the connection is established or terminated prior to the event occurring.

6.4 Security Management

FMT_MEC_EXT.1	The TOE stores its configuration settings and logs in Linux recommended directories. The user must enable the following settings in the TOE application: "Configure certification validation check" "Configure revocation checking" "Enable and configure the SAN trust list"
FMT_CFG_EXT.1	The TOE is shipped with default credential. However, one of the first things the administrator must do is to create new secure credentials. Otherwise, the administrator cannot proceed with the installation and setup. By default, the TOE includes file permissions that protect the TOE's binary and data files from modification from normal unprivileged users. The TOE also includes an integrity check for itself to ensure that no malicious activity occurs.
FMT_SMF.1	The TOE provides several management functions that include the following: ▪ User management ▪ Password change ▪ Password complexity management ▪ TLS/X509 configuration ▪ Querying the current version of the TOE ▪ Encrypting API keys and passphrase in database

6.5 Privacy

FPR_ANO_EXT.1	The TOE does not transmit or store any personally identifiable information about an individual.

6.6 Protection of the TSF

FPT_API_EXT.1	The TOE includes the use of documented Linux system APIs (kernel version 2.6.26 or later). System function calls includes functionality such as • Driver, • Data type, • C library functions, • Kernel functions, • Memory management, • Kernel IPC, • Hardware, and • Firmware.

FPT_AEX_EXT.1	The TOE automatically enables ASLR when the application is compiled (with -pie build flag), and stack-based buffer overflow protection is enabled by default. The TOE does not request any memory mapping at an explicit address. The TOE does not allocate any memory region with both write and execute permissions; users shall also not write user-modifiable files to directories that contain executable files unless they are explicitly told to do so. The TOE is designed to be compatible with the security features that are provided by the platform (SELinux or AppArmor) vendor that is it installed on.
FPT_TUD_EXT.1 FPT_TUD_EXT.2	The TOE has specific versions, which can be queried by the user via the TOE's interface. New versions of the TOE are created by Palo Alto Networks, which an administrator can retrieve to update the current version of the TOE. During the installation process, a digital signature verification check can be performed by the platform to verify that the signed installer has not been modified. All versions of the installer are digitally signed by Palo Alto Networks using RSA 2048 with SHA-256. Customer Support will send verified customers an email with a download link. They also send license key and instructions in the email. The link is only valid for a limited number of downloads and time. The image is securely retrieved from the download server (https://download.demisto.com) The TOE cannot update its own binary code – it relies on the administrator to download and install the new version available. Registered users are emailed when a security update has been published and available on the download server. Customers with support account in the Customer Support Portal (https://support.paloaltonetworks.com/) will also get notifications as well. Based on the Linux distribution, the package can be RPM (e.g., Redhat) or DEB (e.g., Ubuntu).
FPT_IDV_EXT.1	Palo Alto Networks provides a version control system for its software components. The TOE has a unique software versioning that identifies major versions and their subsequent maintenance releases in the following form: <Major>.<Minor>.<Maintenance release> Build version. Major and minor releases introduce new major and minor features for the product, and additional maintenance releases (build version) are released on a regular cadence to fix security issues or bugs identified with the release. Palo Alto Networks provides customers and the public with a Security Advisory page for any security vulnerabilities that have been identified in Palo Alto Networks products (https://security.paloaltonetworks.com/).
FPT_LIB_EXT.1	The TOE utilizes Golang (version 1.16) for its TLS functions and BoringCrypto (version ae223d6138807a13006342edfeef32e813246b39[5]) for its crypto functions.

[5] https://csrc.nist.gov/CSRC/media/projects/cryptographic-module-validation-program/documents/security-policies/140sp3678.pdf

6.7 Trusted Path/Channel

FTP_DIT_EXT.1	All data transmitted between the Server and the web users and Engines (in the operational environment) are encrypted using TLSv1.2. For more details, please see the FCS_HTTPS* and FCS_TLS* requirements above.

7. Protection Profile Claims

This ST is conformant to the [APPSW].

8. Rationale

This Security Target includes by reference the [APPSW] Security Problem Definition, Security Objectives, and Security Assurance Requirements. The security target makes no additions to the [APPSW] assumptions. Security functional requirements have been reproduced verbatim with the protection profile operations completed. Operations on the security requirements follow [APPSW] application notes and assurance activities. The security target did not add or remove any security requirements. Consequently, [APPSW] rationale applies and is complete.