Palo Alto Networks Cortex XSOAR Module

FIPS 140-2 Non-Proprietary Security Policy

Revision Date: February 2, 2021
Documentation Version: 0.5

Palo Alto Networks
www.paloaltonetworks.com

Table of Contents

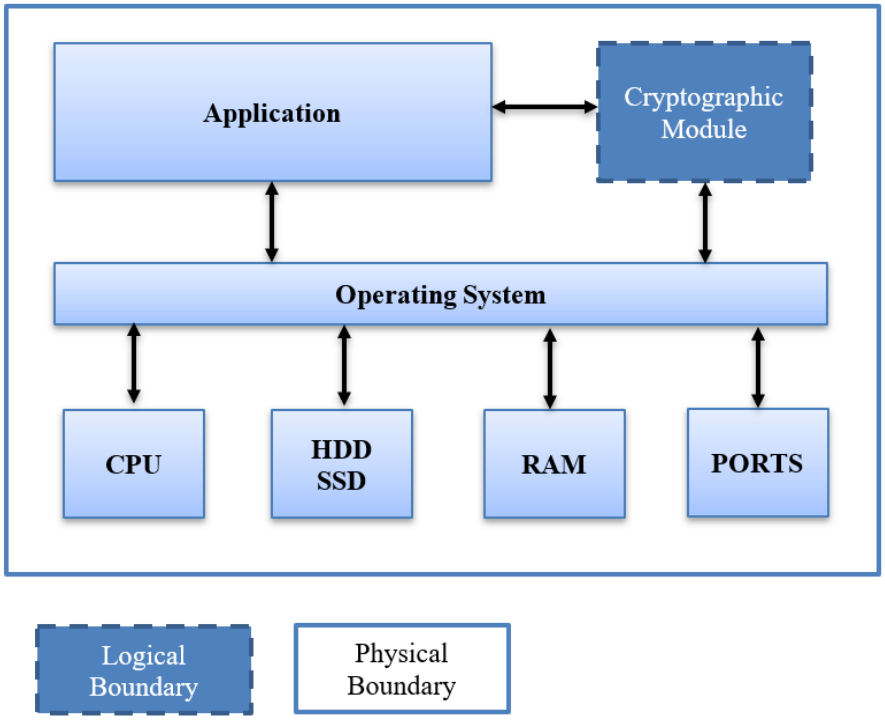
Module Overview

The Palo Alto Networks Cortex XSOAR[1] Module (hereafter referred to as the Module) is a software library providing a C-language Application Program Interface (API) for use by other applications that require cryptographic functionality. The Module is classified by FIPS 140-2 as a Level-1 software module, multi-chip standalone module embodiment. The logical cryptographic boundary of the Module is the BoringCrypto object module file named bcm.o. The physical cryptographic boundary is the General-Purpose Computer (GPC) on which the module is installed. The Module performs no communication other than with the calling application (the process that invokes the Module services). Additionally, the customer will compile the Module into both the Cortex XSOAR Client and Server executables, and while the FIPS lab will test it as an object module, the Security Policy and Validation will reflect that the Module is designed to be compiled into other executables, two examples of which are the Cortex XSOAR Client (i.e., Engine) and Server.

The Module's software version for this validation is 1.0.

Figure 1 demonstrates the logical boundary of the Module.

Figure 1 - Cryptographic Module Boundary



[1] XSOAR stands for eXtended Security Orchestration, Automation, and Response.

Security Levels

Table 1 - Module Security Level Specification

Security Requirements Section	Level
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles, Services, Authentication	1
Finite State Model	1
Physical Security	N/A
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	N/A

Modes of Operation

Approved Mode of Operation

The Module supports two modes of operation: Approved and Non-Approved. The Module will be in FIPS-Approved mode when all power-up self-tests have completed successfully and only Approved algorithms are invoked. The Non-Approved mode is entered when a non-Approved algorithm is invoked. See tables below for a list of Approved and non-Approved algorithms.

Approved and Allowed Algorithms

The Module supports the following algorithms FIPS approved algorithms.

Table 2 - FIPS Approved Algorithms

FIPS Approved Algorithm	CAVP Cert.
AES (e/d, mode: ECB, CBC, CTR, key sizes: 128, 192, 256) with and without PAA AES-GCM (e/d, key sizes: 128, 256) with and without PAA AES-KW (AE, AD, AES-128, AES-256, FWD, 128, 256, 320, 320, 320) with and without PAA	#C1750
KTS (AES Cert. #C1750; key establishment methodology provides 128 or 256 bits of encryption strength) with and without PAA	#C1750
Triple-DES (e/d, mode: ECB, CBC, key option: 1) with and without PAA	#C1750
ECDSA [FIPS 186-4] with and without PAA • PKG: CURVES (P-224 P-256 P-384 P-521) • PKV: CURVES (P-224 P-256 P-384 P-521) • SigGen: CURVES (P-224, P-256, P-384, P-521) with SHA-224/256/384/512 • SigVer: CURVES (P-224, P-256, P-384, P-521) with SHA-1/224/256/384/512	#C1750
RSA [FIPS 186-4] with and without PAA • Key Pair Generation: 2048 and 3072 • [RSASSA-PKCS1_V1_5] ○ SIG (gen) (2048/3072) with SHA-224/256/384/512 ○ SIG (Ver) (1024/2048/3072) with SHA-224/256/384/512. • [RSASSA-PSS] ○ SIG (gen) (2048/3072) with SHA-224/256/384/512 ○ SIG (Ver) (2048/3072) with SHA-224/256/384/512	#C1750
CKG (Vendor Affirmed) with and without PAA	N/A
DRBG [SP800-90A] with and without PAA	#C1750

- CTR based DRBG with AES-256 - No derivation function	
HMAC with SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 with and without PAA	#C1750
SHA-1, SHA-224, SHA-256, SHA-384, SHA-512 with and without PAA	#C1750

Notes:

- There are some algorithm modes that were tested but not implemented by the module. Only the algorithms, modes, and key sizes that are implemented by the module are shown in this table.

Non-FIPS Approved but FIPS Allowed Algorithms

The Module supports the following algorithms Non-FIPS but Approved algorithms that can be used in FIPS mode.

Table 3 –Allowed Algorithms

Algorithm	Notes
RSA (encrypt, decrypt)	The RSA algorithm may be used by the calling application for encryption or decryption of keys. No claim is made for SP 800-56B compliance, and no CSPs are established into or exported out of the Module using these services.
NDRNG	Non-Approved RNG. Used to seed FIPS Approved SP800-90A DRBG. The NDRNG is implemented by the underlying operating system (and not by the Module) which is outside its logical boundary.

Caveats:

- RSA (Key wrapping; key establishment methodology provides 112 or 128 bits of encryption strength)
- The module generates cryptographic keys whose strengths are modified by available entropy

Non-Approved Cryptographic Algorithms

The Module implements the following services which are Non-Approved per the FIPS 140-2 and SP 800-131Ar1 transition.

Table 4 –Non-Approved Algorithms

Algorithm	Notes
RSA	FIPS 186-2 (Non-compliant)
ECDSA	FIPS 186-2 (Non-compliant)

MD5	

Ports and Interfaces

The physical ports of the Module are the same as the GPC on which it is executing. The logical interface is a C-language API. The Module's physical ports and interfaces are described in the table below:

Table 5 - Module Ports and Interfaces

Interface	Logical Interface
Data Input	API Input Parameters
Data Output	API Output Parameters
Control Input	API Function Calls
Status Output	API Return Calls
Power Input	N/A

Roles, Services, and Authentication

The Module implements both authorized roles: The Crypto-Officer (CO) and User. The Module does not support user authentication. The CO and user roles are implicitly assumed by the entity accessing services implemented by the Module. No further authentication is required. The Module does not allow concurrent operators.

The Module does not provide a maintenance role or bypass capability.

Table 6 - Roles and Required Identification and Authentication

Role	Description
Crypto-Officer	Installation of the Module on the host GPC and calling of any API functions.
User	Loading the Module and calling any of the API functions.

All services implemented by the Module are listed below, along with a description of service CSP access. The access types are determined as follows:

- Generate (G): Generate the Critical Security Parameter (CSP) using an Approved Random Bit Generator (RBG)
- Read (R): Export the CSP (to an assigned location in memory)
- Write (W): Enter or establish, and store a CSP (to an assigned location in memory)
- Destroy (D): Overwrite the CSP
- Execute (E): Employ the CSP
- None: No access to the CSP

Table 7 - Services and CSPs Access

Service	Role	Description / CSP	Access Type
Installation	CO	Module installation. CSP: None	None
Initialize	User, CO	Module initialization. CSP: None	None
Self-test	User, CO	Perform self-tests (FIPS_selftest) including software integrity verification. CSP: None	None
Show Status	User, CO	Functions that provide the Module status information.	None

		CSP: None	
Zeroize	User, CO	Function that destroys all CSPs.	R/D
Random Number Generation	User, CO	Used for random number generation. CSPs: Entropy input string, DRBG seed, DRBG V and DRBG Key.	R/W
Asymmetric Key Generation	User, CO	Used to generate asymmetric keys. CSPs: RSA SGK, RSA SVK, ECDSA SGK, ECDSA SVK	G/R/W/E
Symmetric encrypt/decrypt	User, CO	Used to encrypt or decrypt data. CSPs: AES/TDES EDK, AES-GCM key, AES-KW key	R/E
Message Digest	User, CO	Used to generate a SHA-1 or SHA-2 message digest. CSP: None	None
Keyed Hash	User, CO	Used to generate or verify data integrity with HMAC. CSP: HMAC key	G/R/W/E
Key Wrapping	User, CO	Used to encrypt or decrypt a key value on behalf of the calling process (does not establish keys into the Module). CSPs: RSA KDK and RSA KEK	R/E
Digital Signature	User, CO	Used to generate or verify RSA or ECDSA digital signatures. CSPs: RSA SGK, RSA SVK, ECDSA SGK, ECDSA SVK	G/R/W/E
Utility	User, CO	Miscellaneous helper functions. CSP: None	None

Non-Approved Services

Table 8 – Non-Approved Services

Service	Role	Non-Approved Functions	Access Type
Hashing	User, CO	MD5	N/A
Signature Generation and Verification	User, CO	RSA (non-compliant) and ECDSA (non-compliant)	N/A

Please note that prior to using any of the Non-Approved/Compliant services listed in Table 8 above, the CO must zeroize all CSPs. Likewise, to put the module back into the FIPS mode from the non-FIPS mode, the CO must zeroize all Keys/CSPs used in non-FIPS mode, and then strictly follow up the instructions in section “Secure Operation” of this document to put the module into the FIPS mode.

Critical Security Parameters and Public Keys

All CSPs used by the Module are described in this section. All access to these CSPs by Module services are described in the previous table. The CSP names are generic, corresponding to the API parameter data structure.

Table 9 – CSPs and Public Keys

#	CSP/Key Name	Description
1	RSA SGK	RSA (2048/3072 bits) signature generation key
2	RSA KDK	RSA (2048/3072 bits) key decryption (private key transport) key
3	ECDSA SGK	FIPS 186-4 ECDSA (P-224/P-256/P-384/P-521 Curves) signature generation key
4	AES EDK	AES (128/192/256 bits) encrypt/decrypt key
5	AES GCM Key	AES (128/192/256 bits) encrypt/decrypt/generate/verify key
6	TDES EDK	TDES (3-Key) encrypt/decrypt key
7	HMAC Key	Keyed hash key (160/224/256/384/512 bits)
8	SP800-90A CTR_DRBG CSPs	V (128 bits), Seed (256/320/384 bits) and Key (AES 128/192/256 bits), Entropy input (384 bits from entropy source)
9	Key Wrap	AES Key Wrap using 128 or 256-bits keys
10	RSA SVK	RSA (1024/2048/3072 bits) signature verification public key
11	RSA KEK	RSA (2048/3072 bits) key encryption (public key Transport) key
12	ECDSA SVK	ECDSA (P-224/P-256/P-384/P-521 Curves) signature verification public key
13	Software Integrity Key	Software integrity test key using HMAC-SHA-512 key

For all CSPs and Public Keys

Storage: RAM, associated to entities by memory location. The Module stores DRBG state values for the lifetime of the DRBG instance. The Module uses CSPs passed in by the calling application on the stack or registers. The Module does not store any CSP persistently (beyond the lifetime of an API call), with the exception of DRBG state values used for the Modules' default key generation service.

Generation: In accordance with FIPS 140-2 IG D.12, the cryptographic module performs Cryptographic Key Generation as per as per section 6 in SP800-133. The resulting generated seed used in the asymmetric key generation is the unmodified output from SP800-90A DRBG. The calling application is responsible for storage of generated keys returned by the Module.

The module is a software module that contains an approved DRBG that is seeded exclusively from one known entropy source located within the operational environment inside the module's physical boundary but the outside the logical boundary, which is complaint with FIPS 140-2 IG 7.14 #1 (b). The minimum number of bits of entropy requested per each GET function call is at least 192 bits.

For operation in the Approved mode, the module users (the calling applications) shall use entropy sources that contain at least 192 bits of entropy. To ensure full DRBG strength, the entropy sources must meet or exceed the security strengths.

Entry: All CSPs enter the Module's logical boundary in plaintext as API parameters, associated by memory location. However, none cross the physical boundary.

Output: The Module does not output CSPs, other than as explicit results of key generation services. However, none cross the physical boundary.

Destruction: Zeroization of sensitive data is performed automatically by API function calls for temporarily stored CSPs. In addition, the Module provides functions to explicitly destroy CSPs related to random number generation services. The calling application is responsible for parameters passed in and out of the module.

Private and secret keys as well as seeds are provided to the Module by the calling application and are destroyed when released by the appropriate API function calls. Keys residing in internally allocated data structures (during the lifetime of an API call) can only be accessed using the Module defined API. The operating system protects memory and process space from unauthorized access. Only the calling application that creates or imports keys can use or export such keys. All API functions are executed by the invoking calling application in a non-overlapping sequence such that no two API functions will execute concurrently. An authorized application as user (Crypto Officer and User) has access to all key data generated during the operation of the Module.

Operational Environment

The Module will operate in a modifiable operational environment per the FIPS 140-2 definition. The operating system shall be restricted to a single operator mode of operation (i.e., concurrent operators are explicitly excluded). The external application that makes calls to the Module is the single user of the Module, even when the application is serving multiple clients.

Tested Configurations

The Module was tested in the following configurations.

Table 10 – Tested Configurations

Operating Environment	Processor	Platform
Red Hat 7.7 on VMware ESXi 6.5	Intel Xeon E5 with PAA Intel Xeon E5 without PAA	Dell R730 Server
CentOS 7.7 on VMware ESXi 6.5	Intel Xeon E5 with PAA Intel Xeon E5 without PAA	Dell R730 Server

Vendor Affirmed Configurations

The following platforms have not been tested as part of the FIPS 140-2 Level 1 certification however Palo Alto Networks "vendor affirms" that these platforms are equivalent to the tested and validated platforms. Additionally, Palo Alto Networks affirms that the Module will function the same way and provide the same security services on any of the operating systems listed below.

- RHEL 7
- CentOS 7
- Ubuntu 16.04
- Ubuntu 18.04
- Oracle Linux 7
- Amazon Linux 2

As per FIPS 140-2 Implementation Guidance G.5, compliance is maintained for other versions of the respective operational environments where the module binary is unchanged. No claim can be made as to the correct operation of the module or the security strengths of the generated keys if any source code is changed and the module binary is reconstructed.

The GPC(s) used during testing met Federal Communications Commission (FCC) FCC Electromagnetic Interference (EMI) and Electromagnetic Compatibility (EMC) requirements for business use as defined by 47 Code of Federal Regulations, Part 15, Subpart B. FIPS 140-2 validation compliance is maintained when the module is operated on other versions of the GPOS running in single user mode, assuming that the requirements outlined in NIST IG G.5 are met.

The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

Self-Tests

At start-up, Known Answer Tests (KATs) and software integrity check are performed. These tests are automatic and do not need operator intervention. If the value calculated and the known answer do not match, the Module immediately enters into FIPS_ERR state. Once the Module is in FIPS_ERR state, the Module becomes unusable via any interface.

The Module implements each of the following Power On Self-Tests (POSTs):

- AES-CBC encryption and decryption KATs
- AES-GCM encryption and decryption KATs
- Triple-DES-CBC encryption and decryption KATs
- RSA and ECDSA (sign/verify) KATs
- SP 800-90A CTR_DRBG KAT (Note: DRBG health tests as specified in SP800-90A Section 11.3 are performed)
- HMAC-SHA-512 KAT
- SHA KATs (SHA-1, SHA-256 and SHA-512)
- Software integrity check using HMAC-SHA-512.

Each module performs all power-on self-tests automatically when the module is initialized. All power-on self-tests must be passed before a User/Crypto Officer can perform services. The Power-on self-tests can be run on demand by power-cycling the host platform.

Conditional self-tests are run during operation of the module. If any of these tests fail, the module will enter an error state, where no services can be accessed by the operators. The module can be reinitialized to clear the error and resume FIPS mode of operation. Each module performs the following conditional self-tests.:

- Continuous Random Number Test to NDRNG per IG 9.8
- Pairwise Consistency Test for ECDSA
- Pairwise Consistency Test for RSA

Note that the RSA Pairwise consistency tests are performed for both possible nodes of use, e.g., Sign/Verify and Encrypt/Decrypt.

Physical Security

There are no physical security requirements as this is a software module.

Mitigation of Other Attacks

The Module is not designed to mitigate any specific attacks outside the scope of FIPS 140-2. These requirements are not applicable.

Secure Operation

Installation

Download the package from the link that you received from Cortex XSOAR Support. Run the **chmod +x** command to convert the .sh file to an executable file. As root user, execute the .sh file. Accept the EULA and complete the installation process. In a web browser, go to the **https://*serverURL*: *port*** to verify that Cortex XSOAR was successfully installed.

Initialization

The cryptographic module is initialized by loading the module before any cryptographic functionality is available. In User Space the operating system is responsible for the initialization process and loading of the library. The module is designed with a default entry point (DEP) which ensures that the power-up tests are initiated automatically when the module is loaded.

Use of AES-GCM

In approved mode, users of the module must not utilize GCM with an externally generated IV unless the source of the IV is also FIPS approved for GCM IV generation.

The module's implementation of AES-GCM is used together with an application that executes outside of the module's logical cryptographic boundary. The application negotiates the protocol session's keys and the value of the IV. The IV generation method will conform to the requirements specified in Provision 1 of IG A.5.

Per IG A.5, in the event module power is lost and restored the consuming application must ensure that any of its AES-GCM keys used for encryption or decryption are re-distributed.

Use of Triple-DES

In accordance with CMVP IG A.13, when operating in a FIPS approved mode of operation, the same Triple-DES key shall not be used to encrypt more than 2^{20} or 2^{16} 64-bit data blocks