Pensando TLS Library

by Pensando Systems, Inc.

Version 1.0

FIPS 140-2 Level 1 Non-Proprietary Security Policy

Document Version Number: 1.2
Date: February 27, 2024

Table of Contents

1. Module Overview 3
2. Modes of Operation 4
3. Ports and interfaces 7
4. Roles and Services 7
5. Cryptographic Keys and CSPs 8
6. Self-tests 10
7. References 10

1. Module Overview

Pensando TLS Library is a set of standard Transport Layer Security (TLS) functions that are written in the GO programming language. It supports TLS protocol version 1.2 (client and server) and standard cryptographic functions, such as SHA, AES, etc.

This GO TLS Library is used in all Pensando products to secure the management plane communications such as product provisioning, policy distribution, API orchestration, etc.

Table 1.1: Configuration tested by the lab

Module	Platform	Processor	Operating Systems
Pensando TLS Library	HPE:ProLiant DL360 Gen10	Intel Xeon Gold 6140 with and without AES-NI	CentOS v7.7 on VMware ESXi 6.7
Pensando TLS Library	Capri 1.0[1]	Capri 1.0[1]	Linux 4.14.18
Pensando TLS Library	Aruba CX 10000 Switch	Intel Xeon D-1637 with and without AES-NI	ArubaOS-CX version 10.12

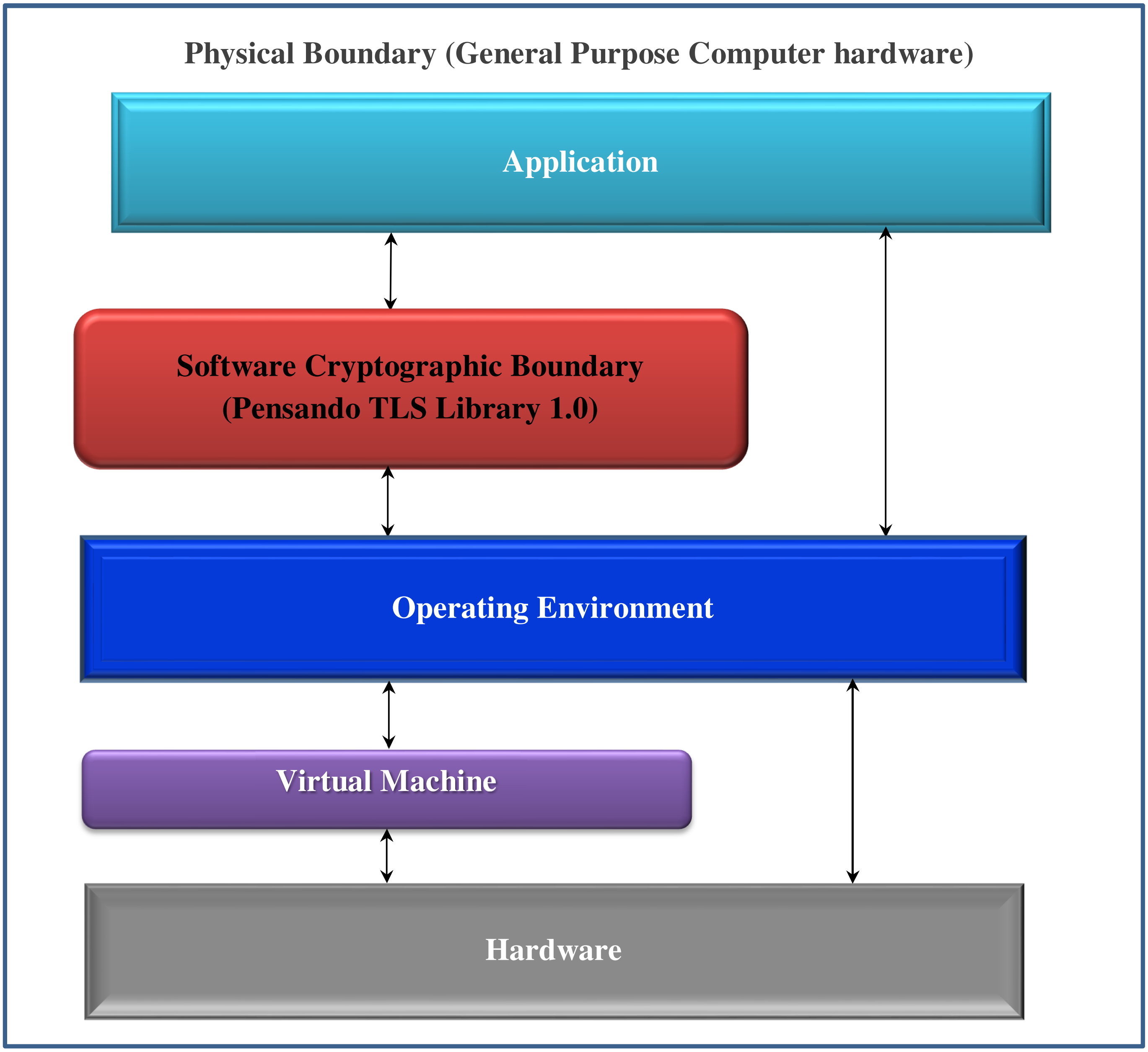
[1]Capri 1.0 is both the platform and the processor. The entire OS as well as the Pensando TLS Library run on it.

Table 1.2: Module Security Level Statement

FIPS Security Area	Security Level
Cryptographic Module Specification	1
Module Ports and Interfaces	1
Roles, Services and Authentication[1]	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

[1] This Level 1 module does not implement authentication.

Figure 1: Pensando TLS Library



2. Modes of Operation

The Pensando TLS Library supports the following two modes of operation to accommodate different operating requirements. The mode is selected implicitly based on the services used.

1) If an operator uses an approved function (Table 2.1), the module is in the FIPS mode.
2) If an operator uses a non-approved function (Table 2.2), the module is in a non-FIPS mode.

The CSPs shall not be shared between the approved and non-approved modes.

2.1 Approved and Allowed Cryptographic Functions

The following approved cryptographic algorithms are used in FIPS approved mode of operation.

Table 2.1: Approved Cryptographic Functions.

CAVP Cert	Library	Algorithm	Standard	Model/ Method	Key Lengths, Curves or Moduli	Use
A1289 A4801	Pensando TLS Library	KAS-ECC-SSC[1]	SP800-56Ar3	ECC Ephemeral Unified Scheme	P-256	TLS Shared Secret Computation
C2155	Pensando TLS Library	AES	FIPS 197, SP 800-38D	CBC, GCM[2]	128, 256	Encryption/ Decryption
				CTR		
C2156 A4801	Pensando TLS Library	AES	FIPS 197, SP 800-38D	CBC, GCM[2]	128, 256	Encryption/ Decryption
				CTR		
		ECDSA[3]	FIPS 186-4	ECDSA KeyGen	P-256, P-384	Key Generation, Key Verification, Signature Generation, Signature Verification
				ECDSA KeyVer		
				ECDSA SigGen	P-224, P-256, P-384, P-521	
				ECDSA SigVer		
		HMAC	FIPS198-1	HMAC-SHA-1 HMAC-SHA-256 HMAC-SHA-384	160, 256, 384	TLS Message Authentication Code
		HMAC DRBG	SP800-90A	SHA2-256		Deterministic Random Bit Generation
		KBKDF	SP800-108	HMAC-SHA-1, HMAC-SHA2-256, HMAC-		Key Derivation

CAVP Cert	Library	Algorithm	Standard	Model/ Method	Key Lengths, Curves or Moduli	Use
				SHA2-384		
		CVL KDF TLS	SP800-135			TLS Key Derivation[4]
		RSA	FIPS 186-4	RSA SigGen RSA SigVer PKCS 1.5 SHA-256, SHA-384, SHA-512	Mod 2048; Mod 3072	Signature Generation, Signature Verification
		SHS	FIPS 180-4	SHA-1, SHA-256, SHA-384, SHA-512		TLS Message Digest
CKG (vendor affirmed)			Cryptographic Key Generation			Key Generation[5]

Note 1: Not all CAVS-tested modes of the algorithms are used in this module.

[1]Key establishment methodology provides 128 bits of encryption strength.

[2]The module's AES-GCM implementation complies with IG A.5 scenario 1 and RFC 5288, and supports acceptable GCM cipher suites from Section 3.3.1 of SP 800-52 Rev 1 or SP 800-52 Rev 2. AES-GCM is only used in TLS version 1.2. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, that encounters this condition will trigger a handshake to establish a new encryption key. New AES-GCM keys are generated by the module if the module loses power.

[3]SHA-1 is only allowed and CAVS tested in ECDSA Signature Verification. It is not used for Signature Generation.

[4]No parts of this protocol, other than the KDF, has been tested by the CAVP and CMVP.

[5]CKG can be used to generate symmetric keys and asymmetric keys. The module directly uses the output of the DRBG. The generated symmetric key or a seed used in the asymmetric key generation is an unmodified output from DRBG. Section 4, example 1, of SP800-133r2 "Using the Output of a Random Bit Generator" is applicable.

Table 2.2: Non FIPS Approved Cryptographic Functions

Algorithm	Use
RC4	Encryption/Decryption
3DES-EDE (non-compliant)	Encryption/Decryption
CHACHA20	Encryption/Decryption
POLY1305	Message Authentication Code

Algorithm	Use
Ed25519	Digital Signature
SHA224 (non-compliant)	Hashing
SHA512/224 (non-compliant)	Hashing
SHA512/256 (non-compliant)	Hashing
RSA Key generation (non-compliant)	Digital Signature
RSA-PSS (non-compliant)	Digital Signature
Diffie-Hellman	Key Establishment
RSA Key Wrapping	Key Establishment

3. Ports and interfaces

The physical ports of the module are the same as those of the computer system on which it is executing. The logical interfaces of the module are implemented via an Application Programming Interface (API). The following table describes each logical interface.

Table 3: FIPS 140-2 Logical Interfaces

Logical Interface	Description
Data Input	Input parameters that are supplied to the API commands
Data Output	Output parameters that are returned by the API commands
Control Input	API commands
Status Output	Return status provided by API commands

4. Roles and Services

The module supports the following roles:

User role: The user uses the cryptographic services provided by the module.

Crypto Officer role: The Crypto Officer installs and manages the module.

Table 4: Roles and Services

Service	Corresponding Roles	Types of Access to Cryptographic Keys and CSPs R – Read or Execute W – Write or Create Z – Zeroize
Installation	Crypto Officer	N/A
Initialize	Crypto Officer	N/A
Self-test	Crypto Officer	N/A
Show status	Crypto Officer User	N/A
Zeroization	Crypto Officer	All:Z
Reboot or shutdown	Crypto Officer	N/A
Deterministic random number generation	User	DRBG CSPs: R, W
Hashing	User	N/A
Symmetric encryption and decryption using AES	User	AES key: R
Message authentication using HMAC	User	HMAC key: R
Digital signature creation and verification using ECDSA and RSA	User	RSA keys: R ECDSA keys: R
Key agreement using ECC DH	User	ECC DH keys: R, W
Symmetric and asymmetric key generation	User	DRBG CSPs: R,W
TLS Key derivation	User	TLS keys: R,W
SP800-108 Key derivation	User	AES key: R HMAC key: R

Non-Approved services are implementations of non FIPS Approved Cryptographic Functions. They are listed in the Table 2.2.

5. Cryptographic Keys and CSPs

The table below describes the cryptographic keys and CSPs used by the module.

Table 5: Cryptographic Keys and CSPs

Key	Description/Usage	Storage
AES Key Established using KDF TLS, KBKDF or DRBG	Used during AES encryption / decryption	RAM in plaintext
ECDSA public and private keys Established using DRBG	Used for Sign/Verify	RAM in plaintext
HMAC Key Established using KDF TLS, KBKDF or DRBG	Used during calculation of HMAC	RAM in plaintext
HMAC_DRBG CSPs: entropy input, V and Key Entropy is loaded externally	Used during generation of random numbers	RAM in plaintext
TLS master secret Established using KDF TLS	Used to derive TLS AES Key and TLS HMAC Key	RAM in plaintext
TLS pre-master secret Established using KAS-ECC-SSC	Used to derive TLS master secret	RAM in plaintext
RSA public and private keys Set by operators	Used for Sign/Verify	RAM in plaintext
Elliptic Curve Diffie Hellman public and private keys Established using DRBG	Diffie-Hellman key agreement	RAM in plaintext

Note-1: public keys are not considered CSPs

Note-2: All keys, that are generated by this module, are generated by using HMAC DRBG. Since the entropy is loaded externally, there is no assurance of the minimum strength of generated keys. The minimum length of the entropy field is 256 bits. Assuming that the entropy source provides full entropy, the module receives 256 bits of entropy.

Note-3: Keys can be provided to the module via API input parameters. The module does not enter or output keys outside its physical boundary. Zeroization is performed using power cycle. See Table 2.1 for size and strength of the keys.

6. Self-tests

The module performs the following power-up and conditional self-tests. Upon failure or a power-up or conditional self-test the module halts its operation.

Table 6: Self-Tests

Algorithm	Power-up Test
Software integrity	HMAC-SHA2-256
AES	KAT(CBC / GCM encryption/decryption are separately tested)
KAS (ECC-SSC)	Primitive "Z" Computation KAT per implementation guidance
ECDSA	Pairwise Consistency Test (curve sizes P-256) using SHA256
HMAC	KAT (HMAC-SHA-1)
KBKDF	KAT
DRBG	KAT
TLS 1.2 KDF	KAT
RSA	KAT (key size tested: 2048, using SHA-256)
SHA	KAT (SHA-256, SHA-512)
	Conditional Test
KAS (ECC-SSC)	ECC DH Private/Public Key Validation tests as per SP800-56Ar3 including ECC Full Public-Key Validation Routine
ECDSA	Pairwise Consistency Test
DRBG	Continuous Random Number Generator test
	DRBG health tests, performed per SP 800-90A Section 11.3

7. References

Table 7: References

Reference	Specification
[ANS X9.31]	Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (rDSA)
[FIPS 140-2]	Security Requirements for Cryptographic modules, May 25, 2001
[FIPS 180-4]	Secure Hash Standard (SHS)
[FIPS 186-2/4]	Digital Signature Standard
[FIPS 197]	Advanced Encryption Standard
[FIPS 198-1]	The Keyed-Hash Message Authentication Code (HMAC)
[FIPS 202]	SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions
[PKCS#1 v2.1]	RSA Cryptography Standard
[PKCS#5]	Password-Based Cryptography Standard
[PKCS#12]	Personal Information Exchange Syntax Standard
[SP 800-38A]	Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode
[SP 800-38B]	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
[SP 800-38C]	Recommendation for Block Cipher Modes of Operation: The CCM Mode for Authentication and Confidentiality
[SP 800-38D]	Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC
[SP 800-38F]	Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping
[SP 800-56A]	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography
[SP 800-56B]	Recommendation for Pair-Wise Key Establishment Schemes Using Integer Factorization Cryptography
[SP 800-56C]	Recommendation for Key Derivation through Extraction-then-Expansion
[SP 800-67R1]	Recommendation for the Triple Data Encryption Algorithm (TDEA) Block Cipher
[SP 800-89]	Recommendation for Obtaining Assurances for Digital Signature Applications
[SP 800-90A]	Recommendation for Random Number Generation Using Deterministic Random Bit Generators
[SP 800-108]	Recommendation for Key Derivation Using Pseudorandom Functions
[SP 800-132]	Recommendation for Password-Based Key Derivation
[SP 800-135]	Recommendation for Existing Application –Specific Key Derivation Functions