TippingPoint Crypto Core OpenSSL

FIPS 140-2 Security Policy

by Trend Micro Inc.

Version 1.0.2l-fips

Document Version: 1.2

May 20, 2021

Prepared by:

leidos

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

Modification History

Date	Modifications
03-02-2021	Version 1.0
04-12-2021	Version 1.1 – Additional Operational Environments
05-20-2021	Version 1.2 – Additional vendor affirmed Operational Environments

References

Reference	Full Specification Name
[ANS X9.31]	Digital Signatures Using Reversible Public Key Cryptography for the Financial Services Industry (RSA)
[FIPS 140-2]	Security Requirements for Cryptographic modules, May 25, 2001
[FIPS 180-4]	Secure Hash Standard
[FIPS 186-4]	Digital Signature Standard
[FIPS 197]	Advanced Encryption Standard
[FIPS 198-1]	The Keyed-Hash Message Authentication Code (HMAC)
[IG]	Implementation Guidance for FIPS 140-2 and the Cryptographic Module Validation Program
[MAN]	Manpages
[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 800-56A]	Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography
[SP 800-67]	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 800-131A]	Transitions: Recommendation for Transitioning the Use of Cryptographic Algorithms and Key Lengths
[SP 800-133]	Recommendation for Cryptographic Key Generation
[SP 800-135]	Recommendation for Existing Application-Specific Key Derivation Functions
[UG]	User Guide

Table of Contents

1. Introduction

This document is the non-proprietary security policy for the TippingPoint Crypto Core OpenSSL Version 1.0.2l-fips hereafter referred to as the Module.

The Module is a software library providing a C-language application program interface (API) for use by other processes that require cryptographic functionality. The Module is classified by FIPS 140-2 as a software module, multi-chip standalone module embodiment. The physical cryptographic boundary is the enclosure of the general purpose computer on which the module is installed. The logical cryptographic boundary of the Module is the shared library files and The Module performs no communications other than with the calling application (the process that invokes the Module services).

The FIPS 140-2 security levels for the Module are as follows:

Table 1: Security Level of Security Requirements

Security Requirement	Security Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services, and 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

1.1 Cryptographic Boundary

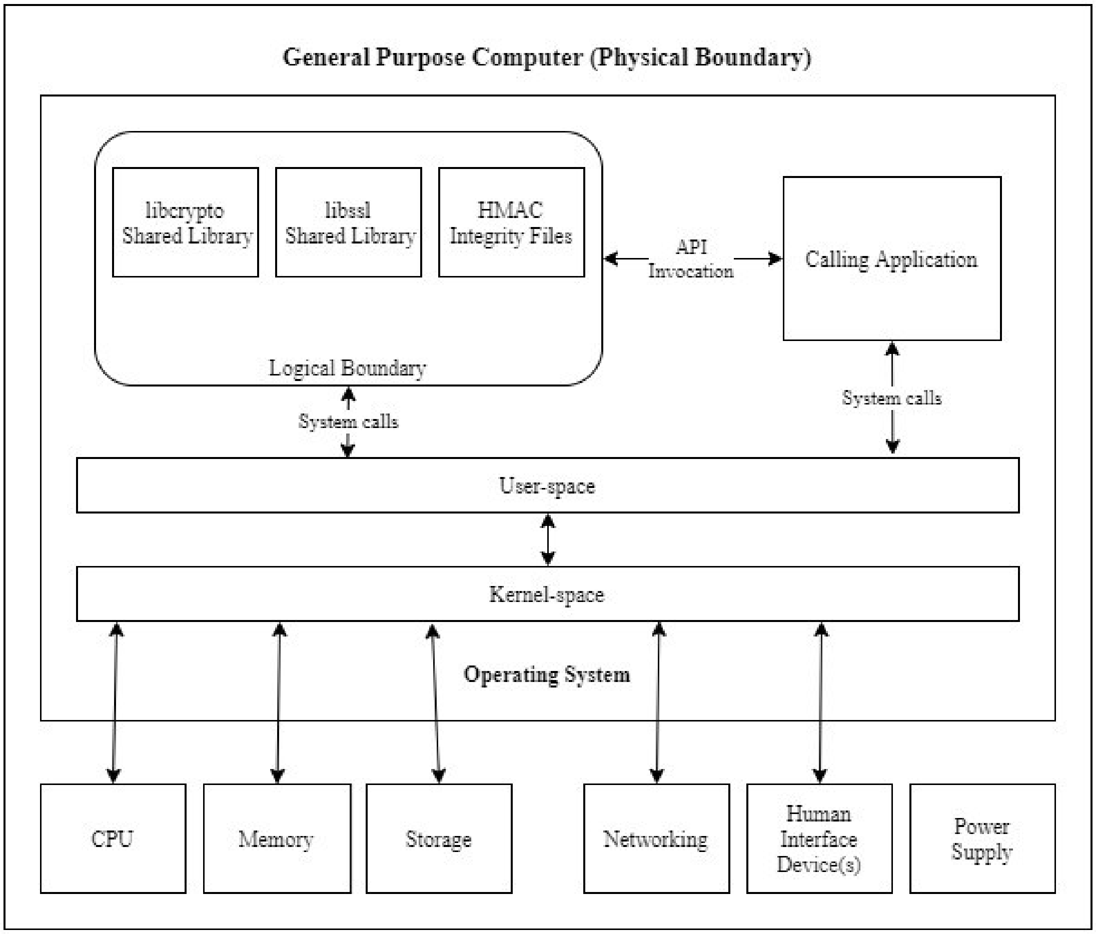
The module's physical boundary consists of the physical enclosure of the general purpose computer it is operating on. The module's logical cryptographic boundary consists of the following shared library binary files and their corresponding integrity files:

- /usr/lib/libcrypto.so.1.0.2

- /usr/lib/libcrypto.so.1.0.2.sha1
- /usr/lib/libssl.so.1.0.2
- /usr/lib/libssl.so.1.0.2.sha1

The module is delivered as part of Trend Micro’s TippingPoint Operating System (TOS).

Figure 1: Module Block Diagram



2. Tested Configurations

Table 2: Tested Configurations

#	Operational Environment	Processor	Optimizations (PAA)
1.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 2200T	Intel Xeon E5-2620	AES-NI
2.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 8200TX	Intel Xeon E5-2648L v3	AES-NI
3.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 8400TX	Intel Xeon E5-2648L v3	AES-NI
4.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with KVM 1.5.3 on Red Hat Enterprise Linux (RHEL) 7 running on an HP Proliant DL360 Gen7	Intel Xeon X5650	AES-NI
5.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 5.5 running on an HP ProLiant DL360 Gen8	Intel Xeon E5-2697 v2	AES-NI
6.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.0 running on an HP ProLiant DL360 Gen9	Intel Xeon E5-2698 v3	AES-NI
7.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.5 running on a Dell PowerEdge Server R640	Intel Xeon E5-2690	AES-NI
8.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.5 running on a Dell PowerEdge Server R640	Intel Xeon E5-2690	None
9.	Linux 4.4 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.7 running on a Dell PowerEdge Server R640	Intel Xeon E5-2683	AES-NI
10.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 440T	Intel Core i3-3220	None

#	Operational Environment	Processor	Optimizations (PAA)
11.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 5500TX	Intel Xeon D-1559	AES-NI
12.	Linux 4.4 running on a Trend Micro TippingPoint Threat Protection System 1100TX	Intel Pentium D1517	AES-NI

2.1 Vendor Affirmed Configurations

The module can execute in additional operational environments, each composed from a combination of the following hardware platforms and hypervisors. The CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when ported to an operational environment that is not listed on the validation certificate. As allowed by the FIPS 140-2 Implementation Guidance G.5, the validation status of the Cryptographic Module is maintained when operated in the following additional operating environments:

Any **Intel Xeon based** server hardware platforms supported by the below mentioned hypervisors. The following hypervisors are Vendor affirmed:

- KVM – Redhat Enterprise Linux
- VMWare ESXi

Any of the following platforms:

Table 3 Vendor Affirmed Operational Environments

#	Operational Environment	Processor	Optimizations (PAA)
1.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 2200T	Intel Xeon E5-2620	AES-NI
2.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 8200TX	Intel Xeon E5-2648L v3	AES-NI
3.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 8400TX	Intel Xeon E5-2648L v3	AES-NI

#	Operational Environment	Processor	Optimizations (PAA)
4.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with KVM 1.5.3 on Red Hat Enterprise Linux (RHEL) 7 running on an HP Proliant DL360 Gen7	Intel Xeon X5650	AES-NI
5.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 5.5 running on an HP ProLiant DL360 Gen8	Intel Xeon E5-2697 v2	AES-NI
6.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.0 running on an HP ProLiant DL360 Gen9	Intel Xeon E5-2698 v3	AES-NI
7.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.5 running on a Dell PowerEdge Server R640	Intel Xeon E5-2690	AES-NI
8.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.5 running on a Dell PowerEdge Server R640	Intel Xeon E5-2690	None
9.	Linux 4.14 on a Trend Micro TippingPoint Virtual Threat Protection System (vTPS) with VMware ESXi 6.7 running on a Dell PowerEdge Server R640	Intel Xeon E5-2683	AES-NI
10.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 440T	Intel Core i3-3220	None
11.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 5500TX	Intel Xeon D-1559	AES-NI
12.	Linux 4.14 running on a Trend Micro TippingPoint Threat Protection System 1100TX	Intel Pentium D1517	AES-NI

3. Ports and Interfaces

The physical ports of the Module are the same as the computer system on which it is executing. The logical interface is a C-language application program interface (API).

Table 4: Logical interfaces

Logical Interface Type	Description
Control input	API entry point and corresponding stack parameters
Data input	API entry point data input stack parameters
Status output	API entry point return values and status stack parameters
Data output	API entry point data output stack parameters

As a software module, control of the physical ports is outside module scope. However, when the module is performing self-tests, or is in an error state, all output on the logical data output interface is inhibited. The module is single-threaded and in error scenarios returns only an error value (no data output is returned). For specific details regarding the API components refer to [MAN] and [UG].

4. Modes of Operation and Cryptographic Functionality

Tables 4a and 4b list the Approved and Non-approved but Allowed algorithms, respectively. Despite additional algorithms/modes being tested by the CAVP, only those algorithms/modes listed below are utilized by the module.

Table 5a: FIPS Approved Cryptographic Algorithms

Function	Algorithm	Options	Cert #
Random Number Generation; Symmetric key generation	[SP 800-90A] DRBG[1] Prediction resistance supported for all variations	Hash DRBG and HMAC DRBG with SHA-1 and all SHA-2 sizes. No reseed CTR DRBG, with and without derivation function, with AES-128, AES-192, and AES-256	2159 C1262
	[SP 800-133] CKG	Unmodified output from the approved DRBG's can be used to generate symmetric keys or asymmetric seeds. The module gathers at least 256-bits of entropy before generating keys.	Vendor Affirmed
Encryption, Decryption and CMAC	[SP 800-67] Triple-DES	3-Key Triple-DES (192-bit) TECB, TCBC, TCFB, TOFB; CMAC generate and verify	2761 C1262
	[FIPS 197] AES	XTS Key Sizes: 128 and 256 ECB, CBC, OFB, CFB 1, CFB 8, CFB 128, CTR; CCM; GCM; CMAC (generate and verify) Key Lengths: 128, 192, and 256	5484 C1262
	[SP 800-38B] CMAC [SP 800-38C] CCM [SP 800-38D] GCM [SP 800-38E] XTS		

5 [1] For all DRBGs the "supported security strengths" is the highest supported security strength per [SP800-90A] and [SP800-57].

Function	Algorithm	Options	Cert #
Message Digests	[FIPS 180-4] SHS	SHA-1, SHA-2 (224, 256, 384, 512)	4401 C1262
Keyed Hash	[FIPS 198] HMAC	SHA-1, SHA-2 (224, 256, 384, 512)	3640 C1262
Digital Signature and Asymmetric Key Generation	[FIPS 186-2] RSA	SigVer9.31, SigVerPKCS1.5, SigVerPSS with all modulus lengths and all SHA sizes. **Note**: Users of this library should keep DRBG as the random function when using these RSA options.	2945 C1262
	[FIPS 186-4] RSA	KeyGen: 2048-bits SigGen9.31 2048/3072 with all SHA-2 sizes except SHA-224, SigGenPKCS1.5 and SigGenPSS 2048/3072 with all SHA-2 sizes. SigVer9.31 1024/2048/3072 with SHA-1 and all SHA-2 sizes except SHA-224, SigVerPKCS1.5 and SigVerPSS 1024/2048/3072 with SHA-1 and all SHA-2 sizes.	2945 C1262
	[FIPS 186-4] DSA	PQG Gen, Key Pair Gen, Sig Gen (2048/3072 with all SHA-2 sizes) PQG Ver, Sig Ver (1024/2048/3072 with all SHA sizes]	1411 C1262
Key Pair Generation, Public Key Validation, Signature Generation, and Signature Verification	[FIPS 186-4] ECDSA	PKG: All NIST defined B, K and P curves except sizes 163 and 192. PKV: All NIST defined B, K and P curves. SigGen: All NIST defined B, K and P curves except sizes 163 and 192, with all SHA-2 sizes. SigVer: All NIST defined B, K and P curves except size 163, with SHA-1 and all SHA-2 sizes.	1470 C1262
ECC CDH (CVL)	[SP 800-56A] (§5.7.1.2)	All NIST defined B, K and P curves except sizes 163 and 192.	1937 C1262
Key Derivation (CVL)	[SP 800-135] (§4.2.1 and §4.2.2)[2]	TLS 1.0/1.1 TLS 1.2 with SHA-256 or SHA-384	C1566 C1262
Key Transport	[IG] (D.9)	AES and HMAC, key establishment methodology provides between 128 and 256 bits of encryption strength. AES GCM, key establishment methodology provides between 128 and 256 bits of encryption strength. Triple-DES and HMAC, key establishment	AES Certs. #5484 and #C1262, HMAC Certs. #3640

[2] No parts of the TLS protocol, other than the KDF, have been tested by the CAVP and CMVP.

Function	Algorithm	Options	Cert #
		methodology provides 112 bits of encryption strength.	and #C1262, Triple-DES Certs. #2761 and #C1262

The Module supports only NIST defined curves for use with ECDSA and ECC CDH.

The module implements the following services which are Non-Approved but allowed:

Table 4b: Non-FIPS Approved But Allowed Cryptographic Functions

Category	Algorithm	Description
Key Agreement	Diffie-Hellman	Diffie-Hellman using 2048 or 3072 bit keys. Key establishment methodology provides 112 or 128 bits of encryption strength. Utilized with CVL #C1566 and #C1262 for the TLS 1.0/1.1 and 1.2 protocols.
Key Agreement	EC Diffie-Hellman	Diffie-Hellman scheme using elliptic curve, supporting all NIST defined B, K and P curves except sizes 163 and 192. Key establishment methodology provides between 112 and 256 bits of encryption strength. Utilized with CVL #C1566 and #C1262 for the TLS 1.0/1.1 and 1.2 protocols.
Key Encryption, Decryption	RSA	The RSA algorithm may be used by the calling application for encryption or decryption of keys. This includes using PKCS1-v1_5 padding. Key establishment methodology provides between 112 and 256 bits of encryption strength.
Message Digests	MD5	MD5 is only to be used as part of the TLS protocol.
Random Number Generation	NDRNG	Entropy data is required to seed the DRBG. The module generates a minimum of 256 bits of entropy before generating keys.

The Module implements the following services which are Non-Approved per the SP 800-131A transition:

Table 4c: FIPS-Non-Approved Cryptographic Functions

Function	Algorithm	Options
Random Number Generation; Symmetric key generation	[ANS X9.31] RNG	AES 128/192/256
Digital Signature and Asymmetric Key Generation	[FIPS 186-2] RSA	GenKey9.31, SigGen9.31, SigGenPKCS1.5, SigGenPSS
	[FIPS 186-2] DSA	PQG Gen, Key Pair Gen, Sig Gen (1024 with all SHA sizes, 2048/3072 with SHA-1)
	[FIPS 186-4] DSA	PQG Gen, Key Pair Gen, Sig Gen (1024 with all SHA sizes, 2048/3072 with SHA-1)
	[FIPS 186-2] ECDSA	PKG and Sig(Gen): All NIST defined B, K and P curves
	[FIPS 186-4] ECDSA	PKG: P-192, K-163, B-163 SigGen: P-192, K-163, B-163 with SHA-1 and

Function	Algorithm	Options
		all SHA-2 sizes. SigVer: K-163, B-163 with SHA-1 and all SHA-2 sizes.
Key Agreement	[RFC 2631] Diffie-Hellman	Key sizes less than 2048-bits.
ECC CDH (CVL)	[SP 800-56A] (§5.7.1.2)	All NIST Recommended B, K and P curves sizes 163 and 192
Encryption/Decryption	Camellia	Any
	CAST	Any
	DES	Any
	IDEA	Any
	RC2	Any
	RC4	Any
	RC5	Any
	RSA	Using any padding scheme other than PKCS1-v1_5
	Triple-DES	2-key
Message Digests	MD2	Any
	MD4	Any
	RIPEMD	Any
	Whirlpool	Any

These non-approved services shall not be used when operating in the FIPS Approved mode of operation.

The Module is a cryptographic engine library, which can be used only in conjunction with additional software. Aside from the Module use of the NIST defined elliptic curves as trusted third party domain parameters, all other FIPS 186-4 assurances are outside the scope of the Module, and are the responsibility of the calling process.

The module will automatically enable FIPS mode if the “fips-mode-enable” command has been invoked on the TippingPoint Operating System. Otherwise, the Module requires an initialization sequence (see IG 9.5): the calling application invokes FIPS_mode_set()[3], which returns a “1” for success and “0” for failure. If FIPS_mode_set() fails then all cryptographic services fail from then on.

[3] The function call in the Module is `FIPS_module_mode_set()` which is typically used by an application via the `FIPS_mode_set()` wrapper function

The calling application can use the ERR_get_error()function to query the reason for the failure.

EC Diffie-Hellman Key Agreement and RSA Key Wrapping provide between 112 and 256 bits of security strength when using key sizes provided in Table 2 of SP 800-57.

4.1 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 Section 4. The CSP names are generic, corresponding to API parameter data structures.

Table 4.1a: Critical Security Parameters

CSP Name	Generation	Input	Storage	Output	Zeroization	Description
RSA SGK	Internal/ External	API Call	RAM	API Call	API Call	RSA (2048 to 16384 bits) signature generation key
RSA KDK	Internal/ External	API Call	RAM	API Call	API Call	RSA (2048 to 16384 bits) key decryption (private key transport) key
DSA SGK	Internal/ External	API Call	RAM	API Call	API Call	[FIPS 186-4] DSA (2048/3072) signature generation key
ECDSA SGK	Internal/ External	API Call	RAM	API Call	API Call	ECDSA (All NIST defined B, K, and P curves except sizes 163 and 192) signature generation key
Diffie-Hellman Private	Internal/ External	API Call	RAM	API Call	API Call	Diffie-Hellman (Key sizes 2048-bits or 3072-bits)
EC Diffie-Hellman Private	Internal/ External	API Call	RAM	API Call	API Call	EC Diffie-Hellman (All NIST defined B, K, and P curves except sizes 163 and 192) private key agreement key
AES EDK	Internal/ External	API Call	RAM	API Call	API Call	AES (128/192/256) encrypt / decrypt key
AES CMAC	Internal/ External	API Call	RAM	API Call	API Call	AES (128/192/256) CMAC generate / verify key
AES GCM	Internal/ External	API Call	RAM	API Call	API Call	AES (128/192/256) encrypt / decrypt / generate / verify key
AES XTS	Internal/	API	RAM	API	API Call	AES (256/512) XTS encrypt /

CSP Name	Generation	Input	Storage	Output	Zeroization	Description
	External	Call		Call		decrypt key
Triple-DES EDK	Internal/ External	API Call	RAM	API Call	API Call	Triple-DES (3-Key) encrypt / decrypt key
Triple-DES CMAC	Internal/ External	API Call	RAM	API Call	API Call	Triple-DES (3-Key) CMAC generate / verify key
HMAC Key	Internal/ External	API Call	RAM	API Call	API Call	Keyed hash key (160/224/256/384/512)
Hash_DRBG CSPs	Internal	N/A	RAM	N/A	API Call	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
HMAC_DRBG CSPs	Internal	N/A	RAM	N/A	API Call	V (160/224/256/384/512 bits) and Key (160/224/256/384/512 bits), entropy input(length dependent on security strength)
CTR_DRBG CSPs	Internal	N/A	RAM	N/A	API Call	V (128 bits) and Key (AES 128/192/256), entropy input (length dependent on security strength)
TLS Pre-Master Secret	Internal/External	API Call	RAM	N/A	API Call	Size dependent on chosen key establishment method.
TLS Master Secret	Internal	N/A	RAM	N/A	API Call	384-bit value used to derive TLS session keys.

The module does not output intermediate key generation values.

Table 4.1b: Public Keys

CSP Name	Description
RSA SVK	RSA (1024 to 16384 bits) signature verification public key
RSA KEK	RSA (2048 to 16384 bits) key encryption (public key transport) key
DSA SVK	[FIPS 186-4] DSA (1024/2048/3072) signature verification key or [FIPS 186-2] DSA(1024) signature verification key
ECDSA SVK	ECDSA (All NIST defined B, K and P curves) signature verification key
Diffie-Hellman Public	Diffie-Hellman (Key sizes 2048-bits or greater)
EC Diffie-Hellman Public	EC Diffie-Hellman (All NIST defined B, K and P curves) public key agreement key

For all CSPs and Public Keys:

Generation: The Module implements SP 800-90A compliant DRBG services for creation of symmetric keys, and for generation of DSA, Elliptic Curve Cryptography, and RSA keys as shown in Table 4a. The calling application is responsible for storage of generated keys returned by the module. The SP 800-90A DRBG's are seeded by the NDRNG. The TLS Master Secret and session keys are derived via the SP 800-135 TLS KDFs.

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

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. 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.

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

Zeroization: Destruction 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.

4.2 Usage Rules

Private and secret keys as well as seeds and entropy input 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. Keys generated by the module in the non-approved mode shall not be used in the approved mode and vice versa.

AES GCM

In the event Module power is lost and restored the calling application must ensure that any AES-GCM keys used for encryption or decryption are re-distributed. The module complies with Scenario 1 of IG A.5. The Initialization Vector is generated as part of the TLS 1.2 (RFC 5246) protocol handshake and key derivation. AES-GCM can only be used in the context of TLS 1.2. When the nonce_explicit part of the IV exhausts the maximum number of possible values for a given session key, the party that encounters this condition must trigger a handshake to establish a new encryption key. The module supports the GCM ciphersuites from SP 800-52 Rev2.

AES XTS

AES-XTS shall only be used for protection of data on storage devices.

The length of the data unit for any instance of an implementation of XTS-AES shall not exceed 2^{20} AES blocks.

DRBG

When the DRBG entropy source is not specified the module generates at least 256 bits of entropy before generating keys. Calling applications shall not change the default entropy provider.

The API call of RAND_cleanup shall not be used.

Triple-DES

The user is responsible for ensuring that a single Triple-DES key shall not be used for more than 2^{16} 64-bit data block encryptions.

5. Roles, Authentication and Services

The Module implements the required User and Crypto Officer roles which are assumed implicitly.

Both roles have access to all of the services provided by the Module.

- User Role (User): Loading the Module and calling any of the API functions.
- Crypto Officer Role (CO): Initialization of the module.

All services implemented by the Module are listed below, along with a description of service CSP access.

Table 6: Services and CSP Access

Service	Role	Description
Initialize	CO	Module initialization. Does not access CSPs.
Self-test	User, CO	Perform self tests (FIPS_selftest). Does not access CSPs.
Show status	User, CO	Functions that provide module status information: Version (as unsigned long or const char *) FIPS Mode (Boolean) Does not access CSPs.
Zeroize	User, CO	Functions that destroy CSPs: fips_drbg_uninstantiate: for a given DRBG context, overwrites DRBG CSPs (Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs) All other services automatically overwrite CSPs stored in allocated memory. Stack cleanup is the responsibility of the calling application.
Random number generation	User, CO	Used for random number and symmetric key generation. Seed or reseed an DRBG instance Determine security strength of DRBG instance Obtain random data Uses and updates Hash_DRBG CSPs, HMAC_DRBG CSPs, CTR_DRBG CSPs.
Asymmetric key generation	User, CO	Used to generate DSA, ECDSA and RSA keys: RSA SGK, RSA SVK; DSA SGK, DSA SVK; ECDSA SGK, ECDSA SVK There is one supported entropy strength for each mechanism

Service	Role	Description
		and algorithm type, the maximum specified in SP800-90A
Symmetric encrypt/decrypt	User, CO	Used to encrypt or decrypt data. Executes using AES EDK, Triple-DES EDK (passed in by the calling process).
Symmetric digest	User, CO	Used to generate or verify data integrity with CMAC. Executes using AES CMAC, Triple-DES, CMAC (passed in by the calling process).
Message digest	User, CO	Used to generate a SHA-1 or SHA-2 message digest. Does not access CSPs.
Keyed Hash	User, CO	Used to generate or verify data integrity with HMAC. Executes using HMAC Key (passed in by the calling process).
Key transport[4]	User, CO	Used to encrypt or decrypt a key value on behalf of the calling process (does not establish keys into the module). Executes using RSA KDK, RSA KEK (passed in by the calling process).
Key agreement	User, CO	Used to perform key agreement primitives on behalf of the calling process (does not establish keys into the module). Executes using Diffie-Hellman Private, Diffie-Hellman Public (passed in by calling process, EC Diffie-Hellman Private, EC Diffie-Hellman Public (passed in by the calling process).
Digital signature	User, CO	Used to generate or verify RSA, DSA or ECDSA digital signatures. Executes using RSA SGK, RSA SVK; DSA SGK, DSA SVK; ECDSA SGK, ECDSA SVK (passed in by the calling process).
TLS protocol	User, CO	Used to protect data via a TLS session. Executes using AES EDK, AES GCM, Triple-DES EDK, HMAC Key.
TLS key agreement	User, CO	Used to establish a TLS protocol session. Executes using AES EDK, AES GCM, Triple-DES EDK, HMAC Key, TLS Pre-Master Secret, TLS Master Secret, RSA SGK, RSA KDK, DSA SGK, ECDSA SGK, EC Diffie-Hellman Private.
Utility	User, CO	Miscellaneous helper functions. Does not access CSPs

[4] "Key transport" can refer to a) moving keys in and out of the module or b) the use of keys by an external application. The latter definition is the one that applies to the TippingPoint Crypto Core OpenSSL.

6. Self-Test

The Module performs the self-tests listed below upon loading the module or upon invocation of Initialize or Self-test.
Table 6a: Power On Self Tests (KAT = Known answer test; PCT = Pairwise consistency test

Table 6a: Power On Self Tests (KAT = Known answer test; PCT = Pairwise consistency test)

Algorithm	Type	Test Attributes
Software integrity	KAT	HMAC-SHA1
HMAC	KAT	One KAT per SHA1, SHA224, SHA256, SHA384 and SHA512 Per IG 9.3, this testing covers SHA POST requirements.
AES	KAT	Separate encrypt and decrypt, ECB mode, 128 bit key length
AES CCM	KAT	Separate encrypt and decrypt, 192 key length
AES GCM	KAT	Separate encrypt and decrypt, 256 key length
XTS-AES	KAT	128, 256 bit key sizes to support either the 256-bit key size (for XTS-AES-128) or the 512-bit key size (for XTS-AES-256)
AES CMAC	KAT	CMAC generate and verify CBC mode, 128, 192, 256 key lengths
Triple-DES	KAT	Separate encrypt and decrypt, ECB mode, 3-Key
Triple-DES CMAC	KAT	CMAC generate and verify, CBC mode, 3-Key
RSA	KAT	Sign and verify using 2048 bit key, SHA-256, PKCS#1
DSA	PCT	Sign and verify using 2048 bit key, SHA-384
DRBG	KAT	CTR_DRBG: AES-256 with and without derivation function HASH_DRBG: SHA-1/224/256/384/512 HMAC_DRBG: SHA-1/224/256/384/512
ECDSA	PCT	KeyGen, sign, verify using P-224, K-233 and SHA512.
ECC CDH	KAT	Shared secret calculation per SP 800-56A §5.7.1.2, IG 9.6

The FIPS_mode_set()[5] function performs all power-up self-tests listed above automatically and with no operator intervention when the module is loaded. It returns a "1" if all power-up self-tests succeed, and a "0" otherwise. If any component of the power-up self-test fails an internal flag is set to prevent subsequent invocation of any cryptographic function calls. The module will only enter the FIPS Approved mode if the module is reloaded and the call to FIPS_mode_set() succeeds.

[5] `FIPS_mode_set()` calls Module function `FIPS_module_mode_set()`

The power-up self-tests may also be performed on-demand by calling FIPS_selftest(), which returns a "1" for success and "0" for failure. Interpretation of this return code is the responsibility of the calling application.

The Module also implements the following conditional tests:

Table 6b: Conditional Tests

Algorithm	Test
AES	XTS, Key_1 != Key_2
NDRNG	FIPS 140-2 continuous test for stuck fault on entropy source
DRBG	Tested as required by [SP800-90A] Section 11
DRBG	FIPS 140-2 continuous test for stuck fault
DSA	Pairwise consistency test on each generation of a key pair
ECDSA	Pairwise consistency test on each generation of a key pair
RSA	Pairwise consistency test on each generation of a key pair

In the event of a DRBG self-test failure the calling application must uninstantiate and re-instantiate the DRBG per the requirements of [SP 800-90A]; this is not something the Module can do itself.

Pairwise consistency tests are performed for both possible modes of use, e.g. Sign/Verify and Encrypt/Decrypt.

7. Operational Environment

The tested operating systems segregate user processes into separate process spaces. Each process space is logically separated from all other processes by the operating system software and hardware. The Module functions entirely within the process space of the calling application, and implicitly satisfies the FIPS 140-2 requirement for a single user mode of operation.

8. Mitigation of other Attacks

The module is not designed to mitigate against attacks which are outside of the scope of FIPS 140-2.