Pulse Secure Cryptographic Module

FIPS 140-2 Level 1 Non-Proprietary Security Policy
Version: 2.5
Last Updated: March 2017

Pulse Secure, LLC
2700 Zanker Road, Suite 200 San Jose, CA 95134 http://www.pulsesecure.net

Revision History

Authors	Date	Version	Comment
Pulse Secure, LLC	2017-Mar	2.5	Updated based on review comments
Pulse Secure, LLC	2017-Feb	2.4	Updated based on review
Pulse Secure, LLC	2016-Oct	2.3	Updated based on lab comments
Pulse Secure, LLC	2016-Aug-23	2.2	Add Pulse One to platform list
Pulse Secure, LLC	2016-Aug-2	2.1	Update per review comments
Pulse Secure, LLC	2016-05-25	2.0	186-4 RSA key generation support Update integrity tests to use HMAC-SHA256
Pulse Secure, LLC	2015-12-16	1.1	Deprecation of X9.31 RNG and Dual EC DRBG
Pulse Secure, LLC	2014-12-16	1.1	Changes for rebranding

Contents

Revision History 2
1 Introduction 4
2 Tested Configurations 6
3 Ports and Interfaces 7
4 Modes of Operation and Cryptographic Functionality 8
4.1 Critical Security Parameters and Public Keys 10
4.2 Special Notes on GCM 12
4.3 Special Notes on Entropy 12
5 Roles, Authentication and Services 14
6 Self-tests 16
7 Operational Environment 17
8 Mitigation of Other Attacks 17

1 Introduction

This document comprises the non-proprietary FIPS 140-2 Security Policy for the Pulse Secure Cryptographic Module, hereafter referred to as the Module.

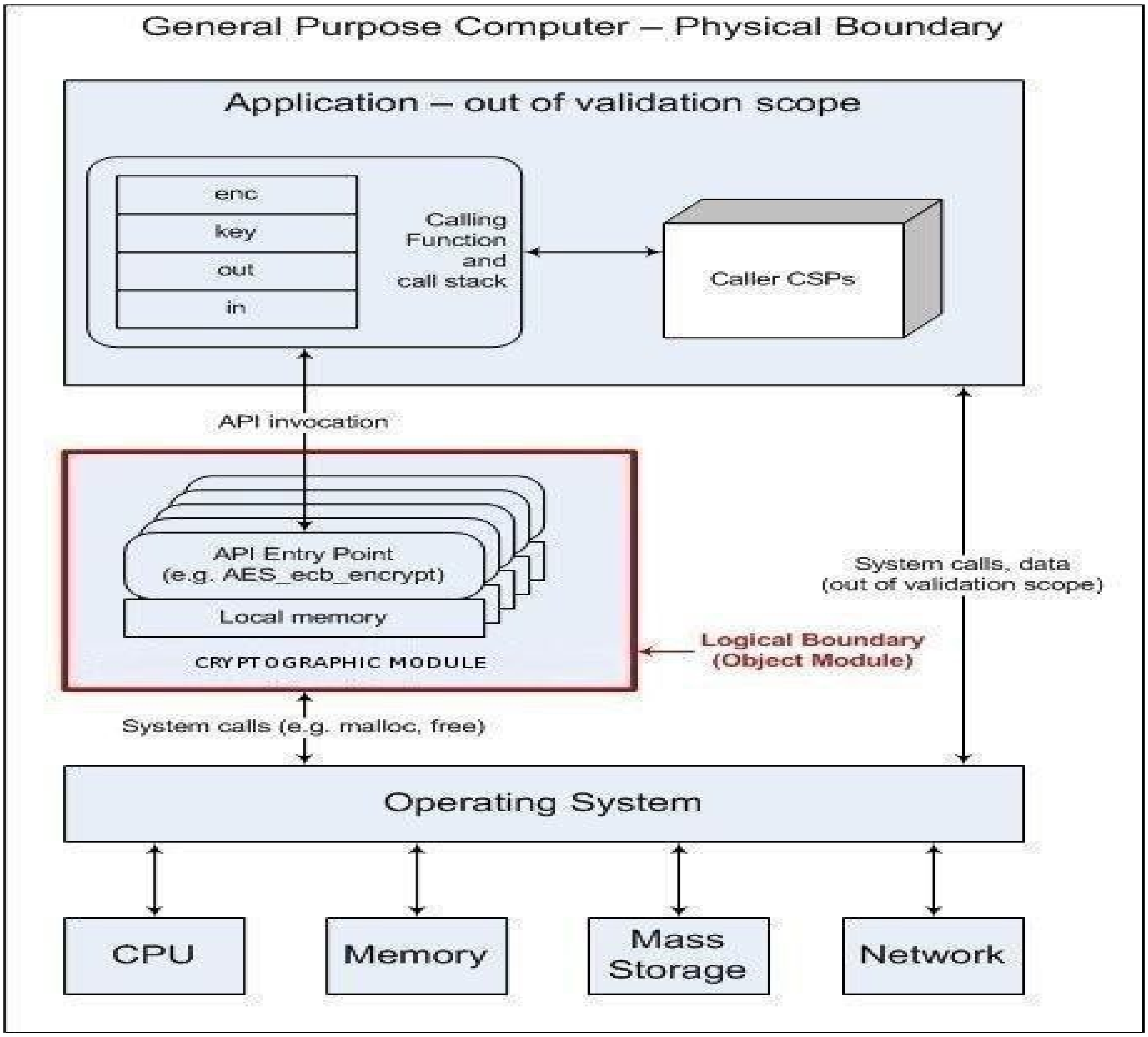
General Purpose Computer – Physical Boundary
Application – out of validation scope
enc
key
out
in
Calling Function and call stack
Caller CSPs
API invocation
API Entry Point (e.g. AES_ecb_encrypt)
Local memory
CRYPTOGRAPHIC MODULE
System calls, data (out of validation scope)
Logical Boundary (Object Module)
System calls (e.g. malloc, free)
Operating System
CPU
Memory
Mass Storage
Network

Figure 1 - Block Diagram

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 general purpose computer on which the Module is installed. The logical cryptographic boundary of the Module is the fipscanister object module, a single object

module file named *fipscanister.o*. The Module performs no communications other than with the calling application (the process that invokes the Module's services).
The FIPS 140-2 security levels for the Module are as follows:

Security Requirement	Security Level
Cryptographic Module Specification	1
Cryptographic Module Ports and Interfaces	1
Roles, Services, and Authentication	2
Finite State Model	1
Physical Security	NA
Operational Environment	1
Cryptographic Key Management	1
EMI/EMC	1
Self-Tests	1
Design Assurance	1
Mitigation of Other Attacks	NA

Table 1 - Security Level of Security Requirements

The Module's software version for this validation is 2.0.

2 Tested Configurations

	Operational Environment	Processor	Optimizations (Target)	Hardware Device
1.	IVE OS 2.0 32-bit	Intel Atom Processor N270 (x86)	None	Pulse Secure MAG2600
2.	IVE OS 2.0 64-bit	Intel Pentium Processor E2160 (x86)	None	Pulse Secure MAG 4610
3.	IVE OS 2.0 64-bit	Intel Pentium Processor E2160 (x86)	None	Pulse Secure MAG SM160
4.	IVE OS 2.0 64-bit	Intel Core2 Quad Q9400 (x86)	None	Pulse Secure MAG SM360
5.	IVE OS 2.0 64-bit	Intel Celeron Processor J1900 (x86)	None	Pulse Secure PSA300
6.	IVE OS 2.0 64-bit	Intel Celeron Processor J1900 (x86)	None	Pulse Secure PSA3000
7.	IVE OS 2.0 64-bit	Intel Pentium Processor G3420 (x86)	None	Pulse Secure PSA5000
8.	IVE OS 2.0 64-bit	Intel Xeon E3-1275v3 (x86)	AES-NI	Pulse Secure PSA 7000f
9.	IVE OS 2.0 64-bit	Intel Xeon E3-1275v3 (x86)	AES-NI	Pulse Secure PSA 7000c
10.	Pulse One version 2.0	Intel Xeon E3-1275v3 (x86)	AES-NI	Pulse Secure PSA 7000f
11.	Pulse One version 2.0	Intel Xeon E3-1275v3 (x86)	AES-NI	Pulse Secure PSA 7000c
12.	IVE OS 2.0 64-bit on VMware ESXi	Intel Xeon E5-2620 v4	None	Dell Power Edge R430/R530, Intel Xeon E5-2620 v4

Table 2 - Supported Platforms

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).

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

Table 3 - Logical interfaces

As a software module, control of the physical ports is outside the Module's 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).

4 Modes of Operation and Cryptographic Functionality

The Module supports Approved and non-approved mode selection of mode depends on which algorithms are chosen from Tables 4a and 4b. Tables 4a and 4b list the Approved and Non-approved but Allowed algorithms, respectively. Table 4c lists Non-approved algorithms not allowed in FIPS-mode of operation. If any algorithm listed in table 4c is chosen, the module is operating in non-FIPS mode of operation.

Function	Algorithm	Options	Cert. #
Encryption, Decryption and CMAC	[FIPS 197] AES [SP 800-38C] CCM [SP 800-38D] GCM	ECB; CBC; CCM; GCM 128/ 192/256	AES 4334
	[SP 800-38B] CMAC	Encryption, Decryption and CMAC generate and verify	AES 4341
ECC CDH Primitive	[SP 800-56A]	CURVES (P-224, P-256, P-384, P-521, K-233, K-283, K-409, B-233, B-283, B-409, B-571)	CVL 1046
Random Number Generation; Symmetric key generation	[SP 800-90Ar1] DRBG Prediction resistance supported for all variations	Hash DRBG (SHA-1, all SHA-2 sizes) HMAC DRBG (SHA-1, all SHA-2 sizes) CTR DRBG (AES 128/192/256)	DRBG 1384
Digital Signature and Asymmetric Key Generation Digital Signature and Asymmetric Key Generation Encryption, Decryption and CMAC	[FIPS 186-4] DSA (Legacy use only)	PQG Ver, Sig Ver (1024 with SHA-1 only)	DSA 1152
	[FIPS 186-4] DSA	PQG Ver, Sig Ver (2048/3072 with all SHA-2 sizes) PQG Gen, Key Pair Gen, Sig Gen (2048/3072 with all SHA-2 sizes)	
	[FIPS 186-4] ECDSA (Legacy use only)	SigVer: CURVES (P-192: (SHA-1, 224, 256, 384, 512) P-224: (SHA-1) P-256: (SHA-1) P-384: (SHA-1) P-521: (SHA-1) K-163: (SHA-1, 224, 256, 384, 512) K-233: (SHA-1, 224, 256, 384, 512) K-283: (SHA-1) K-409: (SHA-1) K-571: (SHA-1) B-163: (SHA-1) B-233: (SHA-1) B-283: (SHA-1) B-409: (SHA-1) B-571: (SHA-1))	ECDSA 1026
	[FIPS 186-4] ECDSA	PKG: CURVES (P-224 P-256 P-384 P-521 K-224 K-256 K-384 K-521 B-224 B-256 B-384 B-521 ExtraRandomBits TestingCandidates) PKV: CURVES (ALL-P ALL-K ALL-B) SigGen: CURVES P-224: (SHA-224, 256, 384, 512) P-256: (SHA-224, 256, 384, 512) P-384: (SHA-224, 256, 384, 512) P-521: (SHA-224, 256, 384, 512) K-233: (SHA-224, 256, 384, 512) K-283: (SHA-224, 256, 384, 512) K-409: (SHA-224, 256, 384, 512) K-571: (SHA-224, 256, 384, 512) B-233: (SHA-224, 256, 384, 512) B-283: (SHA-224, 256, 384, 512) B-409: (SHA-224, 256, 384, 512) B-571: (SHA-224, 256, 384, 512)) SigVer: CURVES (P-192: (SHA-1, 224, 256, 384, 512) P-224: (SHA-1, 224, 256, 384, 512) P-256: (SHA-1, 224, 256, 384, 512) P-384: (SHA-1, 224, 256, 384, 512) P-521: (SHA-1, 224, 256, 384, 512) K-163: (SHA-1, 224, 256, 384, 512) K-233: (SHA-1, 224, 256, 384, 512) K-283: (SHA-	

		1, 224, 256, 384, 512) K-409: (SHA-1, 224, 256, 384, 512) K-571: (SHA-1, 224, 256, 384, 512 B-163: (SHA-1, 224, 256, 384, 512) B-233: (SHA-1, 224, 256, 384, 512) B-283: (SHA-1, 224, 256, 384, 512) B-409: (SHA-1, 224, 256, 384, 512) B-571: (SHA-1, 224, 256, 384, 512))	
Keyed Hash	[FIPS 198] HMAC	HMAC with SHA-1, SHA-2 (224, 256, 384, 512)	HMAC 2880
Digital Signature and Asymmetric Key Generation	[FIPS 186-4] RSA (Legacy use only)	SigVer9.31, SigVerPKCS1.5, SigVerPSS (1024/4096 with all SHA sizes)	RSA 2345
	[FIPS 186-4] RSA	SigVer9.31, SigVerPKCS1.5, SigVerPSS (2048/3072 with all SHA-2 sizes) GenKey9.31, SigGen9.31, SigGenPKCS1.5, SigGenPSS (2048/3072with all SHA-2 sizes)	
Message Digests	[FIPS 180-4] SHA	SHA-1, SHA-2 (224, 256, 384, 512)	SHS 3577
Encryption, Decryption and CMAC	[SP 800-67] Triple-DES	TECB (Encrypt/Decrypt); TCBC (Encrypt/Decrypt)	Triple-DES 2346
	[SP 800-38B] CMAC	3-Key Triple-DES; CMAC generate and verify	Triple-DES 2347

Table 4a – FIPS Approved Cryptographic Functions

The Module supports only NIST defined curves for use with ECDSA and ECC CDH. Refer to the transition tables available at the CMVP Web site (http://csrc.nist.gov/groups/STM/cmvp/).

Category	Algorithm	Description
Key Agreement	EC Diffie-Hellman	Non-compliant (untested) Diffie-Hellman scheme using elliptic curve, supporting all NIST defined B, K and P curves. Key agreement is a service provided for calling process use, but is not used to establish keys into the Module.
Key Encryption, Decryption	RSA	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.
Digital Signature and Asymmetric Key Generation	RSA	RSA Key Generation, Signature Generation, and Signature Verification with all sizes between 2048 and 4096 (aside from the Approved and tested sizes of 2048 and 3072).

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

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

Function	Algorithm	Options
Random Number Generation; Symmetric key generation	[ANSI X9.31] RNG	AES 128/192/256
Symmetric Encryption	[SP800-38E AES-XTS]	AES 128/256 (Not compliant to IG A.9)

Function	Algorithm	Options
Digital Signature and Asymmetric Key Generation	[FIPS 186-4] DSA (Legacy use only)	PQG Ver, Sig Ver (1024 with SHA-1 only)
	[FIPS 186-4] DSA	PQG Ver, Sig Ver (2048/3072 with all SHA-2 sizes)
	[FIPS 186-4] ECDSA (Legacy use only)	PKG: CURVES(P-224 P-384 P-521 K-233 K- 283 K-409 K-571 B-233 B-283 B-409 B-571) PKV: CURVES(P-192 P-224 P-256 P-384 P-521 K-163 K-233 K-283 K-409 K-571 B-163 B-233 B-283 B-409 B-571)
	[FIPS 186-4] ECDSA	PKG: CURVES(P-192 K-163 B-163) SigGen: CURVES(P-192: (SHA-1, 224, 256, 384, 512) P-224:(SHA-1) P-256:(SHA-1) P-384:(SHA-1) P-521:(SHA-1) K-163: (SHA-1, 224, 256, 384, 512) K-233:(SHA-1) K-283:(SHA-1) K-409:(SHA-1) K-571:(SHA-1) B-163: (SHA-1, 224, 256, 384, 512) B- 233:(SHA-1) B-283:(SHA-1) B-409:(SHA-1) B-571:(SHA-1))
	[FIPS 186-4] RSA	GenKey9.31, SigGen9.31, SigGenPKCS1.5, SigGenPSS (1024/1536 with all SHA sizes, 2048/3072/4096 with SHA-1)
Key Agreement	EC Diffie-Hellman	All NIST Recommended B, K and P curves sizes 163 and 192
ECC CDH (CVL)	[SP 800-56A] (§5.7.1.2)	All NIST Recommended B, K and P curves sizes 163 and 192

Table 4c – FIPS Non-Approved Cryptographic Functions

The Module is in the Approved mode as long as none of the algorithms in Table 4c are used, and is in the Non-Approved mode when those algorithms are used.

When the Module is being loaded by the application, a set of power up self-tests are run before invoking the main application code using compile constructor, in the case of any power up self-test fail, the Module loading is aborted and application would exit. The Module requires an initialization sequence (see IG 9.5): the calling application invokes FIPS_mode_set(), which returns a "1" for success and "0" for failure. No services are performed during this sequence. Note that, while in FIPS mode, the module always performs the self-tests immediately upon initialization.

The Module is a cryptographic engine library, which can be used only in conjunction with additional software. Aside from the 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.

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.

CSP Name	Description
RSA SGK	RSA (2048 to 4096 bits) signature generation key

RSA KDK	RSA (2048 to 4096 bits) key decryption (private key transport) key
DSA SGK	[FIPS 186-4] DSA (2048/3072) signature generation key
ECDSA SGK	ECDSA (CURVES(P-224 P-256 P-384 P-521 K-233 K-283 K-409 K-571 B-233 B-283 B-409 B-571)) signature generation key
EC DH Private	EC DH (All NIST defined B, K, and P curves of size 224 or greater) private key agreement key.
AES EDK	AES (128/192/256) encrypt / decrypt key
AES CMAC	AES (128/192/256) CMAC generate / verify key
Triple-DES EDK	Triple-DES (3-Key) encrypt / decrypt key
Triple-DES CMAC	Triple-DES (3-Key) CMAC generate / verify key
HMAC Key	Keyed hash key (160/224/256/384/512)
Hash_DRBG CSPs	V (440/888 bits) and C (440/888 bits), entropy input (length dependent on security strength)
HMAC_DRBG CSPs	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	V (128 bits) and Key (AES 128/192/256), entropy input (length dependent on security strength)
CO-AD-Digest	Pre-calculated HMAC-SHA-256 digest used for Crypto Officer role authentication
User-AD-Digest	Pre-calculated HMAC-SHA-256 digest used for User role authentication

Table 4.1a – Critical Security Parameters

Authentication data is loaded into the Module during the Module build process, performed by an authorized operator (Crypto Officer), and otherwise cannot be accessed.

The Module does not output intermediate key generation values. The Module generates cryptographic keys whose strengths are modified by available entropy.

Public Key Name	Description
RSA SVK	RSA (2048 to 4096 bits) signature verification public key
RSA KEK	RSA (2048 to 4096 bits) key encryption (public key transport) key
DSA SVK	[FIPS 186-4] DSA (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
EC DH Public	EC DH (All NIST defined B, K and P curves) public key agreement key.

Table 4.1b – Public Keys

For all CSPs and Public Keys:

Storage: RAM, associated to entities by memory location. The Module stores RNG and DRBG state values for the lifetime of the RNG or 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 RNG and DRBG state values used for the Module's default key generation service.

Generation: The Module implements SP 800-90Ar1 compliant DRBG services for creation of symmetric keys, and for generation of DSA, elliptic curve, and RSA keys as shown in Table 4a. The calling application is responsible for storage of generated keys returned by the Module.

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 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.

4.2 Special Notes on GCM

Key and IV management are crucial for GCM. A single instance of IV reuse with any given key can expose the hash subkey. This Module's AES-GCM implementation meets the third option in IG A.5.

- The fixed field of the IV is variable in length, but is always at least 32 bits. (This allows for at least 2^{32} distinct module names.)
- There are three options for construction of the IV invocation field upon first invocation for a given key: user-supplied value, all zeros, or DRBG output. (While the DRBG output is random, the construction of subsequent IVs is deterministic.)
- In any of the above cases, the IV increments by 1 after each invocation. As there are 2^{64} possible invocation field values, wraparound will not occur in any reasonable timeframe.
- 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.

4.3 Special Notes on Entropy

This Module meets IG 7.14 option 2b. For operation in the Approved mode, Module users (the calling applications) shall use entropy sources containing at least 112 bits of entropy. To ensure full DRBG strength, the entropy sources must meet or exceed the security strengths shown in the table below.

DRBG Type	Underlying Algorithm	Minimum Seed Entropy
Hash_DRBG or HMAC_DRBG	SHA-1	128 bits
	SHA-224	192 bits
	SHA-256	256 bits
	SHA-384	256 bits
	SHA-512	256 bits
CTR_DRBG	AES-128	128 bits
	AES-192	192 bits
	AES-256	256 bits

Table 5 – DRBG Entropy Requirements

This entropy is supplied by means of callback functions. Those functions must return an error if the minimum entropy strength cannot be met.

5 Roles, Authentication and Services

The Module implements the required User and Crypto-Officer roles, as well as role-based authentication. Only one role may be active at a time and the Module does not allow concurrent operators. The User or Crypto Officer role is assumed by passing the appropriate password to the *FIPS_module_mode_set()* function. The password values may be specified at build time and must have a minimum length of 16 characters. Any attempt to authenticate with an invalid password will result in an immediate and permanent failure condition rendering the Module unable to enter the FIPS mode of operation, even with subsequent use of a correct password.

Authentication data is loaded into the Module during the Module build process, performed by the Crypto-Officer, and otherwise cannot be accessed.

The minimum password length is 16 characters, and each character can be any octet value other than the null terminator (0x00), leaving 255 possible values for each character. As such, the probability of a random successful authentication attempt in one try is a maximum of $1/255^{16}$, or less than $1/10^{37}$. The Module permanently disables further authentication attempts after a single failure, so this probability is independent of time.

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): Installation of the Module on the host computer system and calling of any API functions.

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

Service	Role	Description
Initialize	User, CO	Module initialization, inclusive of all Table 9 tests (FIPS_module_mode_set). Does not access CSPs.
Self-test	User, CO	Perform all Table 9 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	Function that destroys DRBG 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.

Service	Role	Description
Random number generation	User, CO	Used for random number and symmetric key generation. • Seed or reseed an RNG or DRBG instance • Determine security strength of an RNG or DRBG instance • Obtain random data Uses and updates RNG CSPs, 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
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 (algorithms only)	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 (algorithms only)	User, CO	Used to perform key agreement primitives on behalf of the calling process (does not establish keys into the module). Executes using EC DH Private, EC DH 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).
Utility	User, CO	Miscellaneous helper functions. Does not access CSPs.

Table 6 - Services and CSP Access

6 Self-tests

The Module performs the self-tests listed below on invocation of Initialize of self-test.

Algorithm	Type	Test Attributes
Software integrity	Integrity test	HMAC-SHA256 verification of software
AES	KAT	Separate encrypt and decrypt, ECB mode, 128 bit key length
AES CMAC	KAT	Sign and verify CBC mode, 128, 192, 256 key lengths
AES CCM	KAT	Separate encrypt and decrypt, 192 key length
AES GCM	KAT	Separate encrypt and decrypt, 256 key length
DSA	PCT	Sign and verify using 2048 bit key, SHA-384
DRBG	KAT	CTR_DRBG: AES, 256 bit with and without derivation function HASH_DRBG: SHA256 HMAC_DRBG: SHA256
ECDSA	PCT	Keygen, sign, verify using P-224, P-256, P-384, P-521, K-233 and SHA512. The K-233 self-test is not performed for operational environments that support prime curve only (see Table 2).
ECC CDH	KAT	Shared secret calculation per SP 800-56A §5.7.1.2, IG 9.6
HMAC	KAT	One KAT per SHA1, SHA224, SHA256, SHA384 and SHA512
RSA	KAT	Sign and verify using 2048 bit key, SHA-256, PKCS#1
Triple-DES	KAT	Separate encrypt and decrypt, ECB mode, 3-Key
Triple-DES CMAC	KAT	CMAC generate and verify, CBC mode, 3-Key

Table 7a - Power On Self Tests (KAT = Known answer test; PCT = Pairwise consistency test)

The *FIPS_mode_set()* function performs all power up self-tests listed above with no operator intervention required, returning 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.

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:

Algorithm	Test
DRBG	Tested as required by [SP800-90Ar1] 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
Entropy	Continuous test for stuck fault on DRBG inputs

Table 7b - Conditional Tests

In the event of a DRBG self-test failure, the Module enters the same error state as for a power up self-test failure.

Pairwise consistency tests are performed for both possible modes of use, e.g. Sign/Verify and Encrypt/Decrypt. When the test fails during pairwise consistency test function would return a failure and the calling application needs to handle the failure.

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.