Cisco Systems Inc

CiscoSSL FIPS Provider

FIPS 140-3 Non-Proprietary Security Policy

Americas Headquarters:
Cisco Systems, Inc., 170 West Tasman Drive, San Jose, CA 95134-1706 USA

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document is the non-proprietary Security Policy for the Cryptographic Module CiscoSSL FIPS Provider, firmware version 8.1. This Security Policy is provided in accordance with ISO/IEC 19790 Annex B, FIPS 140-3, and SP 800-140B. This Security Policy was prepared as part of the Level 1 FIPS 140-3 validation of the CiscoSSL FIPS provider, and the module meets the overall Level 1 requirements.

The following table lists the level of validation for each area in the FIPS PUB 140-3.

1.2 Security Levels

Section	Title	Security Level
1	General	1
2	Cryptographic module specification	1
3	Cryptographic module interfaces	1
4	Roles, services, and authentication	1
5	Software/Firmware security	1
6	Operational environment	1
7	Physical security	1
8	Non-invasive security	N/A
9	Sensitive security parameter management	1
10	Self-tests	1
11	Life-cycle assurance	1
12	Mitigation of other attacks	1
	Overall Level	1

Table 1: Security Levels

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The CiscoSSL FIPS Provider is a firmware library that provides cryptographic services to a vast array of Cisco's networking and collaboration products. The cryptographic module provides the cipher operations and Key Derivation functions to support the following protocols: IKEv2/IPSec, sRTP, SSH, TLS, SNMPv3, ANS X9.42, and ANS X.9.63. Full implementations of these protocols are not supported by the module. No parts of the protocols, other than the approved cryptographic algorithms and the KDFs, have been tested by the CAVP or CMVP. The tested module version is 8.1. The overall security level is 1.

The object code in the object module file is incorporated into the runtime executable application at the time the binary executable is generated. The module is provided in an executable form (as fips.so shared object). The module performs no communications other than with the consuming host application (the process that invokes the module services via the module's API), which can be considered as the host for the module.

Module Type: Firmware

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary:

The cryptographic boundary of the module is the CiscoSSL FIPS Provider, a dynamically loadable library. The module is comprised of a single object module file called fips.so. This CiscoSSL Provider is loaded into memory dynamically using the CiscoSSL provider API, effectively loading the necessary cryptographic functions and algorithms from the FIPS-

validated module into the application's memory space when needed. The module performs no communication other than with the calling application via APIs that invoke the module

Tested Operational Environment's Physical Perimeter (TOEPP):

The module's TOEPP is the physical perimeter of the tested platforms listed in Table "Tested Operational Environments - Software, Firmware, Hybrid" below. The components of the TOEPP include: Hardware components [Cisco UCS, Storage, RAM, Network Interface Cards].

The module's block diagram is shown in Figure 1 below. The dashed orange border in the figure denotes the cryptographic boundary of the module. The green border denotes the TOEPP of the module.

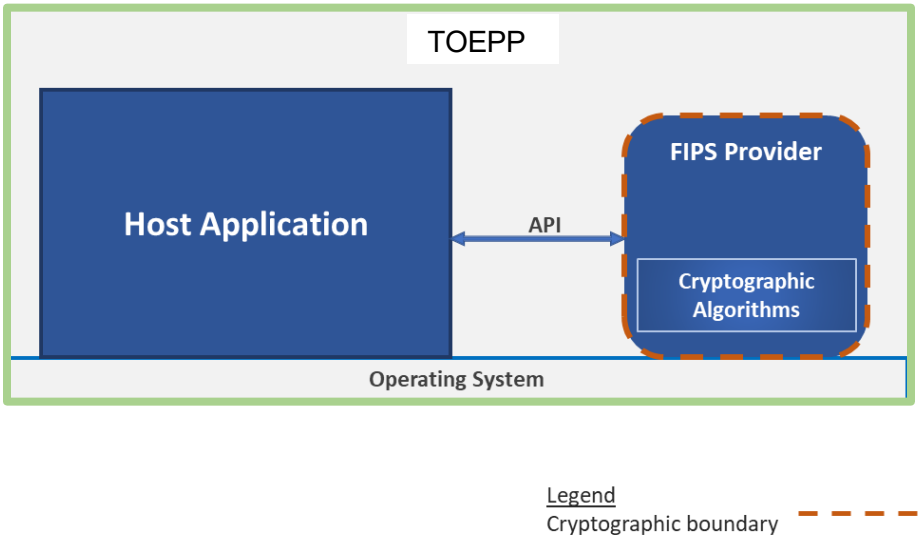

Figure 1: Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets):

Package or File Name	Software/ Firmware Version	Features	Integrity Test
Fips.so	8.1		HMAC SHA2-256

Table 2: Tested Module Identification – Software, Firmware, Hybrid (Executable Code Sets)

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Cisco IOS XE 17.15	Cisco UCS	Intel Xeon Gold 6244	Yes		8.1

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
Cisco IOS XE 17.15	Cisco UCS	Intel Xeon Gold 6244	No		8.1

Table 3: Tested Operational Environments - Software, Firmware, Hybrid

Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid:

N/A for this module.

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.

2.3 Excluded Components

There are no components excluded from the module.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved Mode	Provides services approved by FIPS 140-3	Approved	Returns 1 when approved services are run successfully
Non-Approved Mode	Provides services not approved for use in FIPS 140-3	Non-Approved	Returns 1 when DSA KeyGen or KRB5KDF non-approved services are run successfully

Table 4: Modes List and Description

The module supports both approved and non-approved modes of operation. The module will only enter the approved mode if the module is reloaded and the call to SELF_TEST_post() succeeds, and only approved services are invoked. The module enters non-approved mode when a non-approved service is invoked.

Mode Change Instructions and Status:

When a non-approved service is invoked while in approved mode of operation, the module implicitly transitions to a non-approved mode. Similarly, when a call to an approved service is made while in non-approved mode of operation, the module transitions to approved mode of operation.

2.5 Algorithms

Approved Algorithms:

PAA

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A5711	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A5711	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A5711	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A5711	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A5711	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A5711	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A5711	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A5711	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A5711	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
Counter DRBG	A5711	Prediction Resistance - Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - Yes	SP 800-90A Rev. 1
Deterministic ECDSA SigGen (FIPS186-5)	A5711	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5

Algorithm	CAVP Cert	Properties	Reference
DSA PQGVer (FIPS186-4)	A5711	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA SigVer (FIPS186-4)	A5711	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
ECDSA KeyGen (FIPS186-5)	A5711	Curve - P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A5711	Curve - P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A5711	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No, Yes	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A5711	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
Hash DRBG	A5711	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512, SHA3-256, SHA3-512	SP 800-90A Rev. 1
HMAC DRBG	A5711	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512, SHA3-256, SHA3-512	SP 800-90A Rev. 1
HMAC-SHA-1	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA3-256	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A5711	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
KAS-ECC CDH-Component SP800-56Ar3 (CVL)	A5711	Curve - P-256, P-384, P-521	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A5711	Domain Parameter Generation Methods - P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A5711	Domain Parameter Generation Methods - FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-IFC-SSC	A5711	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KAS1 - KAS Role - initiator, responder KAS2 - KAS Role - initiator, responder	SP 800-56A Rev. 3
KDA HKDF SP800-56Cr2	A5711	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A5711	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A5711	MAC Salting Methods - default, random KDF Mode - feedback Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDF ANS 9.42 (CVL)	A5711	KDF Type - DER Hash Algorithm - SHA-1, SHA2-224, SHA2-	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
		256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 8-4096 Increment 8	
KDF ANS 9.63 (CVL)	A5711	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF IKEv2 (CVL)	A5711	Diffie-Hellman Shared Secret Length - Diffie-Hellman Shared Secret Length: 2048 Derived Keying Material Length - Derived Keying Material Length: 3072 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF SNMP (CVL)	A5711	Password Length - Password Length: 256, 64	SP 800-135 Rev. 1
KDF SP800-108	A5711	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SRTP (CVL)	A5711	AES Key Length - 128, 192, 256	SP 800-135 Rev. 1
KDF SSH (CVL)	A5711	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KMAC-128	A5711	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KMAC-256	A5711	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KTS-IFC	A5711	Modulo - 2048, 3072, 4096, 6144 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
PBKDF	A5711	Iteration Count - Iteration Count: 1-10000 Increment 1 Password Length - Password Length: 8-128 Increment 8	SP 800-132

Algorithm	CAVP Cert	Properties	Reference
RSA Decryption Primitive Sp800-56Br2 (CVL)	A5711	Modulo - 2048, 3072, 4096	SP 800-56B Rev. 2
RSA KeyGen (FIPS186-5)	A5711	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A5711	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA Signature Primitive (CVL)	A5711	Modulo - 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-5)	A5711	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
Safe Primes Key Generation	A5711	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096	SP 800-56A Rev. 3
Safe Primes Key Verification	A5711	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096	SP 800-56A Rev. 3
SHA-1	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-224	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/256	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA3-224	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202

Algorithm	CAVP Cert	Properties	Reference
SHA3-384	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-512	A5711	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHAKE-128	A5711	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A5711	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TDES-CBC	A5711	Direction - Decrypt	SP 800-67 Rev. 2
TDES-CMAC	A5711	Direction - Verification	SP 800-67 Rev. 2
TDES-ECB	A5711	Direction - Decrypt	SP 800-67 Rev. 2
TLS v1.2 KDF RFC7627 (CVL)	A5711	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A5711	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 5: Approved Algorithms - PAA

Non-PAA

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A5712	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A5712	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A5712	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A5712	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A5712	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-GCM	A5712	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A5712	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A5712	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A5712	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
Counter DRBG	A5712	Prediction Resistance - Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - Yes	SP 800-90A Rev. 1
Deterministic ECDSA SigGen (FIPS186-5)	A5712	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No	FIPS 186-5
DSA PQGVer (FIPS186-4)	A5712	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA SigVer (FIPS186-4)	A5712	L - 1024, 2048, 3072 N - 160, 224, 256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
ECDSA KeyGen (FIPS186-5)	A5712	Curve - P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A5712	Curve - P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A5712	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No, Yes	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A5712	Curve - P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-	FIPS 186-5

Algorithm	CAVP Cert	Properties	Reference
		384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	
Hash DRBG	A5712	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512, SHA3-256, SHA3-512	SP 800-90A Rev. 1
HMAC DRBG	A5712	Prediction Resistance - Yes Mode - SHA-1, SHA2-256, SHA2-512, SHA3-256, SHA3-512	SP 800-90A Rev. 1
HMAC-SHA-1	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A5712	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
KAS-ECC CDH-Component SP800-56Ar3 (CVL)	A5712	Curve - P-256, P-384, P-521	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A5712	Domain Parameter Generation Methods - P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A5712	Domain Parameter Generation Methods - FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
KAS-IFC-SSC	A5712	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KAS1 - KAS Role - initiator, responder KAS2 - KAS Role - initiator, responder	SP 800-56A Rev. 3
KDA HKDF SP800-56Cr2	A5712	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8 HMAC Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512	SP 800-56C Rev. 2
KDA OneStep SP800-56Cr2	A5712	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A5712	MAC Salting Methods - default, random KDF Mode - feedback Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDF ANS 9.42 (CVL)	A5712	KDF Type - DER Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 8-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.63 (CVL)	A5712	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF IKEv2 (CVL)	A5712	Diffie-Hellman Shared Secret Length - Diffie-Hellman Shared Secret Length: 2048 Derived Keying Material Length - Derived Keying Material Length: 3072 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KDF SNMP (CVL)	A5712	Password Length - Password Length: 256, 64	SP 800-135 Rev. 1
KDF SP800-108	A5712	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SRTP (CVL)	A5712	AES Key Length - 128, 192, 256	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
KDF SSH (CVL)	A5712	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
KMAC-128	A5712	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KMAC-256	A5712	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KTS-IFC	A5712	Modulo - 2048, 3072, 4096, 6144 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	SP 800-56B Rev. 2
PBKDF	A5712	Iteration Count - Iteration Count: 1-10000 Increment 1 Password Length - Password Length: 8-128 Increment 8	SP 800-132
RSA Decryption Primitive Sp800-56Br2 (CVL)	A5712	Modulo - 2048, 3072, 4096	SP 800-56B Rev. 2
RSA KeyGen (FIPS186-5)	A5712	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2pow100 Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A5712	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA Signature Primitive (CVL)	A5712	Modulo - 2048, 3072, 4096	FIPS 186-4
RSA SigVer (FIPS186-5)	A5712	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
Safe Primes Key Generation	A5712	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096	SP 800-56A Rev. 3
Safe Primes Key Verification	A5712	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, modp-2048, modp-3072, modp-4096	SP 800-56A Rev. 3
SHA-1	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA2-224	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/256	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA3-224	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-384	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-512	A5712	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHAKE-128	A5712	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A5712	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TDES-CBC	A5712	Direction - Decrypt	SP 800-67 Rev. 2
TDES-CMAC	A5712	Direction - Verification	SP 800-67 Rev. 2
TDES-ECB	A5712	Direction - Decrypt	SP 800-67 Rev. 2
TLS v1.2 KDF RFC7627 (CVL)	A5712	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A5712	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 6: Approved Algorithms - Non-PAA

The module implements the cryptographic algorithms listed in the above tables. The module also supports RSA KeyGen, SigGen and SigVer with modulus size greater than 4096, where CAVP testing is not available.

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
CKG Section 4	Key Type:Asymmetric	N/A	SP 800-133 rev2 Section 4, example 1
CKG Section 4 Non-PAA	Key Type:Asymmetric	N/A	SP 800-133 rev2 Section 4, example 1

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

N/A for this module.

There are no non-approved and allowed algorithms, hence the table is excluded.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

There are no algorithms that are non-approved but allowed with no security claimed, hence the table is excluded.

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
DSA KeyGen	Asymmetric Key Generation (FIPS 186-4)
KRB5KDF	Kerberos RFC3961 section 5.1 KDF

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Random Number Generation	DRBG	Used for random number and symmetric key generation		Counter DRBG: (A5711, A5712) Hash DRBG: (A5711, A5712) HMAC DRBG:

Name	Type	Description	Properties	Algorithms
				(A5711, A5712)
Asymmetric Key Generation	AsymKeyPair-KeyGen	Used to generate ECDSA, RSA, DH, ECDH, keys		ECDSA KeyGen (FIPS186-5): (A5711, A5712) RSA KeyGen (FIPS186-5): (A5711, A5712) Safe Primes Key Generation: (A5711, A5712) CKG Section 4: () Key Type: Asymmetric CKG Section 4 Non-PAA: () Key Type: Asymmetric
Key Derivation Function (KDF)	KAS-135KDF KAS-56CKDF KBKDF PBKDF	Used to derive keys using KBKDF, PBKDF2, HKDF, SP 800-56C rev2, One-Step KDF (KDA), Two-Step KDF (KDA), SP 800-135 rev1 TLS 1.2, SSHv2, SNMPv3, SRTP, IKEv2, ANSI X9.63- 2001, ANSI X9.42-2001 KDFs and TLS 1.3 KDF		KDA HKDF SP800-56Cr2: (A5711, A5712) KDF ANS 9.42: (A5711, A5712) KDF ANS 9.63: (A5711, A5712) KDF IKEv2: (A5711, A5712) KDF SNMP: (A5711, A5712) KDF SP800-108: (A5711, A5712) KDF SRTP: (A5711, A5712)

Name	Type	Description	Properties	Algorithms
				KDF SSH: (A5711, A5712) PBKDF: (A5711, A5712) TLS v1.2 KDF RFC7627: (A5711, A5712) TLS v1.3 KDF: (A5711, A5712) KDA OneStep SP800-56Cr2: (A5711, A5712) KDA TwoStep SP800-56Cr2: (A5711, A5712)
Symmetric Encrypt/Decrypt	BC-Auth BC-UnAuth	Used to encrypt or decrypt data. Executes using AES EDK (passed in by the calling application)		AES-CBC: (A5711, A5712) AES-CBC-CS1: (A5711, A5712) AES-CBC-CS2: (A5711, A5712) AES-CBC-CS3: (A5711, A5712) AES-CCM: (A5711, A5712) AES-CFB1: (A5711, A5712) AES-CFB128: (A5711, A5712)

Name	Type	Description	Properties	Algorithms
				AES-CFB8: (A5711, A5712) AES-CTR: (A5711, A5712) AES-ECB: (A5711, A5712) AES-GCM: (A5711, A5712) AES-OFB: (A5711, A5712) AES-XTS Testing Revision 2.0: (A5711, A5712)
Message Digest (SHS)	SHA	Used to generate a SHA-1, SHA-2, or SHA-3 message digest		SHA-1: (A5711, A5712) SHA2-224: (A5711, A5712) SHA2-256: (A5711, A5712) SHA2-384: (A5711, A5712) SHA2-512: (A5711, A5712) SHA2-512/224: (A5711, A5712) SHA2-512/256: (A5711, A5712) SHA3-224: (A5711, A5712) SHA3-256: (A5711,

Name	Type	Description	Properties	Algorithms
				A5712) SHA3-384: (A5711, A5712) SHA3-512: (A5711, A5712) SHAKE-128: (A5711, A5712) SHAKE-256: (A5711, A5712)
Keyed Hash (HMAC/KMAC/CMAC/GMAC)	MAC	Used to generate or verify data integrity with HMAC, KMAC or CMAC. Executes using HMAC , KMAC, or AES Key (passed in by the calling application)		HMAC-SHA-1: (A5711, A5712) HMAC-SHA2-224: (A5711, A5712) HMAC-SHA2-256: (A5711, A5712) HMAC-SHA2-384: (A5711, A5712) HMAC-SHA2-512: (A5711, A5712) HMAC-SHA2-512/224: (A5711, A5712) HMAC-SHA2-512/256: (A5711, A5712) HMAC-SHA3-224: (A5711, A5712) HMAC-SHA3-256:

Name	Type	Description	Properties	Algorithms
				(A5711, A5712) HMAC-SHA3-384: (A5711, A5712) HMAC-SHA3-512: (A5711, A5712) KMAC-128: (A5711, A5712) KMAC-256: (A5711, A5712) AES-CMAC: (A5711, A5712) AES-GMAC: (A5711, A5712)
Key Wrapping (KW)	BC-Auth	Used to encrypt a key value on behalf of the calling application. Executes using AES Key Wrapping Key (passed in by the calling application). Key sizes 128, 192 and 256 providing 128, 192 and 256 bits of encryption strength. AES- KW, AES-KWP is CAVP tested per FIPS		AES-KW: (A5711, A5712) AES-KWP: (A5711, A5712)

Name	Type	Description	Properties	Algorithms
		140-3 IG D.G.		
Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)	KAS-SSC	Used to perform key agreement primitives on behalf of the calling application (does not establish keys into the module). Executes using DH Private, DH Public, EC DH Private, EC DH Public, RSA SGK, RSA SVK (passed in by the calling application). For ECC: Curves P-256, P-384 and P- 521 providing 128 to 256 bits of encryption strength. For FFC: 2048, 3072 and 4096 bit keys providing 112 to 152 bits of security strength. For IFC: 2048, 3072, 4096, 6144, 8192 bit		KAS-ECC CDH-Component SP800-56Ar3: (A5711, A5712) KAS-ECC-SSC Sp800-56Ar3: (A5711, A5712) KAS-FFC-SSC Sp800-56Ar3: (A5711, A5712) KAS-IFC-SSC: (A5711, A5712)

Name	Type	Description	Properties	Algorithms
		modulus providing 112 to 200 bits of encryption strength. The module follows SP 800-56A rev3 KAS-ECC-SSC (FIPS 140-3 IG D.F Scenario 2 path 1), SP 800-56A rev3 KAS-FFC-SSC (FIPS 140-3 IG D.F Scenario 2 path 1) and SP 800-56B rev2 KAS-IFC-SSC (FIPS 140-3 IG D.F Scenario 1 path 1)		
Digital Signature	DigSig-SigGen DigSig-SigVer	Used to generate or verify RSA, or ECDSA, digital signatures. Executes using RSA SGK, RSA SVK; ECDSA SGK, ECDSA SVK, (passed in by the calling application)		ECDSA SigGen (FIPS186-5): (A5711, A5712) ECDSA SigVer (FIPS186-5): (A5711, A5712) RSA SigGen (FIPS186-5): (A5711, A5712) RSA SigVer (FIPS186-5): (A5711, A5712) RSA

Name	Type	Description	Properties	Algorithms
				Signature Primitive: (A5711, A5712) Deterministic ECDSA SigGen (FIPS186-5): (A5711, A5712)
Asymmetric Key Verification	AsymKeyPair-KeyVer	Used to verify ECDSA public key and SafePrime keys		ECDSA KeyVer (FIPS186-5): (A5711, A5712) Safe Primes Key Verification: (A5711, A5712)
Key Transport	BC-AuthEncrypt	Used to support key transport, supports KTS- OAEP. 2048, 3072, 4096 and 6144 bit modulus providing 112 to 176 bits of encryption strength. The module follows SP 800-56B rev2 KTS-IFC (FIPS 140-3 IG D.G).		KTS-IFC: (A5711, A5712) RSA Decryption Primitive Sp800-56Br2: (A5711, A5712)
Legacy Symmetric Decrypt	BC-UnAuth	Used for TDES Decrypt Only.		TDES-CBC: (A5711, A5712) TDES-ECB: (A5711, A5712)

Name	Type	Description	Properties	Algorithms
Legacy Keyed Hash CMAC	MAC	Used for TDES Verify only.		TDES-CMAC: (A5711, A5712)
Legacy Digital Signature	DigSig-SigVer	Used to verify DSA digital signatures.		DSA SigVer (FIPS186-4): (A5711, A5712) DSA PQGVer (FIPS186-4): (A5711, A5712)

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

AES GCM IV Generation

In the case of AES-GCM, the IV generation method is user-selectable, and the value can be computed in more than one manner as follows:

1) TLS 1.2: The module's AES-GCM implementation conforms to IG C.H, scenario #1, following RFC 5288. The module is compatible with TLS 1.2 protocol and provides the primitives to support the AES GCM cipher suites from SP 800-52 rev1 Section 3.3.1. The counter portion of the IV is set by the module within its cryptographic boundary. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition will trigger a handshake to establish a new encryption key in accordance with RFC 5246 for TLS 1.2, respectively.

2) IKEv2: The module's AES-GCM implementation conforms to IG C.H, scenario #1 following RFC 7296 for IPSec/IKEv2. The AES GCM IV is generated according to RFC5282. The counter portion of the IV is set by the module within its cryptographic boundary. When the IV exhausts the maximum number of possible values for a given session key, the first party, client or server, to encounter this condition will trigger a handshake to establish a new encryption key. In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption shall be established.

3) TLS 1.3: The module's AES-GCM implementation conforms to IG C.H Scenario#5. The module is compatible with TLS v1.3 and provides support for the acceptable GCM cipher suites from Section 8.4 of RFC 8446 and confirms that the IV is generated and used within the protocol's implementation. The counter portion of the IV is set by the module within its cryptographic boundary. In case the module's power is lost and then restored, a new key for use with the AES GCM encryption/decryption must be established.

4) Non-protocol specific usage: The module's AES-GCM implementation conforms to IG C.H, scenario #3, when operating in approved mode of operation, AES GCM, IVs are

generated both internally and deterministically and are a minimum of 96-bits in length as specified in SP 800-38D, Section 8.2.1. The selection of the IV construction method is the responsibility of the user of this cryptographic module.

Note: Externally generated IVs are not allowed for AES-GCM encryption.

PBKDF

In line with the requirements of SP 800-132 and FIPS 140-3 IG D.N, keys generated using the approved PBKDF must only be used for storage applications. The algorithm uses option 1a as specified in SP 800-132 Section 5.4. Any other use of the approved PBKDF is non-conformant. In approved mode the module enforces that any password used must encode to at least 14 bytes (112 bits) and that the salt is at least 16 bytes (128 bits) long. The iteration count associated with the PBKDF should be as large as practical.

As the module is a general-purpose firmware module, it is not possible to anticipate all the levels of use for the PBKDF, however a user of the module should also note that a password should at least contain enough entropy to be unguessable and contain enough entropy to reflect the security strength required for the key being generated.

AES-XTS

In line with the requirements of SP 800-38E and FIPS 140-3 IG C.I, the keys are generated independently according to Section 6.3 of SP 800-133 rev2 and verification of the keys (key1 ≠ key2) is performed before using them in the AES-XTS algorithm.

Key Agreement

The module implements the following CAVP tested key agreement methods:
SP 800-56A rev3 KAS-ECC-SSC (FIPS 140-3 IG D.F Scenario 2 path 1)
SP 800-56A rev3 KAS-FFC-SSC (FIPS 140-3 IG D.F Scenario 2 path 1)
SP 800-56B rev2 KAS-IFC-SSC (FIPS 140-3 IG D.F Scenario 1 path 1)

Key Transport

The module does not establish SSPs using an approved key transport scheme (KTS). However, it does offer approved authenticated algorithms that can be used by an external operator/application as part of an approved KTS.

SHA-1

SHA-1 is approved for non-digital-signature applications (general hashing, HMAC, DRBG), disallowed for digital signature generation, and is Legacy Use for digital signature verification.
The module does not support use of SHA-1 for digital signature generation.
The module does not support the use of SHA-1 for verification of RSA or ECDSA signatures.
The module does support legacy use of SHA-1 for verification of DSA signatures.

SHA3 and SHAKE

Per FIPS 140-3 IG C.C, all SHA3 and SHAKE functions are tested on all the operational

environments. The higher-level algorithms using SHA3 (HMAC-SHA3) are also tested on all the operational environments.

RSA

Per FIPS 140-3 IG C.F, RSA SigGen is tested with 2048, 3072, 4096-bit modulus and RSA SigVer is tested with 2048, 3072, 4096-bit modulus. The module also supports RSA KeyGen, SigGen and SigVer with modulus size greater than 4096, for which CAVP testing is not available.

Legacy use algorithms

Per SP 800-131A rev2, TDES-CBC/TDES-ECB Decrypt, TDES-CMAC Verification, DSA SigVer using SHA-1 is allowed for legacy use.

Counter DRBG

CTR_DRBG was CAVP tested both with and without the derivation function; but module design ensures that only the version with the derivation function is usable to callers of the module's API.

Algorithm Component (CVL)

- KAS-ECC CDH Component - the KAS-ECC CDH-Component (CVL) shall only be used within the context of an SP 800-56Arev3 KAS.
- RSA SigGen Component - the RSA SigGen (CVL) shall only be used within the context of a FIPS 186-5 signature generation.
- ECDSA SigGen Component - the ECDSA SigGen (CVL) shall only be used within the context of a FIPS 186-5 signature generation.
- RSA Decryption Component - the RSA Decryption Primitive (CVL) shall only be used within the context of a SP 800-56Brev2 KTS.
- Key Derivation Functions - The key derivation functions (CVL) from the following protocols and standards documented in SP 800-135rev1, shall only be used in the context of their respective protocols.
- TLS 1.3 Key Derivation Function - The TLS 1.3 key derivation function (CVL) documented in Section 7.1 of RFC 8446 shall only be used in the context of the TLS 1.3 protocol.

2.8 RBG and Entropy

The module passively receives entropy from outside the boundary. The caveat "No assurance of the minimum strength of generated SSPs (e.g., keys)" applies to this module.

Applications shall use entropy sources that meet the security strength required for the random

number generation mechanism as shown in [SP 800-90A rev1] Table 2 (Hash_DRBG, HMAC_DRBG, CTR_DRBG). A minimum of 112-bits of entropy must be supplied and CTR_DRBG enforces the use of DF. This entropy is supplied by means of callback functions. Those functions must return an error if the minimum entropy strength cannot be met.

2.9 Key Generation

The module generates symmetric and asymmetric keys following the sections of SP 800-133 rev2 as specified in Table “Vendor-Affirmed Algorithms” above.

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 application space from unauthorized access. Only the calling application that creates or imports keys can use or export such keys. All API functions (Module Services) are executed by the calling application invoking an API. Each API either succeeds or fails and is logically non-interruptible from the point of view of the calling application.

The module supports generation of ECDSA, RSA, EC Diffie-Hellman and Diffie-Hellman key pairs per Section 5 in SP 800-133 rev2. The output of SP 800-90A rev1 random bit generator is used for generating the seed used in asymmetric key generation. The module also complies with Sections 6.1 and 6.2 of SP 800-133 rev2.

2.10 Key Establishment

The module implements key agreement methods per FIPS 140-3 IG D.F and key transport methods per FIPS 140-3 IG D.G (SP 800-38F AES-KW and AES-KWP, SP 800-56B rev2 KTS-IFC). Detailed information is provided in Table “Security Function Implementations” Section above.

2.11 Industry Protocols

In reference to FIPS 140-3 IG D.C, the module implements the KDFs of SSH, TLS, IKE, SRTP, SNMP, ANS X9.42 and ANS X9.63 but no parts of the protocols other than the approved cryptographic algorithms and the KDFs have been tested by the CAVP and CMVP.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API entry point data input stack parameters
N/A	Data Output	API output parameters resulting from call execution
N/A	Control Input	API entry point and corresponding stack parameters

Physical Port	Logical Interface(s)	Data That Passes
N/A	Status Output	API return value resulting from call execution

Table 10: Ports and Interfaces

The logical interface is a C-language application program interface (API). The Data Input interface consists of the input parameters of the API functions. The Data Output interface consists of the output parameters of the API functions. The Control Input interface consists of the actual API functions. The Status Output interface includes the return values of the API functions.

3.2 Control Interface Not Inhibited

Please note that the module does not support a control output interface and is not applicable for this module.

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module does not implement authentication mechanisms and does not allow concurrent operators.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto-Officer	Role	Crypto-Officer	None
User	Role	User	None

Table 11: Roles

The module meets all FIPS 140-3 level 1 requirements for Roles. The Module implements both a User Role (User) as well as the Crypto Officer (CO) role. The User and Crypto Officer roles are implicitly assumed by the application accessing services implemented by the Module.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Random Number Generation	Used for random number and symmetric key	1	DRBG struct (RBG State); DRBG_Seed	Status return; Random value	Random Number Generation	Crypto-Officer - Entropy Input: W,E,Z - DRBG_S

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	generation					eed: G,E,Z - DRBG_C: W,E - DRBG_Key: W,E - DRBG_V: W,E User - Entropy Input: W,E,Z - DRBG_Seed: G,E,Z - DRBG_C: W,E - DRBG_Key: W,E - DRBG_V: W,E
Asymmetric Key Generation	Generate asymmetric key pairs	1	ECDSA: curve identifier. RSA: domain parameter targets. DH: keypair information. EC DH: curve identifier.	Status return; general private and public keys	Asymmetric Key Generation	Crypto-Officer - RSA SGK: G,R - RSA SVK: G,R - ECDSA SGK: G,R - ECDSA SVK: G,R - RSA KEK: G,R - RSA KDK: G,R - DH Private: G,R

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- DH Public: G,R - EC DH Private: G,R - EC DH Public: G,R User - RSA SGK: G,R - RSA SVK: G,R - ECDSA SGK: G,R - ECDSA SVK: G,R - RSA KEK: G,R - RSA KDK: G,R - DH Private: G,R - DH Public: G,R - EC DH Private: G,R - EC DH Public: G,R
Key Derivation Function (KDF)	Used to derive keys using KBKDF, PBKDF2, HKDF, SP 800-56C rev2 One-	1	Key agreement shared secret; flags	Status return; derived keying material	Key Derivation Function (KDF)	Crypto-Officer - KDF Derived Key: G,R - Shared Secret: W,E User - KDF Derived

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	Step KDF (KDA), SP 800-56C rev2 Two-Step KDF (KDA), SP 800-135 rev1 TLS 1.2, SSHv2, SNMPv3, SRTP, IKEv2, ANSI X9.63-2001, ANSI X9.42-2001 KDFs and TLS 1.3 KDF					Key: G,R - Shared Secret: W,E
Symmetric Encrypt/Decrypt	Used to encrypt or decrypt data. Executes using AES EDK, GMC, or XTS key, or TDES DK [decrypt only] (passed in by the calling application)	1	Encryption or decryption key; plaintext or ciphertext data; flags	Status return. Plaintext or ciphertext data	Symmetric Encrypt/Decrypt Legacy Symmetric Decrypt	Crypto-Officer - AES EDK: W,E - AES GCM: W,E - AES XTS: W,E - TDES DK: W,E User - AES EDK: W,E - AES GCM: W,E - AES XTS: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- TDES DK: W,E
Message Digest (SHS)	Used to generate a SHA-1, SHA-2, or SHA-3 message digest	1	Data to be hashed	Status return. Hashed data	Message Digest (SHS)	Crypto-Officer User
Keyed Hash	Used to generate or verify data integrity with HMAC, KMAC, CMAC or GMAC. Executes using HMAC, KMAC or AES Key (passed in by the calling application)	1	Data to be hashed and key	Status return; MAC output value.	Keyed Hash (HMAC/KMAC/CMAC/GMAC) Legacy Keyed Hash CMAC	Crypto-Officer - HMAC Key: W,E - KMAC Key: W,E - AES CMAC: W,E - AES GCM: W,E - TDES CMAC: W,E User - HMAC Key: W,E - KMAC Key: W,E - AES CMAC: W,E - AES GCM: W,E - TDES CMAC: W,E
Key Wrapping Encrypt (KW)	Used to encrypt a key value on behalf of the calling applicati	1	Keying material and key	Encrypted key	Key Wrapping (KW)	Crypto-Officer - AES Key Wrapping: W,E - Keying Material:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	on. Executes using AES Key Wrapping Key (passed in by the calling application). AES-KW, AES-KWP is CAVP tested per FIPS 140-3 IG D.G.					W - AES-KW Encrypted Key: G,R User - AES Key Wrapping: W,E - Keying Material: W - AES-KW Encrypted Key: G,R
Key Wrapping Decrypt (KW)	Used to decrypt a key value on behalf of the calling application. Executes using AES Key Wrapping Key (passed in by the calling application). AES-KW, AES-KWP is CAVP tested per FIPS	1	Encrypted Key and key	Keying material	Key Wrapping (KW)	Crypto-Officer - AES Key Wrapping: W,E - AES-KW Encrypted Key: W - Keying Material: G,R User - AES Key Wrapping: W,E - AES-KW Encrypted Key: W - Keying Material: G,R

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	140-3 IG D.G.					
Key Agreement/ Agreement Component (SP 800-56A rev3, SP 800-56B rev2)	Used to perform key agreement primitives on behalf of the calling application (does not establish keys into the module). Executes using DH Private, DH Public, EC DH Private, EC DH Public, RSA SGK, RSA SVK (passed in by the calling application)	1	Key structs (key agreement keys); flags	Status return; key agreement shared secret	Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)	Crypto-Officer - DH Private: W,E - DH Public: R - DH Peer Public: W,E - EC DH Private: W,E - EC DH Public: R - EC DH Peer's Public: W,E - RSA KDK: W,E - RSA KEK: R - RSA Peer's KEK: W,E - Shared Secret: G,R User - DH Private: W,E - DH Public: R - DH Peer Public: W,E - EC DH Private: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- EC DH Public: R - EC DH Peer's Public: W,E - RSA KDK: W,E - RSA KEK: R - RSA Peer's KEK: W,E - Shared Secret: G,R
Digital Signature	Used to generate or verify RSA, DSA, ECDSA, digital signatures. Executes using RSA SGK, RSA SVK; DSA SVK [verify only]; ECDSA SGK, ECDSA SVK, (passed in by the calling application)	1	Sign: signing key; message. Verify: signature value; flags; sizes	Status return; Signature value	Digital Signature Legacy Digital Signature	Crypto-Officer - RSA SGK: W,E - RSA SVK: W,E - DSA SVK: W,E - ECDSA SGK: W,E - ECDSA SVK: W,E User - RSA SGK: W,E - RSA SVK: W,E - DSA SVK: W,E - ECDSA SGK:

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						W,E - ECDSA SVK: W,E
Asymmetric Key Verification	Used to verify ECDSA keys	1	Public Key	Status return	Asymmetric Key Verification	Crypto-Officer - ECDSA SVK: W,E User - ECDSA SVK: W,E
Module initialization	The module is initialized when the provider is loaded	1	N/A	N/A	None	Crypto-Officer User
Perform Self-Test	Perform self-tests on demand	1	N/A	Success/failure message	None	Crypto-Officer User
Key Transport Encryption	Used for Key Transport on behalf of the calling application (does not establish keys into the module)	1	Key to be transported, Key	Status return, Encrypted key to be transported	Key Transport	Crypto-Officer - RSA Peer's KEK: W,E - Keying Material: W - KTS-IFC Encrypted Key: G,R User - RSA Peer's KEK: W,E - Keying Material: W

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- KTS-IFC Encrypted Key: G,R
Key Transport Decryption	Used for Key Transport on behalf of the calling application (does not establish keys into the module)	1	Encrypted Key, Key	Status return, Decrypted keying material that was transported	Key Transport	Crypto-Officer - RSA KDK: W,E - KTS-IFC Encrypted Key: W - Keying Material: G,R User - RSA KDK: W,E - KTS-IFC Encrypted Key: W - Keying Material: G,R
Show Module Name and Version	Used to output module name and version	N/A	N/A	name: CiscoSSL FIPS Provider; Version: 8.1	None	Crypto-Officer User
Show Status	Used to output module status	N/A	N/A	status: active	None	Crypto-Officer User

Table 12: Approved Services

Legend:
* [G]enerate: The module generates or derives the SSP.
* [R]ead: The SSP is read from the module (e.g. the SSP is output).
* [W]rite: The SSP is updated, imported, or written to the module.
* [E]xecute: The module uses the SSP in performing a cryptographic operation.
* [Z]eroise: The module zeroises the SSP.

The module meets all FIPS 140-3 level 1 requirements for Services. The initialization process is described in the Secure Distribution, Operation, and User Guidance section of this document.

CO services with associated input and output are listed in the above table. All the services provided by the module can be accessed by both the User and the Crypto Officer roles. The User Role (User) can load the module and call any of the API functions. The Crypto Officer Role (CO) is responsible for installation of the module on the host computer system and calling of any API functions.

4.4 Non-Approved Services

Name	Description	Algorithms	Role
DSA Key Generation	DSA Asymmetric Key Generation (FIPS 186-4)	DSA KeyGen	CO, User
Kerberos KDF	Kerberos RFC3961 section 5.1 KDF	KRB5KDF	CO, User

Table 13: Non-Approved Services

4.5 External Software/Firmware Loaded

Not Applicable for this module.

5 Software/Firmware Security

5.1 Integrity Techniques

The module runs a HMAC SHA2-256 integrity verification on the shared object file (fips.so) during initialization by the host application. The module also runs the self-test for HMAC SHA2-256 prior to running the integrity check.

5.2 Initiate on Demand

The operator can initiate on-demand integrity test by calling SELF_TEST_post() or rebooting the host platform.

5.3 Additional Information

The integrity verification key used for firmware integrity test is not an SSP.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Non-Modifiable

How Requirements are Satisfied:

The module was tested on the platforms listed in Table 2 for the purposes of this FIPS 140-3 validation. The module is expected to execute correctly on any production grade CPU with

commonly used operating system. No operational environment restrictions are required for operation in the approved mode.

CiscoSSL FIPS Provider is a Firmware module and classified as a non-modifiable OE. The requirements under ISO/IEC 19790, section 7.6 “Operational environment”, are met by the module for Level 1 firmware requirements.

7 Physical Security

Per ISO/IEC 19790, section 7.7, the module is defined as a multi-chip standalone firmware cryptographic module. The module runs on a host appliance made of production-grade components with standard passivation techniques.

8 Non-Invasive Security

Not Applicable for this module.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Volatile Memory	Dynamic

Table 14: Storage Areas

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), except for DRBG state values used for the module’s default key generation service. The module implements SP 800-90A rev1 compliant DRBG services for creation of symmetric keys, and for generation of elliptic curve, and RSA keys as shown in Table 4. The calling application is responsible for storage of generated keys returned by the module.

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input	Calling process	RAM	Plaintext	Manual	Electronic	
API Output	RAM	Calling process	Plaintext	Manual	Electronic	
API Input KW	Calling process	RAM	Encrypted	Manual	Electronic	Key Wrapping (KW)
API Output KW	RAM	Calling process	Encrypted	Manual	Electronic	Key Wrapping (KW)

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API Input KT	Calling process	RAM	Encrypted	Manual	Electronic	Key Transport
API Output KT	RAM	Calling process	Encrypted	Manual	Electronic	Key Transport

Table 15: SSP Input-Output Methods

All CSPs enter the module's boundary in plaintext except those being unwrapped (AES-KW/KWP decryption) or decapsulated (RSA decryption within KTS-IFS or KAS-IFC) as API parameters, associated by memory location. However, none crosses the physical perimeter module does not output CSPs, other than as explicit results of key generation services or keys passed into the module by the calling application.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
OPENSSL_cleanse()	API call clears the temporarily stored CSPs	Zeroized SSPs will no longer be accessible through API calls	Allowed
Power Cycle	Power Cycle zeroizes all stored SSPs	Operating System zeroizes all the stored SSPs	Allowed

Table 16: SSP Zeroization Methods

Zeroization of sensitive data is performed automatically by API function calls for temporarily stored CSPs. The calling application is responsible for parameters passed in and out of the module. Successful completion of the zeroization service is determined by clean execution of OPENSSL_cleanse() without any errors being returned or a successful reboot of the host platform.

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
RSA SGK	Used to generate Digital Signatures	2048, 3072, 4096 bits - 112, 128, 152 bits	Signature Generation Key - CSP	Asymmetric Key Generation		Digital Signature
RSA KDK	Used in Asymmetric Key Operatio	2048, 3072, 4096 bits -	Key Transport Key - CSP	Asymmetric Key		Key Transport

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	n to decrypt keys	112, 128, 152 bits		Generation		
ECDSA SGK	Used to generate Digital Signatures	256, 384, 521 bits - 128, 192, 256 bits	Signature Generation Key - CSP	Asymmetric Key Generation		Digital Signature
DH Private	Used for Key Agreement	2048, 3072, 4096 bits - 112, 128, 152 bits	Key Agreement Key - CSP	Asymmetric Key Generation		Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)
EC DH Private	Used for Key Agreement	256, 384, 521 bits - 128, 192, 256 bits	Key Agreement Key - CSP	Asymmetric Key Generation		Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)
AES EDK	Used for Symmetric encrypt and decrypt operations	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP	Random Number Generation		Symmetric Encrypt/Decrypt
AES CMAC	Used for MAC calculation and verification	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP	Random Number Generation		Keyed Hash (HMAC/KMAC/CMAC/GMAC)
AES GCM	Used for authenticated cipher operations	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP	Random Number Generation		Symmetric Encrypt/Decrypt Keyed Hash (HMAC/KMAC/CMAC/GMAC)
AES XTS	Used for cipher operation	128, 256 bits - 128, 256 bits	Symmetric Key - CSP	Random Number		Symmetric Encrypt/Decrypt

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
				Generation		
AES Key Wrapping	Used for key wrapping	128, 192, 256 bits - 128, 192, 256 bits	Symmetric Key - CSP	Random Number Generation		Key Wrapping (KW)
HMAC Key	Used for MAC generation and verification	128 to 524288 bits - greater than 128 bits	Keyed Hash - CSP	Random Number Generation		Keyed Hash (HMAC/KMAC/CMAC/GMAC)
KMAC Key	Used for MAC generation and verification	128-1024 bits - 128, 256 bit	Keyed Hash - CSP	Random Number Generation		Keyed Hash (HMAC/KMAC/CMAC/GMAC)
Entropy Input	Entropy input from an external source used for DRBG seeding, defined per FIPS 140-3 IG D.L	128-2^35 bits - 128 - 256 bits	Entropy Input - CSP			Random Number Generation
DRBG_C	Element of Hash DRBG state, defined per FIPS 140-3 IG D.L	440-888 bits - 160-256 bits	DRBG State - CSP	Random Number Generation		Random Number Generation
DRBG_Key	Element of CTR DRBG or	CTR_DRBG: 128-256,	CTR_DRBG_Key, HMAC_DRB	Random Number		Random Number Generation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	HMAC DRBG state, defined per FIPS 140-3 IG D.L	HMAC DRBG: 128-256 - CTR_DRBG: 128-256, HMAC DRBG: 128-256	G_Key - CSP	Generation		
DRBG_Seed	Seed used for DRBG Instantiation and Reseed, defined per FIPS 140-3 IG D.L	128-256 bit - 128-256 bit	DRBG Seed - CSP	Random Number Generation		Random Number Generation
DRBG_V	Element of CTR, Hash or HMAC DRBG state, defined per FIPS 140-3 IG D.L	CTR_DRBG: 128-256, Hash DRBG: 128-256, HMAC DRBG: 128-256 - CTR_DRBG: 128-256, Hash DRBG: 128-256, HMAC DRBG: 128-256	DRBG State - CSP	Random Number Generation		Random Number Generation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Shared Secret	Used for Key Agreement	- 112 - 256 bits	Shared Secret - CSP		Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)	Key Derivation Function (KDF)
KDF Derived Key	Key derived from KDFs	128, 256 bits - 128, 256 bits	Derived Key - CSP	Key Derivation Function (KDF)		
Keying Material	Plaintext keying material to be encrypted	112 bits or more - 112 bits or more	Symmetric Key - CSP			Key Wrapping (KW) Key Transport
AES-KW Encrypted Key	Encrypted Keying Material	112 bits or more - 112 bits or more	Symmetric Key - CSP	Key Wrapping (KW)		
KTS-IFC Encrypted Key	Encrypted Keying Material	112 bits or more - 112 bits or more	Symmetric Key - CSP	Key Transport		
TDES DK	TDES Decryption key	168 bits - 128 bits	Symmetric Key - CSP			Legacy Symmetric Decrypt
TDES CMAC	Used for MAC verification	168 bits - 128 bits	Symmetric Key - CSP			Legacy Keyed Hash CMAC
RSA SVK	RSA signature verification public key	2048, 3072, 4096 and larger bits - 112, 128, 152 bits	Verification Key - PSP	Asymmetric Key Generation		Digital Signature

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
RSA KEK	RSA key encryption (public key transport) key	2048, 3072, 4096 bits - 112, 128, 152 bits	Encryption Key - PSP	Asymmetric Key Generation		Key Transport
RSA Peer's KEK	RSA key encryption (public key transport) key	2048, 3072, 4096 bits - 112, 128, 152 bits	Encryption Key - PSP			Key Transport
DSA SVK	DSA signature verification public key	1024, 2048, 3072 bits - 80, 112, 128 bits	Verification Key - PSP			Legacy Digital Signature
ECDSA SVK	ECDSA signature verification public key	256, 384, 521 bits - 128, 192, 256 bits	Verification Key - PSP	Asymmetric Key Generation		Digital Signature
DH Public	DH public key agreement key	2048, 3072 bits - 112, 128 bits	Public Key Agreement Key - PSP	Asymmetric Key Generation		Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)
DH Peer Public	DH peer's public key agreement key	2048, 3072 bits - 112, 128 bits	Public Key Agreement Key - PSP			Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)
EC DH Public	EC DH public key agreement key	256, 384, 521 bits - 128, 192, 256 bits	Public Key Agreement Key - PSP	Asymmetric Key Generation		Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
EC DH Peer's Public	EC DH peer's public key agreement key	256, 384, 521 bits - 128, 192, 256 bits	Public Key Agreement Key - PSP			Key Agreement/Agreement Component (SP 800- 56A rev3, SP 800-56B rev2)

Table 17: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
RSA SGK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	RSA SVK:Paired With
RSA KDK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	RSA KEK:Paired With
ECDSA SGK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	ECDSA SVK:Paired With
DH Private	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DH Public:Paired With DH Peer Public:Used With
EC DH Private	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	EC DH Public:Paired With EC DH Peer's Public:Used With
AES EDK	API Input API	RAM:Plaintext	Until zeroized by	OPENSSL_cleanse() Power Cycle	

Name	Input-Output	Storage	Storage Duration	Zeroization	Related SSPs
	Output		reboot or API call		
AES CMAC	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
AES GCM	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
AES XTS	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
AES Key Wrapping	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	Keying Material:Encrypts Keying Material:Decrypts AES-KW Encrypted Key:Encrypts AES-KW Encrypted Key:Decrypts
HMAC Key	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
KMAC Key	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
Entropy Input	API Input	RAM:Plaintext	Until zeroized by reboot	OPENSSL_cleanse() Power Cycle	DRBG_Seed:Constituent

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
			or API call		
DRBG_C	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DRBG_Seed:Derived From DRBG_V:Used With
DRBG_Key	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DRBG_Seed:Derived From DRBG_V:Used With
DRBG_Seed		RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DRBG_C:Derives DRBG_Key:Derives DRBG_V:Derives Entropy Input:Incorporates
DRBG_V	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DRBG_Seed:Derived From DRBG_Key:Used With
Shared Secret	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DH Private:Established By DH Peer Public:Established By EC DH Private:Established By EC DH Peer's Public:Established By KDF Derived Key:Derives
KDF Derived Key	API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	Shared Secret:Derived From
Keying Material	API Input API	RAM:Plaintext	Until zeroized by	OPENSSL_cleanse() Power Cycle	AES Key Wrapping:Encrypted By

Name	Input-Output	Storage	Storage Duration	Zeroization	Related SSPs
	Output		reboot or API call		AES Key Wrapping:Decrypted By RSA KEK:Encrypted By RSA Peer's KEK:Encrypted By RSA KDK:Decrypted By AES-KW Encrypted Key:Plaintext form of KTS-IFC Encrypted Key:Plaintext form of
AES-KW Encrypted Key	API Input KW API Output KW	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	Keying Material:Encrypted form of
KTS-IFC Encrypted Key	API Input KT API Output KT	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	Keying Material:Encrypted form of
TDES DK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
TDES CMAC	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
RSA SVK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	RSA SGK:Paired With

Name	Input-Output	Storage	Storage Duration	Zeroization	Related SSPs
RSA KEK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	RSA KDK:Paired With KTS-IFC Encrypted Key:Encrypts
RSA Peer's KEK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	KTS-IFC Encrypted Key:Encrypts
DSA SVK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	
ECDSA SVK	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	ECDSA SGK:Paired With
DH Public	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DH Private:Paired With
DH Peer Public	API Input	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	DH Private:Used With
EC DH Public	API Input API Output	RAM:Plaintext	Until zeroized by reboot or API call	OPENSSL_cleanse() Power Cycle	EC DH Private:Paired With
EC DH Peer's Public	API Input	RAM:Plaintext	Until zeroized by	OPENSSL_cleanse() Power Cycle	EC DH Private:Used With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
			reboot or API call		

Table 18: SSP Table 2

9.5 Transitions

In line with SP 800-57 part 1, CMVP is expected to release an updated SP 800-131A *Transitioning the Use of Cryptographic Algorithms and Key Lengths* document announcing a transition at the end of 2030 for any algorithm or key length of less than 128 bit security.

Please see the latest revision of SP 800-131A and the CMVP Programmatic Transitions page for transitions that may affect this module.

FIPS 186-4 transition, Feb 5, 2024. The module mostly uses the digital signature algorithm compliant with the current FIPS 186-5 standard. However as allowed under IG C.K it does continue to support FIPS 186-4 DSA signature verification, which is legacy use when using keys with a strength of at least 112-bits; see section 2.7 on Legacy Use Algorithms. There is no further scheduled transition for this digital signature verification algorithm.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A5711)	256 bits	Firmware Integrity Test	SW/FW Integrity	Returns 1 when power up self tests succeed	The SELF_TEST_post() function performs all power-up self-tests listed above with no operator intervention required when the module loads, returning a “1” if all power-up self-tests succeed, and a “0” otherwise. The power-up self-tests may also be performed on-demand by calling this function and interpretation of the return code is the responsibility of the calling application
HMAC-SHA2-256 (A5712)	256 bits	Firmware Integrity Test	SW/FW Integrity	Returns 1 when power up	The SELF_TEST_post() function performs all power-up self-tests listed above with no operator intervention required

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
				self tests succeed	when the module loads, returning a “1” if all power-up self-tests succeed, and a “0” otherwise. The power-up self-tests may also be performed on-demand by calling this function and interpretation of the return code is the responsibility of the calling application

Table 19: Pre-Operational Self-Tests

The module performs the conditional HMAC cryptographic algorithm self-test and, if successful, proceeds to the pre-operational firmware integrity test. The module is single threaded and will not return to the calling application until the CASTs are complete. If the self-tests fail, the module goes to an error state and subsequent calls to the module will fail and thus no further cryptographic operations are possible. The CO can clear the error state by restarting the host platform.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB Encrypt KAT	128 bits	KAT	CAST	Returns 1 on successful completion	Encrypt KAT	Upon power-up and call of SELF_TEST_post( ) function
AES-ECB Decrypt KAT	128 bits	KAT	CAST	Returns 1 on successful completion	Decrypt KAT	Upon power-up and call of SELF_TEST_post( ) function
AES-GCM Encrypt KAT	256 bits	KAT	CAST	Returns 1 on successful completion	Encrypt KAT	Upon power-up and call of SELF_TEST_post( ) function
AES-GCM Decrypt KAT	256 bits	KAT	CAST	Returns 1 on successful	Decrypt KAT	Upon power-up and call of SELF_TEST_post( ) function

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				completion		
AES-CMAC Generate KAT	128 bits	KAT	CAST	Returns 1 on successful completion	Generate KAT	Upon power-up and call of SELF_TEST_post( ) function
Counter DRBG KAT	AES-128 with derivation function	KAT	CAST	Returns 1 on successful completion	Instantiate, Generate, Reseed	Upon power-up and call of SELF_TEST_post( ) function
Hash DRBG KAT	SHA2-256	KAT	CAST	Returns 1 on successful completion	Instantiate, Generate, Reseed	Upon power-up and call of SELF_TEST_post( ) function
HMAC DRBG KAT	SHA-1	KAT	CAST	Returns 1 on successful completion	Instantiate, Generate, Reseed	Upon power-up and call of SELF_TEST_post( ) function
Deterministic ECDSA Sign KAT	P-256 with SHA2-256	KAT	CAST	Returns 1 on successful completion	Sign KAT	Upon power-up and call of SELF_TEST_post( ) function
DSA Verify KAT	2048-bit with SHA2-256	KAT	CAST	Returns 1 on successful completion	Verify KAT	Upon power-up and call of SELF_TEST_post( ) function
ECDSA Sign KAT	P-256 with SHA2-256	KAT	CAST	Returns 1 on successful completion	Sign KAT	Upon power-up and call of SELF_TEST_post( ) function
ECDSA Verify KAT	P-256 with SHA2-256	KAT	CAST	Returns 1 on successfu	Verify KAT	Upon power-up and call of

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				l completio n		SELF_TEST_pos t( ) function
RSA Sign KAT	2048 bit with SHA2-256	KAT	CAS T	Returns 1 on successfu l completio n	Sign KAT	Upon power-up and call of SELF_TEST_pos t( ) function
RSA Verify KAT	2048 bit with SHA2-256	KAT	CAS T	Returns 1 on successfu l completio n	Verify KAT	Upon power-up and call of SELF_TEST_pos t( ) function
KAS-ECC-SSC KAT	P-256	KAT	CAS T	Returns 1 on successfu l completio n	Ephemeral Unified Shared Secret (Z) Computatio n	Upon power-up and call of SELF_TEST_pos t( ) function
KAS-FFC-SSC KAT	L=2048/N=25 6	KAT	CAS T	Returns 1 on successfu l completio n	dhEphem Shared Secret (Z) Computatio n	Upon power-up and call of SELF_TEST_pos t( ) function
KAS-IFC-SSC KAT	2048 bit	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-56B rev2] Section 8.2.2 RSA Primitive Computatio n 2.0	Upon power-up and call of SELF_TEST_pos t( ) function
KTS-IFC KAT	2048 bit	KAT	CAS T	Returns 1 on successfu l completio n	RSA Encrypt KAT & Decrypt KAT	Upon power-up and call of SELF_TEST_pos t( ) function
SHA-1 KAT	SHA-1	KAT	CAS T	Returns 1 on successfu l completio n	Simple SHA KAT	Upon power-up and call of SELF_TEST_pos t( ) function

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
SHA2-512 KAT	SHA2-512	KAT	CAST	Returns 1 on successful completion	Simple SHA KAT	Upon power-up and call of SELF_TEST_post( ) function
SHA3-256 KAT	SHA3=256	KAT	CAST	Returns 1 on successful completion	Simple SHA KAT	Upon power-up and call of SELF_TEST_post( ) function
HMAC-SHA2-256 KAT	SHA2-256 with a 256-bit key	KAT	CAST	Returns 1 on successful completion	Generate	Upon power-up and call of SELF_TEST_post( ) function
KMAC-128 KAT	KMAC-128	KAT	CAST	Returns 1 on successful completion	Generate	Upon power-up and call of SELF_TEST_post( ) function
KDA OneStep SP800-56Cr2 KAT	SHA2-224	KAT	CAST	Returns 1 on successful completion	[SP 800-56C rev2] Section 4 OneStep KDF (AKA OpenSSL single-step or SS-KDF)	Upon power-up and call of SELF_TEST_post( ) function
KDA TwoStep SP800-56Cr2 KAT	SHA2-256	KAT	CAST	Returns 1 on successful completion	[SP 800-56C rev2] Section 5 TwoStep KDF (HKDF variant)	Upon power-up and call of SELF_TEST_post( ) function
KDF ANS 9.42 KAT	SHA-1	KAT	CAST	Returns 1 on successful completion	[SP 800-135 rev1] Section 5.1 ANSI X9.42-2001 KDF KAT	Upon power-up and call of SELF_TEST_post( ) function
KDF ANS 9.63 KAT	SHA2-256	KAT	CAST	Returns 1 on successfu	[SP 800-135 rev1] Section 5.1	Upon power-up and call of

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
				l completio n	X9.63-2001 KDF KAT	SELF_TEST_pos t( ) function
KDF IKEv2 KAT	SHA-1	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-135 rev1] Section 4.1.2 IKEv2 KDF KAT	Upon power-up and call of SELF_TEST_pos t( ) function
KDF SNMP KAT	SHA1, 15-char password	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-135 rev1] Section 5.4 SNMPv3 KDF KAT	Upon power-up and call of SELF_TEST_pos t( ) function
KDF SP800-108 KAT	HMAC SHA2-256 (16-byte key)	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-108 rev1] Section 4.1 KAT for a Counter Mode KDF	Upon power-up and call of SELF_TEST_pos t( ) function
KDF SRTP KAT	AES-256-CTR	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-135 rev1] Section 5.3 SRTP KDF KAT	Upon power-up and call of SELF_TEST_pos t( ) function
KDF SSH KAT	SHA-1	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-135 rev1] Section 5.2 SSHv2 KDF KAT	Upon power-up and call of SELF_TEST_pos t( ) function
PBKDF KAT	SHA2-256, 9-byte password, 5-byte salt, iteration count of 4096	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-132] Section 5.3 KAT of Master Key derivation	Upon power-up and call of SELF_TEST_pos t( ) function
TLS v1.2 KDF RFC7627 KAT	SHA2-256	KAT	CAS T	Returns 1 on successfu l completio n	[SP 800-135 rev1] Section 4.2.2 TLS 1.2 KAT	Upon power-up and call of SELF_TEST_pos t( ) function
TLS v1.3 KDF KAT	TLS1.3 Extract and	KAT	CAS T	Returns 1 on	RFC8446] Section 7.1	Upon power-up and call of

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
	Expand 32 bytes			successful completion	TLS v1.3 KDF KAT	SELF_TEST_post( ) function
TDES-CBC Decrypt KAT	Keying Option: 1	KAT	CAST	Returns 1 on successful completion	Decrypt KAT	Upon power-up and call of SELF_TEST_post( ) function
TDES-CMAC Verify KAT	Keying Option: 1	KAT	CAST	Returns 1 on successful completion	Verify KAT	Upon power-up and call of SELF_TEST_post( ) function
ECDSA KeyGen PCT	PCT performed using the generated key pair	Sign, Verify PTC	PCT	Returns 1 on successful completion	Sign, Verify PCT	Performed on ECC (ECDSA, KAS- ECC CDH-Component, KAS- ECC-SSC) key pair generation, prior to returning the key pair on conclusion of the call
RSA KeyGen PCT	PCT performed using the generated key pair	Encrypt, Decrypt PTC	PCT	Returns 1 on successful completion	Encrypt, Decrypt PTC	Performed on IFC (RSA, KAS-IFC- SSC, KTS-IFC) key pair generation, prior to returning the key pair on conclusion of the call
KAS-FFC-SSC PCT	56Arev3 new keypair assurances	56Arev3 Section 5.6.2.1.4 option 'b' tests	PCT	Returns 1 on successful completion	56Arev3 Section 5.6.2.1.4 option 'b' tests	Performed of DH key generation, prior to returning the key pair on conclusion of the call

Table 20: Conditional Self-Tests

The SELF_TEST_post() function performs all self-tests listed above with no operator intervention required when the module loads. The module returns a “1” if all self-tests succeed,

and a “0” otherwise. The pre-operational and conditional self-tests may also be performed on-demand by calling this function and interpretation of the return code is the responsibility of the calling application.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A5711)	Firmware Integrity Test	SW/FW Integrity	On reboot or SELF_TEST_post() function call	Manual or reboot
HMAC-SHA2-256 (A5712)	Firmware Integrity Test	SW/FW Integrity	On reboot or SELF_TEST_post() function call	Manual or reboot

Table 21: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB Encrypt KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
AES-ECB Decrypt KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
AES-GCM Encrypt KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
AES-GCM Decrypt KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
AES-CMAC Generate KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
Counter DRBG KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
Hash DRBG KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
HMAC DRBG KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
Deterministic ECDSA Sign KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
DSA Verify KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA Sign KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
ECDSA Verify KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
RSA Sign KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
RSA Verify KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KAS-ECC-SSC KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KAS-FFC-SSC KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KAS-IFC-SSC KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KTS-IFC KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
SHA-1 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
SHA2-512 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
SHA3-256 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
HMAC-SHA2-256 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KMAC-128 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDA OneStep SP800-56Cr2 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDA TwoStep SP800-56Cr2 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF ANS 9.42 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF ANS 9.63 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF IKEv2 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF SNMP KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF SP800-108 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF SRTP KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
KDF SSH KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
PBKDF KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
TLS v1.2 KDF RFC7627 KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
TLS v1.3 KDF KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
TDES-CBC Decrypt KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
TDES-CMAC Verify KAT	KAT	CAST	On reboot or SELF_TEST_post() function call	Manual or reboot
ECDSA KeyGen PCT	Sign, Verify PTC	PCT	On Key Generation	Manual
RSA KeyGen PCT	Encrypt, Decrypt PTC	PCT	On Key Generation	Manual
KAS-FFC-SSC PCT	56Arev3 Section 5.6.2.1.4 option 'b' tests	PCT	On Key Generation	Manual

Table 22: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Critical Error State	This state is entered when pre-operational or conditional KAT self tests fail	Failure of pre-operational or conditional KAT self tests	Restarting the module	0
Soft Error State	This state is entered when conditional PCT self tests fail	Failure of conditional PCT self tests	Automatic after zeroising the keypair that failed the PCT	False, plus a console message listing the failed test

Table 23: Error States

If any pre-operational or conditional KAT self-test fails, an internal flag is set to prevent subsequent invocation of any cryptographic function calls. The module will only enter the Approved mode if the module is reloaded and the call to SELF_TEST_post() succeeds. The CAST used to perform the approved integrity technique is passed before the execution of the pre-operational firmware integrity test (HMAC- SHA2-256). If a conditional PCT self-test fails the module automatically zeroises the keypair that failed the test and then transitions out of that error state and back to normal operation.

10.5 Operator Initiation of Self-Tests

The operator can initiate the self-tests by calling SELF_TEST_post() or rebooting the host platform.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

Per FIPS 140-3 classification, this is a multi-chip standalone cryptographic module. CiscoSSL FIPS Provider 8.1 is a C language-based firmware module that runs on production grade chassis. A complete revision history of the source code is collaborated by Bitbucket, and version controlled by Git. Code changes are tracked by commits tied to a username. All User documents are tracked in Cisco Document Central which requires username/password and access permission.
Coverity runs static analysis on the source code before committing to the secure repository.

Secure Distribution
The module is distributed only for use by Cisco personnel and as such is accessible only from the secure Cisco internal repository. Only authorized Cisco personnel have access to the module. The SHA512 fingerprint of the validated distribution tarball file can be obtained by contacting Cisco.

Secure Initialization
The module is ready to use after extracting it from the distribution tarball. The operating system loads the module into its user space. The initialization sequence starts with a check of the

integrity of the runtime executable using a HMAC-SHA2-256 digest computed at build time. If the computed HMAC-SHA2-256 digest matches the stored known digest, then the cryptographic algorithm self-tests are performed. If any self-test fails, an internal global error flag is set to prevent subsequent invocation of any cryptographic function calls. Any such failure is a hard error that can only be recovered by reloading the module. Upon encountering a failure, the module will return an integer of 0. The module will only enter the Approved mode if the module is reloaded and the call to SELF_TEST_post() succeeds. The function call “. /openssl list -providers” returns the name and the version of the module.

Secure Operation
The tested operating systems segregate user processes into separate process spaces. Each process space is an independent virtual memory area that is logically separated from all other processes by the operating system firmware and hardware. The module functions entirely within the process space of the process that invokes it.
Additional information on switching between approved and non-approved mode is provided under “Mode Change Instructions and Status” in Section 2.4 of this SP.

11.2 Administrator Guidance

An additional guidance document, if required, can be obtained by contacting Cisco Systems, Inc. using the information posted on the validation certificate.

11.3 Non-Administrator Guidance

Not Applicable for this module.

12 Mitigation of Other Attacks

12.1 Attack List

The module implements two mitigations against timing-based side-channel attacks, namely Constant time Implementations and Blinding.

12.2 Mitigation Effectiveness

Constant-time Implementations protect cryptographic implementations in the Module against timing analysis since such attacks exploit differences in execution time depending on the cryptographic operation, and constant-time implementations ensure that the variations in execution time cannot be traced back to the key, CSP or secret data. Numeric Blinding protects the RSA, DSA and ECDSA algorithms from timing attacks. These algorithms are vulnerable to such attacks since attackers can measure the time of signature operations or RSA decryption.

To mitigate this the Module generates a random blinding factor which is provided as an input to the decryption/signature operation and is discarded once the operation has completed and resulted in an output. This makes it difficult for attackers to attempt timing attacks on such operations without the knowledge of the blinding factor and therefore the execution time cannot be correlated to the RSA/DSA/ECDSA key.