The OpenSSL Project

OpenSSL FIPS Provider

FIPS 140-3 Non-Proprietary Security Policy

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

Introduction
Federal Information Processing Standards Publication 140-3 — Security Requirements for Cryptographic Modules specifies requirements for cryptographic modules to be deployed in a Sensitive but Unclassified environment. The National Institute of Standards and Technology (NIST) and Canadian Centre for Cyber Security (CCCS) Cryptographic Module Validation Program (CMVP) run the FIPS 140-3 program. The NVLAP accredits independent testing labs to perform FIPS 140-3 testing; the CMVP validates modules meeting FIPS 140-3 validation. Validated is the term given to a module that is documented and tested against the FIPS 140-3 criteria.

More information is available on the CMVP website at:
https://csrc.nist.gov/projects/cryptographic-module-validation-program.

About this Document
This document describes the non-proprietary Security Policy for the OpenSSL FIPS Provider cryptographic module (hereafter referred to as “the Module”) from The OpenSSL Project. It contains specification of the security rules under which the Module operates, including the security rules derived from the requirements of the FIPS 140-3 standard.

The OpenSSL Project may also be referred to as “OpenSSL” in this document.

The following trademarks are referenced within this Security Policy:
• Linux®: Linux is the registered trademark of Linus Torvalds in the U.S. and other countries.
• Unix®: UNIX is a registered trademark of The Open Group.
• Microsoft Windows®: Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

1.2 Security Levels

The Module meets FIPS 140-3 overall Level 1 requirements, with security levels as follows:

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	N/A
8	Non-invasive security	N/A

Section	Title	Security Level
9	Sensitive security parameter management	1
10	Self-tests	1
11	Life-cycle assurance	3
12	Mitigation of other attacks	1
	Overall Level	1

Table 1: Security Levels

1.3 Additional Information

In accordance with AS02.05, [ISO19790] §7.7 *Physical Security* is optional and does not apply to the Module. In accordance with current CMVP policy, [ISO19790] §7.8 *Non-Invasive Security* is not applicable.

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The Module is a cryptographic software library providing a C-language application program interface (API) for use by applications that require cryptographic functionality and is designated as a software module with a multi-chip standalone embodiment based on the descriptions of [ISO19790] AS02.03. The Module is intended for use by US and Canadian Federal agencies and other markets that require FIPS 140-3 validated cryptographic functionality.

The Module's formal name and version are "OpenSSL FIPS Provider" and "3.1.2", respectively.

The Module design corresponds to the Module security rules. Security rules enforced by the Module are described in the appropriate context of this document.

Module Type: Software

Module Embodiment: MultiChipStand

Cryptographic Boundary:

Figure 1 depicts the Module operational environment, with the cryptographic boundary highlighted in red inclusive of all Module entry points (API calls). The Module is defined as a Software module per [ISO19790] AS02.03. The cryptographic boundary of the Module is the FIPS Provider, a dynamically loadable library. The Module performs no communication other than with the calling application via APIs that invoke the Module.

The pre-operational approved integrity test is performed over all components within the cryptographic boundary.

Tested Operational Environment's Physical Perimeter (TOEPP):

The Tested Operational Environment's Physical Perimeter (TOEPP) is the General Purpose Computer.

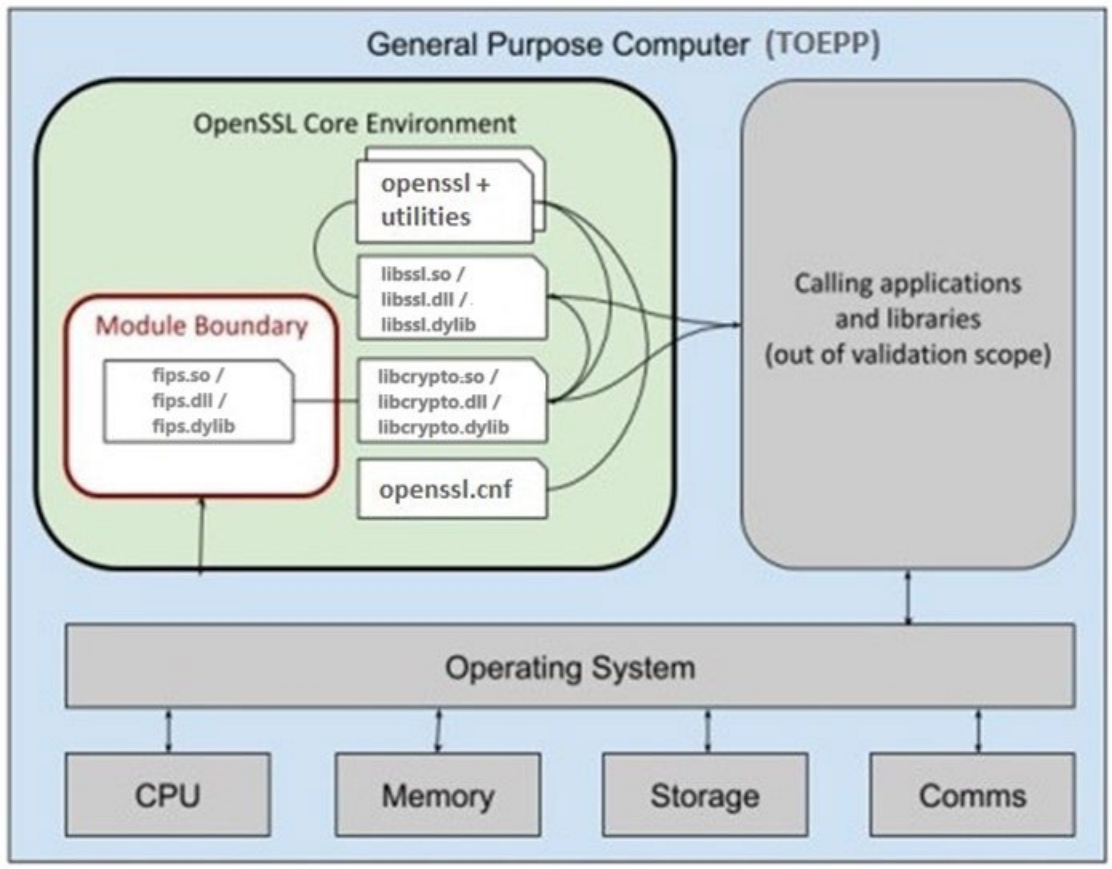

Figure 1: OpenSSL FIPS Provider Block Diagram

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	3.1.2	fips.so for Unix/Linux platforms	HMAC-SHA2-256
fips.dll	3.1.2	fips.dll for Windows platforms	HMAC-SHA2-256

Package or File Name	Software/ Firmware Version	Features	Integrity Test
fips.dylib	3.1.2	fips.dylib for Mac platforms	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)
Ubuntu Linux 22.04.1 Server	Dell Inspiron 7573	Intel i7-8550U	No	N/A	3.1.2
Ubuntu Linux 22.04.1 Server	Dell Inspiron 7573	Intel i7-8550U	Yes	N/A	3.1.2
Debian 11.5	Dell Inspiron 7573	Intel i7-8550U	No	N/A	3.1.2
Debian 11.5	Dell Inspiron 7573	Intel i7-8550U	Yes	N/A	3.1.2
FreeBSD 13.1	Dell Inspiron 7591 2 in 1	Intel i7-10510U	No	N/A	3.1.2
FreeBSD 13.1	Dell Inspiron 7591 2 in 1	Intel i7-10510U	Yes	N/A	3.1.2
Windows 10 Pro	Dell Inspiron 7591 2 in 1	Intel i7-10510U	No	N/A	3.1.2
Windows 10 Pro	Dell Inspiron 7591 2 in 1	Intel i7-10510U	Yes	N/A	3.1.2
macOS 11.5.2	Apple M1 Mac Mini	M1	No	N/A	3.1.2
macOS 11.5.2	Apple M1 Mac Mini	M1	Yes	N/A	3.1.2
macOS 11.5.2	Apple i7 Mac Mini	Intel i7	No	N/A	3.1.2
macOS 11.5.2	Apple i7 Mac Mini	Intel i7	Yes	N/A	3.1.2

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

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

No operational environments are vendor affirmed.

2.3 Excluded Components

No components are excluded from [FIPS140-3] requirements.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved mode	The module must be installed and configured per instructions provided in Section 11 of this document and the module is in the Approved mode by default as a result. The installation of the Module as described in Section 11 results in the settings described below this table, which are required for operation in the Approved mode	Approved	fips=yes
Non-Approved mode	The module is in the Approved mode of operation by default. Use of the non-Approved Algorithms Not Allowed in the Approved Mode will place the module in the non-approved mode of operation.	Non-Approved	fips=no

Table 4: Modes List and Description

The Module supports an Approved mode and a non-Approved mode of operation.

The inherent properties of the Module are:
1. Manual key entry is not supported.
2. Data output is inhibited during self-tests, zeroisation, SSP generation and error states.
3. The Module does not perform any cryptographic function if any self-test has failed.

The conditions for using the Module in the [FIPS140-3] Approved mode of operation are:
1. Installation of the Module as described in Section 11 results in the settings described below, which are required for operation in the Approved mode:

a. security-checks = 1
Enforce minimum key strengths and approved curve names.
b. conditional-errors = 1
Enforce the Module entering the error state on conditional test errors such as PCT failure.
c. drbg-no-trunc-md=1
Disallow use of truncated digests with HASH and HMAC DRBGs (IG D.R)
d. tls1-prf-ems-check=1
Enforce Extended Master Secret (EMS) use with TLS 1.2 (IG D.Q)

2. The Module is a cryptographic library used by a calling application. The calling application is responsible for:

a. Use of the primitives in the correct sequence.
b. Use of keys in accordance with [SP800-140Dr2] (as the keys used by the Module for cryptographic purposes are provided over the call stack by the calling application).
c. Use of a [SP800-90B] compliant entropy source. Entropy is supplied to the Module via callback functions. The callback functions return an error if the minimum entropy strength cannot be met.

Mode Change Instructions and Status:

Use of the Approved algorithms and Non-Approved Algorithms Allowed in the Approved Mode will ensure operation of the module in the Approved mode of operation. Use of the non-

Approved Algorithms Not Allowed in the Approved Mode will place the module in the non-approved mode of operation.

Degraded Mode Description:

The module does not support a degraded mode of operation.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A3548	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A3548	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A3548	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CCM	A3548	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A3548	Direction - Generation, Verification Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A3548	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-GMAC	A3548	Direction - Decrypt, Encrypt IV Generation - External, Internal IV Generation Mode - 8.2.1 Key Length - 128, 192, 256	SP 800-38D
AES-KW	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A3548	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A3548	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E

Algorithm	CAVP Cert	Properties	Reference
Counter DRBG	A3548	Prediction Resistance - Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - No, Yes	SP 800-90A Rev. 1
DSA KeyGen (FIPS186-4)	A3548	L - 2048, 3072 N - 224, 256	FIPS 186-4
DSA PQGGen (FIPS186-4)	A3548	L - 2048, 3072 N - 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA PQGVer (FIPS186-4)	A3548	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 SigGen (FIPS186-4)	A3548	L - 2048, 3072 N - 224, 256 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-4
DSA SigVer (FIPS186-4)	A3548	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 (FIPS186-4)	A3548	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Secret Generation Mode - Testing Candidates	FIPS 186-4
ECDSA KeyVer (FIPS186-4)	A3548	Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521	FIPS 186-4
ECDSA SigGen (FIPS186-4)	A3548	Component - No, Yes Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, 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 186-4
ECDSA SigVer (FIPS186-4)	A3548	Component - No, Yes Curve - B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521 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	FIPS 186-4
Hash DRBG	A3548	Prediction Resistance - Yes Mode - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-256, SHA3-512	SP 800-90A Rev. 1

Algorithm	CAVP Cert	Properties	Reference
HMAC DRBG	A3548	Prediction Resistance - Yes Mode - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-256, SHA3-512	SP 800-90A Rev. 1
HMAC-SHA-1	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-256	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A3548	Key Length - Key Length: 8-524288 Increment 8	FIPS 198-1
KAS-ECC CDH-Component SP800-56Ar3 (CVL)	A3548	Curve - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521	SP 800-56A Rev. 3
KAS-ECC-SSC Sp800-56Ar3	A3548	Domain Parameter Generation Methods - B-233, B-283, B-409, B-571, K-233, K-283, K-409, K-571, P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A3548	Domain Parameter Generation Methods - FB, FC, ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192 Scheme - dhEphem - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-IFC-SSC	A3548	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme -	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
		KAS1 - KAS Role - initiator, responder KAS2 - KAS Role - initiator, responder	
KDA HKDF SP800-56Cr2	A3548	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	A3548	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-8192 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A3548	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)	A3548	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)	A3548	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512 Key Data Length - Key Data Length: 128, 4096	SP 800-135 Rev. 1
KDF KMAC Sp800-108r1	A3548	Derived Key Length - Derived Key Length: 112-4096 Increment 8	SP 800-108 Rev. 1
KDF SP800-108	A3548	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 8, 72, 128, 776, 3456, 4096	SP 800-108 Rev. 1
KDF SSH (CVL)	A3548	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	A3548	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KMAC-256	A3548	Message Length - Message Length: 0-65536 Increment 8 Key Data Length - Key Data Length: 128-1024 Increment 8	SP 800-185
KTS-IFC	A3548	Modulo - 2048, 3072, 4096, 6144 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-	SP 800-56B Rev. 2

Algorithm	CAVP Cert	Properties	Reference
		basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 1024	
PBKDF	A3548	Iteration Count - Iteration Count: 1-10000 Increment 1 Password Length - Password Length: 8-128 Increment 8	SP 800-132
RSA KeyGen (FIPS186-4)	A3548	Key Generation Mode - B.3.3, B.3.6 Modulo - 2048, 3072, 4096 Primality Tests - Table C.2, Table C.3 Private Key Format - Standard	FIPS 186-4
RSA SigGen (FIPS186-4)	A3548	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 2048, 3072, 4096	FIPS 186-4
RSA Signature Primitive (CVL)	A3548	Private Key Format - CRT	FIPS 186-4
RSA SigVer (FIPS186-4)	A3548	Signature Type - ANSI X9.31, PKCS 1.5, PKCSPSS Modulo - 1024, 2048, 3072, 4096	FIPS 186-4
Safe Primes Key Generation	A3548	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
Safe Primes Key Verification	A3548	Safe Prime Groups - ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192	SP 800-56A Rev. 3
SHA-1	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-224	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4

Algorithm	CAVP Cert	Properties	Reference
SHA2-512/256	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA3-224	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-384	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-512	A3548	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHAKE-128	A3548	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A3548	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TLS v1.2 KDF RFC7627 (CVL)	A3548	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A3548	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 5: Approved Algorithms

The Module implements the Approved cryptographic functions listed in Table 5.

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
DSA PQGGen [FIPS 186-4]	Key Size, Key Strength:L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method:PQGGen using SHA3	OpenSSL Project OpenSSL 3.x FIPS Provider	Vendor affirmed per IG C.C and IG C.B Resolution (bullet point #3)
DSA PQGVer [FIPS 186-4]	Key Size, Key Strength:L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method:PQGVer using SHA3	OpenSSL Project OpenSSL 3.x FIPS Provider	Vendor affirmed per IG C.C and IG C.B Resolution (bullet point #3)
DSA SigGen [FIPS 186-4]	Key Size, Key Strength:L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128)	OpenSSL Project OpenSSL 3.x FIPS Provider	Vendor affirmed per IG C.C and IG C.B Resolution (bullet point #3)

Name	Properties	Implementation	Reference
	Mode/Method:SigGen using SHA3		
DSA SigVer [FIPS186-4]	Key Size, Key Strength:L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method:SigVer using SHA3	OpenSSL Project OpenSSL 3.x FIPS Provider	Vendor affirmed per IG C.C and IG C.B Resolution (bullet point #3)
CKG - Section 4 and 5.1	Key Type :Asymmetric	N/A	NIST SP800-133r2 Section 4: Using the Output of a Random Bit Generator; Section 5.1: Key Pairs for Digital Signature Schemes
CKG - Section 4 and 5.2	Key Type:Asymmetric	N/A	NIST SP800-133r2 Section 4: Using the Output of a Random Bit Generator; Section 5.2: Key Pairs for Key Establishment
CKG - Section 4 and Section 6.1	Key Type:Symmetric	N/A	NIST SP800-133r2 Section 4: Using the Output of a Random Bit Generator; Section 6.1: Direct Generation of Symmetric Keys
CKG - Section 6.2	Key Type:Symmetric	N/A	NIST SP 800-133r2 Section 6.2: Derivation of Symmetric keys
CKG - Section 6.3	Key Type:Symmetric	N/A	NIST SP 800-133rev2, Section 6.3: Symmetric Keys Produced by Combining Multiple Keys and Other Data
CKG – Section 4	Key Type:Symmetric	N/A	NIST SP800-133r2 Section 4: Using the Output of a Random Bit Random bits returned to the calling application

Table 6: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

Name	Properties	Implementation	Reference
AES	AES KW, KWP (Cert.#A3548):Symmetric key unwrapping	OpenSSL Project OpenSSL 3.x FIPS Provider	Per IG D.G Additional Comment 5

Table 7: Non-Approved, Allowed Algorithms

Non-Approved, Allowed Algorithms with No Security Claimed:

Name	Caveat	Use and Function
N/A	N/A	N/A

Table 8: Non-Approved, Allowed Algorithms with No Security Claimed

The module does not support any Non-Approved Algorithms Allowed in the Approved Mode of Operation with No Security Claimed.

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
Triple-DES	Provides 3-Key ECB and CBC mode, but indicated as fips=no, Encryption, Decryption
Ed448	SHAKE256, Ed448 provides 224 bits of security, Digital Signature Generation
Ed25519	SHA2-512, Ed25519 provides 128 bits of security, Digital Signature Generation
X448	Provides 224 bits of security, Key Agreement
X25519	Provides 128 bits of security, Key Agreement
ECDSA SigVer Component	Provides between 80 and 256 bits for security, Curves: B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521, Digital Signature Verification
FIPS 186-2 RSA SigGen/SigVer	Provides >= 80 bits of security, RSA signature generation/verification per FIPS 186-2
FIPS 186-2 RSA KeyGen	Provides >= 112 bits of security, RSA key generation per FIPS 186-2
X942KDF-CONCAT	Usage of X942KDF-CONCAT with PRF SHA-1, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256, KECCAK-KMAC128 and KECCAK-KMAC256
X963KDF	Usage of X963KDF with PRF SHA-1, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE128, SHAKE256, KECCAK-KMAC128 and KECCAK-KMAC256
HKDF	Provides < 112 bits of security, Usage of HKDF with key length less than 112 bits
OneStep KDF	Usage of OneStep KDF with PRF SHAKE128, SHAKE256
HMAC	Provides < 112 bits of security, Usage of HMAC with key length less than 112 bits for MAC generation
Hash and HMAC DRBG	Usage of Hash and HMAC DRBGs with PRFs SHA2-224, SHA2-384, SHA2-512/224 and SHA2-512/256

Table 9: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Symmetric Encryption and Decryption	BC-Auth BC-UnAuth	Symmetric Encryption and Decryption	Key Length:128, 192 and 256 bits Key Length (XTS):128 and 256 bits	AES-CBC AES-CBC-CS1 AES-CBC-CS2 AES-CBC-CS3 AES-CCM AES-CFB1 AES-CFB128 AES-CFB8 AES-CMAC AES-CTR AES-ECB AES-GCM AES-GMAC AES-OFB AES-XTS Testing Revision 2.0
Message Digest	SHA	Message Digest	SHA-1 :(s = 160) Large Message Sizes: 1, 2, 4, 8gigabytes SHA2:SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256). Large Message Sizes: 1, 2, 4, 8gigabytes SHA3:SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512). See Note 1. Large Message Sizes: 1, 2, 4, 8gigabytes SHAKE:SHAKE-128 (s = 128), SHAKE-	SHA-1 SHA2-224 SHA2-256 SHA2-384 SHA2-512 SHA2-512/224 SHA3-224 SHA3-256 SHA3-384 SHA3-512 SHAKE-128 SHAKE-256 SHA2-512/256

Name	Type	Description	Properties	Algorithms
			256 (s = 256). See Note 1.	
Keyed Hash	BC-Auth MAC	Keyed Hash	HMAC-SHA-1 [FIPS198-1]:SHA-1 (s = 160) HMAC-SHA2 [FIPS198-1]:SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256) HMAC-SHA3 [FIPS198-1]:SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512) KMAC:KMAC-128 (112 ≤ s ≤ 128), KMAC-256 (112 ≤ s ≤ 256). See Note 8.	HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512 HMAC-SHA2-512/224 HMAC-SHA2-512/256 HMAC-SHA3-224 HMAC-SHA3-256 HMAC-SHA3-384 HMAC-SHA3-512 AES-CMAC KMAC-128 KMAC-256 AES-GMAC
RSA Digital Signature Generation and Verification	DigSig-SigGen DigSig-SigVer	RSA Digital Signature Generation and Verification	Signature type: ANSI X9.31 tested with the listed moduli and the following hash algorithms: SHA2-256, SHA2-384, SHA2-512:k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152) Signature type: PKCS 1.5 tested with the listed moduli and the following hash algorithms: SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256:k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152)	RSA SigGen (FIPS186-4) RSA SigVer (FIPS186-4)

Name	Type	Description	Properties	Algorithms
			Signature type: PKCSPSS tested with the listed moduli and the following hash algorithms: SHA2- 224, SHA2-256, SHA2-384, SHA2- 512, SHA2-512/224, SHA2-512/256:k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152) Signature type: ANSI X9.31 tested with the listed moduli and the following hash algorithms: SHA-1*, SHA2-256, SHA2-384, SHA2-512:k=1024 (s ≤ 112), k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152) Signature type: PKCS 1.5 tested with the listed moduli and the following hash algorithms: SHA-1*, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256:k=1024 (s ≤ 112), k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152) Signature type: PKCSPSS tested with the listed moduli and the following hash algorithms: SHA-1*, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-	

Name	Type	Description	Properties	Algorithms
			512/256:k=1024 (s ≤ 112), k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152)	
ECDSA Signature Generation and Signature Verification	DigSig-SigGen DigSig-SigVer	ECDSA Signature Generation and Signature Verification	SigGen (includes SigGen Component) (tested with SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512):B-233, K-233, P-224 (s ~= 112); B-283, K-283, P-256 (s ~= 128); B-409, K-409, P-384 (s ~= 192); B-571, K-571, P-521 (s ~= 256) SigVer (tested with SHA-1*, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512):B-163, K-163, P-192 (s < 112); B-233, K-233, P-224 (s ~= 112); B-283, K-283, P-256 (s ~= 128); B-409, K-409, P-384 (s ~= 192); B-571, K-571, P-521 (s ~= 256)	ECDSA SigGen (FIPS186-4) ECDSA SigVer (FIPS186-4)
DSA Digital Signature Generation and Verification	DigSig-SigGen DigSig-SigVer	DSA Digital Signature Generation and Verification	SigGen (tested with SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256); SigGen using SHA3; no ACVP testing is available:L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s =	DSA SigGen (FIPS186-4) DSA SigVer (FIPS186-4) DSA SigGen [FIPS 186-4] Key Size, Key Strength: L = 2048/N = 224 (s = 112), L = 2048/N = 256

Name	Type	Description	Properties	Algorithms
			112) L = 3072/N = 256 (s = 128) SigVer (tested with SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256); SigVer using SHA3; no ACVP testing is available:L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128)	(s = 112) L = 3072/N = 256 (s = 128) Mode/Method: SigGen using SHA3 DSA SigVer [FIPS186-4] Key Size, Key Strength: L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method: SigVer using SHA3
RSA Signature Primitive	DigSig-SigGen	Signature primitive	Private Key format:CRT Public Exponent Mode:Fixed : k = 2048	RSA Signature Primitive
Asymmetric Key Pair Generation	AsymKeyPair-KeyGen AsymKeyPair-KeyVer	Generation of asymmetric key pairs	RSA KeyGen:k=2048 (s ~= 112), k=3072 (s ~= 128), k=4096 (s ~= 152) DSA KeyGen:L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) ECDSA KeyGen: Secret Generation Mode: Testing Candidates:B-233, K-233, P-224 (s ~= 112); B-283, K-283, P-256 (s ~= 128); B-409, K-409, P-384 (s ~= 192); B-571, K-571, P-521 (s ~= 256) Safe Primes Key Generation, Safe Primes Key	RSA KeyGen (FIPS186-4) DSA KeyGen (FIPS186-4) ECDSA KeyGen (FIPS186-4) Safe Primes Key Generation ECDSA KeyVer (FIPS186-4) Safe Primes Key Verification CKG - Section 4 and 5.1 Key Type : Asymmetric CKG - Section 4 and 5.2 Key Type: Asymmetric

Name	Type	Description	Properties	Algorithms
			Verification:ffdhe2048 (s = 112), ffdhe3072 (112 ≤ s ≤ 128), ffdhe4096 (112 ≤ s ≤ 152), ffdhe6144 (112 ≤ s ≤ 176), ffdhe8192 (112 ≤ s ≤ 200), MODP-2048 (s = 112), MODP-3072 (112 ≤ s ≤ 128), MODP-4096 (112 ≤ s ≤ 152), MODP-6144 (112 ≤ s ≤ 176), MODP-8192 (112 ≤ s ≤ 200) ECDSA KeyVer:B-163, K-163, P-192 (s < 112); B-233, K-233, P-224 (s ~= 112); B-283, K-283, P-256 (s ~= 128); B-409, K-409, P-384 (s ~= 192); B-571, K-571, P-521 (s ~= 256) DSA PQGGen (FIPS186-4), DSA PQGGen [FIPS 186-4] (VA):L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) DSA PQGVer (FIPS186-4), DSA PQGVer [FIPS 186-4] (VA):L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128)	DSA PQGGen (FIPS186-4) DSA PQGVer (FIPS186-4) DSA PQGGen [FIPS 186-4] Key Size, Key Strength: L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method: PQGGen using SHA3 DSA PQGVer [FIPS 186-4] Key Size, Key Strength: L = 1024/N = 160 (s < 112) L = 2048/N = 224 (s = 112), L = 2048/N = 256 (s = 112) L = 3072/N = 256 (s = 128) Mode/Method: PQGVer using SHA3
Random Number Generation	DRBG	Random Number Generation - Hash_DRBG, CTR_DRBG and HMAC_DRBG	Counter DRBG [SP800-90Ar1]:AES-128 (s = 128), AES-192 (s = 192), AES-256 (s = 256) Hash DRBG [SP800-90Ar1]:SHA-1 (s = 160), SHA2-256 (s =	Counter DRBG Hash DRBG HMAC DRBG CKG – Section 4 Key Type: Symmetric

Name	Type	Description	Properties	Algorithms
			256), SHA2-512 (s = 512) SHA3-256 (s = 256), SHA3-512 (s = 512) HMAC DRBG [SP800-90Ar1]:SHA-1 (s = 160), SHA2-256 (s = 256), SHA2-512 (s = 512) SHA3-256 (s = 256), SHA3-512 (s = 512)	
Key Derivation	KBKDF PBKDF	Derive Keying Material	KDA HKDF:SHA-1 (s = 160), SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256), SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512) KDA OneStep:SHA-1 (s = 160), SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256), SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512); HMAC-SHA-1 (s = 160), HMAC-SHA2-224 (s = 224), HMAC-SHA2-256 (s = 256), HMAC-SHA2-384 (s = 384), HMAC-SHA2-512 (s = 512), HMAC-SHA2-512/224 (s = 224), HMAC-SHA2-512/256 (s = 256), HMAC-SHA3-224 (s = 224), HMAC-SHA3-256 (s = 256),	KDA HKDF SP800-56Cr2 KDA OneStep SP800-56Cr2 KDA TwoStep SP800-56Cr2 KDF ANS 9.42 KDF ANS 9.63 KDF KMAC Sp800-108r1 KDF SP800-108 KDF SSH PBKDF TLS v1.2 KDF RFC7627 TLS v1.3 KDF CKG - Section 6.2 Key Type: Symmetric

Name	Type	Description	Properties	Algorithms
			HMAC-SHA3-384 (s = 384), HMAC-SHA3-512 (s = 512); KMAC-128 (112 ≤ s ≤ 128), KMAC-256 (112 ≤ s ≤ 256) KDA TwoStep [SP800-56Cr2]:HMAC-SHA-1 (s = 160), HMAC-SHA2-224 (s = 224), HMAC-SHA2-256 (s = 256), HMAC-SHA2-384 (s = 384), HMAC-SHA2-512 (s = 512), HMAC-SHA2-512/224 (s = 224), HMAC-SHA2-512/256 (s = 256), HMAC-SHA3-224 (s = 224), HMAC-SHA3-256 (s = 256), HMAC-SHA3-384 (s = 384), HMAC-SHA3-512 (s = 512) KDF ANS 9.42 [SP800-135r1]:SHA-1 (s = 160), SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256), SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512) KDF ANS 9.63 [SP800-135r1]:SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512) KDF KMAC [SP800-108r1]:KMAC-128 (112 ≤ s ≤ 128), KMAC-256 (112 ≤ s ≤ 256)	

Name	Type	Description	Properties	Algorithms
			KDF [SP800-108r1]:CMAC-AES128 (s = 128), CMAC-AES192 (s = 192), CMAC-AES256 (s = 256), HMAC-SHA-1 (s = 160), HMAC-SHA2-224 (s = 224), HMAC-SHA2-256 (s = 256), HMAC-SHA2-384 (s = 384), HMAC-SHA2-512 (s = 512), HMAC-SHA2-512/224 (s = 224), HMAC-SHA2-512/256 (s = 256), HMAC-SHA3-224 (s = 224), HMAC-SHA3-256 (s = 256), HMAC-SHA3-384 (s = 384), HMAC-SHA3-512 (s = 512) KDF SSH [SP800-135r1]:AES-128 (s = 128), AES-192 (s = 192), AES-256 (s = 256); SHA-1 (s = 160), SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512) PBKDF [SP800-132]:SHA-1 (s = 160), SHA2-224 (s = 224), SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512), SHA2-512/224 (s = 224), SHA2-512/256 (s = 256), SHA3-224 (s = 224), SHA3-256 (s = 256), SHA3-384 (s = 384), SHA3-512 (s = 512) TLS v1.2 KDF RFC7627: TLS [RFC7627] key derivation with	

Name	Type	Description	Properties	Algorithms
			Extended Master Secret (EMS) support, using the listed hash algorithms:SHA2-256 (s = 256), SHA2-384 (s = 384), SHA2-512 (s = 512) TLS v1.3 KDF [RFC8446]:HMAC-SHA2-256 (s = 256), HMAC-SHA2-384 (s = 384)	
KAS-1	KAS-SSC	Scheme: EphemeralUnified, KAS Role: Initiator, Responder	SP800-56Ar3 KAS-ECC-SSC per IG D.F Scenario 2 path (1):B-233, K-233, P-224, B-283, K-283, P-256, B-409, K-409, P-384, B-571, K-571, and P-521 curves providing 112, 128, 192, or 256 bits of encryption strength	KAS-ECC-SSC Sp800-56Ar3
KAS-2	KAS-SSC	Scheme: dhEphem. KAS Role: Initiator, Responder	SP800-56Ar3 KAS-FFC-SSC IG D.F Scenario 2 path (1):2048, 3072, 4096, 6144, and 8192-bit key providing 112, 128, 152, 176, or 200 bits of encryption strength	KAS-FFC-SSC Sp800-56Ar3
KAS-3	KAS-SSC	Scheme: KAS1, KAS2. KAS Role: Initiator, Responder	SP800-56Br2 KAS-IFC-SSC IG D.F Scenario 1 path (1):2048, 3072, 4096, 6144, and 8192-bit key providing 112, 128, 152, 176, or 200 bits of encryption strength	KAS-IFC-SSC
KTS-1	KTS-Wrap	Key Transport in compliance with [SP800- 38F] when approved using AES KW or KWP	SP 800-38F KTS (key wrapping) per IG D.G :128, 192, and 256-bit keys providing 128, 192, or 256 bits of encryption strength	AES-KW AES-KWP

Name	Type	Description	Properties	Algorithms
KTS-2	KTS-Wrap	Key Transport in compliance with [SP800- 38F] when approved AES (any mode) and approved HMAC, KMAC, GMAC or CMAC are used in combination	SP 800-38F KTS (key wrapping) per IG D.G : 128, 192, and 256-bit keys providing 128, 192, or 256 bits of encryption strength	AES-CBC AES-CFB1 AES-CFB128 AES-CFB8 AES-CTR AES-ECB AES-OFB AES-XTS Testing Revision 2.0 AES-CBC-CS2 AES-CBC-CS3 AES-CCM AES-CMAC AES-GCM AES-GMAC AES-KW AES-KWP HMAC-SHA-1 HMAC-SHA2-224 HMAC-SHA2-256 HMAC-SHA2-384 HMAC-SHA2-512 HMAC-SHA2-512/224 HMAC-SHA2-512/256 HMAC-SHA3-224 HMAC-SHA3-256 HMAC-SHA3-384 HMAC-SHA3-512 KMAC-128 KMAC-256 AES-CBC-CS1
KTS-3	KTS-Wrap	Key Transport in compliance with [SP800- 38F] when approved using an	SP 800-38F KTS (key wrapping) per IG D.G : 128, 192, and 256-bit keys providing 128, 192, or	AES-CCM AES-CMAC AES-GCM AES-GMAC

Name	Type	Description	Properties	Algorithms
		Authenticated AES mode (AES CCM; AES GCM; AES GMAC; AES CMAC)	256 bits of encryption strength	
KTS-4	KTS-Encap	Key Transport; Scheme: KTS-OAEP-basic (no key confirmation): RSA-OAEP, Key Encapsulation, Key Unencapsulation Key Generation Methods: rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2- prime-factor	SP 800-56Brev2 KTS-IFC (key encapsulation and un-encapsulation) per IG D.G:2048, 3072, 4096, and 6144-bit key providing 112, 128, 152, or 176 bits of encryption strength	KTS-IFC
KAS ECC CDH Component	KAS-SSC	KAS-ECC-SSC primitive	Curves:B-233, K-233, P-224 (s ~= 112); B-283, K-283, P-256 (s ~= 128); B-409, K-409, P-384 (s ~= 192); B-571, K-571, P-521 (s ~= 256).	KAS-ECC CDH-Component SP800-56Ar3
Perform self-tests (All)	BC-Auth BC-UnAuth DigSig-SigGen DigSig-SigVer DRBG KAS-SSC KBKDF MAC PBKDF SHA XOF	All self-tests executed by the module at boot		AES-ECB AES-GCM Hash DRBG Counter DRBG HMAC DRBG DSA SigGen (FIPS186-4) DSA SigVer (FIPS186-4) ECDSA SigGen (FIPS186-4) ECDSA SigVer (FIPS186-4) RSA SigGen (FIPS186-4) RSA SigVer (FIPS186-4)

Name	Type	Description	Properties	Algorithms
				HMAC-SHA2-256 SHA-1 SHA3-256 SHA2-512 KDF ANS 9.42 KDF ANS 9.63 KAS-ECC-SSC Sp800-56Ar3 KAS-FFC-SSC Sp800-56Ar3 KAS-IFC-SSC KDA OneStep SP800-56Cr2 KDA TwoStep SP800-56Cr2 KDF SSH KDF SP800-108 PBKDF TLS v1.2 KDF RFC7627 TLS v1.3 KDF
Cryptographic Key Generation (CKG)	CKG	Direct generation of symmetric keys per NIST SP 800-133r2		CKG - Section 4 and Section 6.1
Software Integrity Test	MAC	HMAC-SHA2-256 used to perform the software integrity test	Key size: 256 bits	HMAC-SHA2-256
Cryptographic Key Generation (CKG) - AES XTS	CKG	AES XTS Key generated to comply with the approved key generation guidelines of NIST SP 800-133rev2, Section 6.3, Symmetric Keys Produced by Combining Multiple Keys and Other Data	Key size:128, 256 bits	CKG - Section 6.3

Table 10: Security Function Implementations

Equivalent strength in bits is given for each key or algorithm type (as some algorithms do not use or produce keys). The term s is used throughout to indicate security strength, following the notation used in the majority of the sources.

Note 1: Preimage resistance strength applies to hash algorithms used in DRBG, KDFs. Described also in [SP800-57P1r5] Table 3.
Note 2: Elliptic curve strengths are annotated as approximate (i.e., s ~=) since [SP800-186] Table 1 provides approximate security strengths.
Note 3: [SP800-186] (cited in [SP800-140Cr2]) and [FIPS140-3_IG] C.K indicate that the Binary (B-) and Koblitz (K-) curves are deprecated.
Note 4: Approved elliptic curves for ECC key agreement are given in [SP800-56Ar3] Table 24.
Note 5: In Digital Signature applications, security strength is primarily associated with the asymmetric key pair specification. The hash function used must have equivalent strength equal to or greater than the security strength of the associated key pair.
Note 6: Approved key types for FFC key agreement are given in [SP800-56Ar3] Tables 25, 26. The group notation of Table 26 is used for consistency with CAVP algorithm listings and ACVP capability registration.
Note 7: Approved key types for IFC key agreement are given in [SP800-56Br2] Table 4. IFC key types approved for Digital Signature Generation and Verification are given also in [SP800-57P1r5] Table 2. Equivalent strengths are annotated as approximate (i.e., s ~=) since [SP800-56Br2] Table 4 provides approximate security strengths.
Note 8: Security strengths for KDA One Step are given in [SP800-56Cr2] Table 1 (hash), Table 2 (HMAC) and Table 3 (KMAC).
Note 9: Security strength for L=2048/N=256 is determined in accordance with [FIPS140-3_IG] D.B *Strength of SSP Establishment Methods* as y = min(x, N/2), where x is 112 and therefore y = min(112, 128) = 112.

Other reference sources for the strengths are as follows:
• AES (AES-128, AES-192, AES-256): [SP800-57P1r5] Table 2.
• ECC (B-163, B-233, B-283, B-409, B-571, K-163, K-233, K-283, K-409, K-571, P-192, P-224, P-256, P-384, P-521): [SP800-186] Table 1.
• FFC (L=1024/N=160, L=2048/N=224, L=2048/N=256, L=3072/N=256): [SP800-57P1r5] Table 2.
• FFC (ffdhe2048, ffdhe3072, ffdhe4096, ffdhe6144, ffdhe8192, MODP-2048, MODP-3072, MODP-4096, MODP-6144, MODP-8192): [SP800-56Ar3] Tables 25 and 26.
• IFC (k=1024, k=2048, k=3072, k=4096, k=6144, k=8192): [SP800-56Br2] Table 4.
• KMAC (KMAC128, KMAC256): [SP800-56Cr2] Table 3.
• SHA-1, SHA2 (SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256): [SP800-107] Table 1.
• SHA3 (SHA3-224, SHA3-256, SHA3-384, SHA3-512): [SP800-57P1r5] Table 3.
• SHAKE (SHAKE128, SHAKE256): [SP800-185] Section 8.1.

2.7 Algorithm Specific Information

a. AES-GCM Usage
   AES GCM IV generation must be compliant to [FIPS140-3_IG] C.H Key/IV Pair Uniqueness Requirements from SP 800-38D Scenario 1(a), tested per option (ii) under C.H TLS/DTLS 1.2 protocol IV generation per RFC7627, Scenario 1(d) SSHv2 per RFC4252, RFC4253 and RFC5647 and Scenario 5 TLS 1.3 per RFC8446. IV

constructed in compliance with a protocol shall only be used in the context of the AES-GCM mode encryptions within the protocol.

The Module does not implement the TLS and SSH protocols itself, however, it provides the cryptographic functions required for implementing the protocols. AES GCM encryption is used in the context of the SSH and TLS protocol versions 1.2 and 1.3. The module provides the primitives to support the AES GCM ciphersuites from [SP800-52r1] Section 3.3.1. The module's implementation of AES-GCM is used together with an application that runs outside the module's cryptographic boundary. The application negotiates the protocol session's keys and the 32-bit nonce value of the IV.

When the IV exhausts the maximum number of possible values for a given session key (2^64 - 1), this results in a failure in encryption and a handshake to establish a new encryption key will be required. It is the responsibility of the user of the module, i.e., the first party, client or server, to encounter this condition, to trigger this handshake in accordance with the TLS/SSH protocol.

The Module also supports internal IV generation using the module's approved DRBG. The IV is at least 96 bits in length per [SP800-38D] Section 8.2.2. Per [FIPS140-3_IG] C.H Scenario 2 and [SP800-38D], the approved DRBG generates outputs such that the (key, IV) pair collision probability is less than 2^-32.

In each case, in the event that the Module power is lost and restored the user must ensure that the AES GCM encryption/decryption keys are re-distributed in accordance with IG C.H Scenario 3. The module does not support persistent storage of SSPs.

The Module also supports importing of GCM IVs when an IV is not generated within the Module. In the approved mode, an IV must not be imported for encryption from outside the cryptographic boundary of the Module as this will result in a non-conformance. This is in accordance with IG 2.4.A: If the module operator (e.g., calling application) can do things outside of the module's control/visibility that can take an otherwise approved algorithm and use it in a non-approved way (e.g., use PBKDF and/or AES XTS outside of storage applications), the corresponding module service may still be considered approved (and if so, shall have an approved indicator per AS02.24) and the Security Policy shall clarify how to use the service in an approved manner (per ISO 19790 B.2.2 on Overall security design and the rules of operation).

b. PBKDF Usage

The lower limit on the supported length of a password/passphrase used in key derivation is 1-character. The ASCII system comprises of 94 printable characters (letters, digits, punctuation, and symbols). For a 1-character password/passphrase chosen from 94 printable ASCII characters, the total combinations are: 94^1. Thus, the probability of guessing the correct password/passphrase on a random attempt is: 1/94^1 ~0.010.

The module being a software module, does not restrict the usage of a password/string used as the password and input to the PBKDF. The onus is on the calling application to provide a password of an appropriate length based on the intended security strength (and size) of the key to be derived.

In accordance with NIST SP 800-132, passwords shorter than 10 characters are usually considered to be weak. There are many other properties that may render a password

weak. For example, it is not advisable to use sequences of numbers or sequences of letters as passwords. Easily accessed personal information, such as the user's name, phone number, and date of birth, should not be used directly as a password.

Passphrases frequently consist solely of letters, but they make up for their lack of entropy by being much longer than passwords, typically 20 to 30 characters. Passphrases shorter than 20 characters are usually considered weak.

The module complies with NIST SP 800-132 Section 5.4 Option 1 a and IG D.N. The iteration count values used range from 1 to 10000 per NIST SP 800-132 Section 5.2 whereby the iteration count shall be selected as large as possible, as long as the time required to generate the key using the entered password is acceptable for the users. Keys derived from passwords, as shown in SP 800-132, may only be used in storage applications. The security strength of the derived key is at least 112 bits. The module implements CKG per NIST SP 800-133r2 Section 6.2.2.

c. AES-XTS Usage
Usage In accordance with [SP800-38E], the XTS-AES algorithm shall only be used for confidentiality on storage devices. The Module complies with [FIPS140-3_IG] C.I by explicitly checking that Key_1 ≠ Key_2 before using the keys in the XTS-AES algorithm to process data with them. The module implements CKG per NIST SP 800-133r2 Section 6.3.

d. Legacy Usage
The module supports the following implementations for legacy use/support per NIST SP 800-131Ar2:
    - RSA (modulus 1024 bits), DSA (modulus 1024 bits), ECDSA (B-163, K-163 and P-192, curves) digital signature verification providing less than 112 bits of security strength.
    - RSA, ECDSA and DSA digital signature verification with SHA-1 used as the underlying hash algorithm.

e. Component Validation List (CVL)
In accordance with IG 2.4.B, all tested components that may be called during the operation of the module and shown in the module's CVL certificates have been listed individually in Table 5. All vendor affirmed components that may be called during the operation of the module have also been listed individually in Table 6 per IG 2.4.B.

f. FIPS 202 Usage
In accordance with IG C.C Resolution 2. a., each SHA-3 and SHAKE function has been tested and validated on all of the module's operating environments. Per Resolution 2. c., SHA-3 hash functions used as part of the higher-level DRBG algorithms for which the CAVP testing is not yet available have been vendor affirmed as documented in Table 6.

g. RSA Usage
    - Per IG C.E, the module generates RSA signature keys using an approved key generation procedure per RSA KeyGen validated for conformance to FIPS 186-4 Cert. #A3548.
    - Per IG C.F, the RSA SigGen and SigVer implementations have been tested for all implemented RSA modulus lengths where CAVP testing is available. The

module supports generation of RSA keys with the following untested approved moduli/sizes: 4096 <nlen<= 16384. The module also supports the following untested, approved moduli for the RSA SigGen and SigVer: 4096 <nlen<= 16384.

h. TLS 1.2 KDF
In accordance with IG D.Q, the module has been implemented to enforce usage of the extended master secret in the TLS 1.2 KDF as specified per Section 2.4.1. d. (tls1-prf-ems-check set option set to 1 due to initialization of the module per Section 11). The module complies with RFC 7627.

i. DRBG Usage
Per IG D.R, the Hash_DRBG and HMAC_DRBG implementations use SHA-1, SHA2-256, SHA2-512, SHA3-256 and SHA3-512.

j. NIST SP 800-108 KDF Usage
The SP 800-108 KDF is not used to generate asymmetric keys directly in that the module restricts generation of keys to approved methods only. For keys passed into the module, the onus lies on the calling application to ensure correct generation of such keys using approved mechanisms. The module implements CKG per NIST SP 800-133r2 Section 6.2.3.

k. SHA-1 Usage:
The module implements SHA-1 for usage in the following (this can be vetted from the SFI Table 9 in the Security Policy):

I. As a PRF in the KDFs X942 KDF-CONCAT, X963 KDF, KDA HKDF, KDA OneStep, KDF, ANS 9.42 [SP800-135r1], KDF SSH [SP800-135r1], PBKDF [SP800-132],
II. As a standalone SHA-1 hash function
III. As a PRF in HMAC-SHA-1
IV. As the underlying hash function for RSA SigVer, ECDSA SigVer and DSA SigVer for legacy use/support per NIST SP 800-131Ar2 as specified in the Security Policy Section 2.7 d.
V. As the underlying hash function in Hash DRBG and HMAC DRBG

2.8 RBG and Entropy

The Module relies on the use of a [SP800-90B] compliant entropy source outside the Module boundary. The calling application is responsible for use of an [SP800-90B] compliant entropy source with sufficient entropy based on the required security strength. Entropy is supplied to the Module via callback functions (see Section 2.4 2. c). Minimum Number of Bits of Entropy, depending on the target security strength of generated SSPs are 128, 192 or 256 bits. When using the Counter DRBG implementation without the derivation function enabled, full entropy from the entropy source is required. The following caveat applies to the module: No assurance of the minimum strength of generated SSPs (e.g., keys).

2.9 Key Generation

The module implements NIST SP 800-90Ar1 DRBGs and supports the following sections per NIST SP 800-133r2 (CKG): Sections 4, 5.1, 5.2, 6.1, 6.2 and 6.3.

2.10 Key Establishment

Key Agreement

Per IG D.F:

The module supports Key Agreement Schemes per NIST SP800-56Ar3 and [FIPS140-3_IG] D.F Scenario 2 (path 1) and NIST SP 800-56Br2 and [FIPS140-3_IG] D.F Scenario 1 (path 1). The KAS-1, KAS-2, KAS-3 in the SFI Table 9 have been documented accordingly. The Approved Algorithm list includes the tested components (KAS-ECC-SSC, KAS-FFC-SSC and KAS-IFC-SSC) as individual entries.

The Module obtains the [FIPS140-3_IG] D.F required key agreement assurances:
[SP800-56Ar3] in accordance with Section 5.6.2.
[SP800-56Br2] in accordance with Section 6.4.

Per IG C.F Additional Comment 1.e:
The elliptic curve used in the key agreement scheme and the associated domain parameters provide more than 112 bits of security as seen in the KAS-1 entry per Table 9.

Per IG C.F Additional Comment 2:
The KAS-ECC-SSC and KAS-FFC-SSC implementations each support a scheme of the Diffie-Helllman variety.

Per IG D.G:
The module supports the Key Transport per NIST SP 800-56Br2 (RSA-OAEP) denoted by KTS-4 in the SFI Table 9. The RSA modulus sizes and key generation method have been documented in the table as well. The module can also optionally be used in the context of IETF protocols and provide key transport using any approved AES mode(s) and an approved MAC.

The corresponding entries KTS-1, KTS-2 and KTS-3 in the SFI Table 9 have been documented accordingly. All KTS entries have been documented in accordance with Additional Comment 4 in the IG. The module also supports the following untested approved moduli for KTS-4: 6144 < nlen <=16384, where nlen denotes the modulus.

Per IG D.A and IG D.B:
The strengths of the established key have been documented in accordance with IG D.A Additional Comment 4. and per the Resolution in IG D.B.

2.11 Industry Protocols

The Module conforms to Resolution 3 per [FIPS140-3_IG] D.C *References to the Support of Industry Protocols*: while it provides [SP800-56Ar3] conformant schemes and API entry points oriented to SSH and TLS usage, the Module does not contain the full implementation of SSH or TLS. The following caveat is required:

No parts of the SSH and TLS 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	Control Input	API entry point: stack frame including non-sensitive parameters
N/A	Data Input	API call parameters passed by reference or value for cryptographic service input
N/A	Status Output	API return value: enumerated status resulting from call execution
N/A	Data Output	API call parameters passed by reference for cryptographic service output

Table 11: Ports and Interfaces

Table 10 defines the Module's [FIPS140-3] logical interfaces; the Module does not interact with physical ports. The Control Output logical interface is not applicable to the Module and is intentionally omitted from Table 10.

4 Roles, Services, and Authentication

4.1 Authentication Methods

The Module does not provide an authentication or identification method of its own. The CO role is assumed by meeting the conditions of Section 11 of this document.

4.2 Roles

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

Table 12: Roles

The Module supports the mandatory Cryptographic Officer (CO) operational role only (implicitly identified) and does not support a maintenance role or a bypass capability.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Initialize	Module initialization	FIPS_OK	Core handle, dispatch in and out, provider context	Initialization status (1 = pass, 0 = fail)	Random Number Generation	Crypto Officer - DRBG_EI: G,W,E,Z - DRBG_State: G - Software Integrity key: E
Core (all except Teardown) (Show Status, Show Version)	Show status; Core operations dispatched by FIPS provider: Metadata (Gettable parameters; Get parameters; Get capabilities); Query; Self-test	FIPS_OK	Provider context, parameters types (array), capability, callback pointer and arguments, operation ID	Parameter types (array) with: Name, Version, BuildInfo, Status, SecurityChecks; Status return, TLS group capabilities, Null or array of available operations	None	Crypto Officer
Core: Perform self-tests	Run the self-test sequence	FIPS_OK	Provider context	Status (1 = pass, 0 = fail)	Perform self-tests (All) Software Integrity Test	Crypto Officer
Core: Teardown (Perform zeroisation)	Uninstantiate the module; includes Zeroise	FIPS_OK	Provider context	None	None	Crypto Officer - DS_SGK: Z - DS_SVK: Z - GKP_Pri

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						vate: Z - GKP_Public: Z - KAS_Private: Z - KAS_Public: Z - KAS_SS: Z - KD_DKM: Z - KH_Key: Z - KTS_KDK: Z - KTS_KEK: Z - KTS_SS: Z - DRBG_EI: Z - DRBG_Seed: Z - DRBG_State: Z - SC_EDK: Z - Software Integrity key: Z
Asymmetric cipher (Key Transport) (Perform	Encapsulate or decapsulate key material	[KTS-IFC: RSA, 4, (2048,	Encapsulate: Key struct (KTS_KD	Status return; KTS_SS	KTS-4	Crypto Officer - KTS_KD

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
approved security functions)	on behalf of the calling process (does not establish keys into the module)	3072, 4096, 6144, 8192)]	K); Decapsulate (KTS_KEK)			K: E - KTS_KEK: E - KTS_SS: R
Cipher (Encryption/Decryption and Key Wrapping) (Perform approved security functions)	Encrypt or decrypt data, including AEAD modes (CCM, GCM) and key wrap (KW, KWP) (CSPs are passed in by the calling process or generated within the module)	[AES-ECB: AES-128-ECB, AES-192-ECB, AES-256-ECB]; [AES-CBC: AES-128-CBC, AES-192-CBC, AES-256-CBC]; [AES-CBC-CS: AES-128-CBC-CTS, AES-192-CBC-CTS, AES-256-CBC-CTS]; [AES-OFB: AES-128-	SC_EDK and KH_Key (for key wrapping); flags	Status return. Plaintext or ciphertext data, or wrapped key	Symmetric Encryption and Decryption Keyed Hash KTS-1 KTS-2 KTS-3 Cryptographic Key Generation (CKG) Cryptographic Key Generation (CKG) - AES XTS	Crypto Officer - SC_EDK: E - KH_Key: E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		OFB, AES-192-OFB, AES-256-OFB]; [AES-CFB1: AES-128-CFB1, AES-192-CFB1, AES-256-CFB1]; [AES-CFB8: AES-128-CFB8, AES-192-CFB8, AES-256-CFB8]; [AES-CFB128: AES-128-CFB, AES-192-CFB, AES-256-CFB]; [AES-CTR: AES-128-CTR, AES-192-CTR,				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		AES-256-CTR]; [AES-CCM: AES-128-CCM, AES-192-CCM, AES-256-CCM]; [AES-GCM: AES-128-GCM, AES-192-GCM, AES-256-GCM]; [AES-XTS: AES-128-XTS, AES-256-XTS]; [AES-KW, KWP: AES-128-WRAP, AES-256-WRAP]				
Key derivation (Perform approved security functions)	Derive keying material	[PBKDF: PBKDF2, (SHA-1, SHA2-224,	KAS_SS; flags	Status return; KD_DKM	Key Derivation	Crypto Officer - KAS_SS: W,E -

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512)]; [TLS1-PRF, (SHA2-256, SHA2-384, SHA2-512)]; [TLS13-KDF, (SHA2-256, SHA2-384)]; [X963-KDF, (SHA2-224, SHA2-256, SHA2-384, SHA2-512)]; [X942KDF-ASNI, (SHA1, SHA2-224, SHA2-				KD_DKM: G,R - KTS_SS: W,E - PBKDF Password: W,E,Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512)]; [NIST SP 800-108r1 KDF KMAC: KBKDF, (KMAC-128, KMAC-256)]; [NIST SP 800-108r1 KDF: KBKDF, MAC: CMAC, Cipher: AES-128-CBC, AES-192-CBC, AES-256-CBC, MAC: HMAC-SHA1, HMAC-				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512]; [KDF SSH: SSHKDF, (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512)];				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		[OneStep KDF: SSKDF, (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, HMAC-SHA1, HMAC-SHA2-224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, SHA3-224, SHA3-256,				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		SHA3-384, SHA3-512, KMAC-128, KMAC-256)]; [TwoStep KDF: HKDF, MAC: HMAC, (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512]; [HKDF: HKDF, MAC: HMAC, (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512];				
Key exchange (Perform approved security functions)	Perform key agreement primitives on behalf of the calling process (does not establish keys into the module)	[KAS-FFC-SSC: DHX]; [KAS-ECC-SSC: EC]	Key structs (KAS_Private and KAS_Public); flags	Status return; KAS_SS	KAS-1 KAS-2 KAS-3 KAS ECC CDH Component	Crypto Officer - KAS_Private: E - KAS_Public: E - KAS_SS: G
Key management (Perform approved security functions)	Generate asymmetric key pairs	[SafePrimes: DHX]; [RSA KeyGen: RSA, (2048, 3072, 4096)]; [ECDSA KeyGen: EC]; [DSA KeyGen: DSA, (L=2048, N=28, 32), (L=3072, N=32)]	ECDSA: curve identifier. DSA/RSA: modulus size	Status return; Key struct (GKP_Private, GKP_Public)	Asymmetric Key Pair Generation	Crypto Officer - GKP_Private: G - GKP_Public: G
Message authentication (Perform approved	Generate or verify data integrity. (CSPs are passed in by	[HMAC: HMAC-SHA1, HMAC-SHA2-	KH_Key	Status return; Tag value	Keyed Hash Cryptographic Key	Crypto Officer - KH_Key: E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
security functions)	the calling process or generated within the module)	224, HMAC-SHA2-256, HMAC-SHA2-384, HMAC-SHA2-512, HMAC-SHA2-512/224, HMAC-SHA2-512/256, HMAC-SHA3-224, HMAC-SHA3-256, HMAC-SHA3-384, HMAC-SHA3-512]; [CMAC]; [KMAC: KMAC-128, KMAC-256]; [GMAC: AES-128-GCM, AES-192-GCM, AES-256-GCM]			Generation (CKG)	
Message digest (Perform approved	Generate a message digest	[SHA-1, SHA2-224, SHA2-	Message ; flags	Status return; Hash value	Message Digest	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
security functions)		256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256, SHA3-224, SHA3-256, SHA3-384, SHA3-512, SHAKE-128, SHAKE-256]				
Random (Perform approved security functions)	Generate random bits using the DRBG	[Hash DRBG: HASH-DRBG, (SHA1, SHA2-256, SHA2-512)]; [HMAC-DRBG, (SHA1, SHA2-256, SHA2-512)]; [CTR-DRBG, (AES-128-CTR, AES-192-CTR, AES-256-CTR)]	DRBG struct (RBG State); DRBG_EI	Status return; Random value	Random Number Generation	Crypto Officer - DRBG_EI: E - DRBG_Seed: E - DRBG_State: E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Signature (Perform approved security functions)	Generate or verify digital signatures (SSPs are passed in by the calling process)	[RSA SigGen: RSA, (2048, 3072, 4096), (SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256)]; [RSA SigVer: RSA, (1024, 2048, 3072, 4096), (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256)]; [RSA Signature Primitive: RSA, 2048, hash algorithm: (null)];	Sign: Key struct (DS_SGK); message; Verify: signature value; Key struct (DS_SVK); flags; sizes	Status return; Signature value	RSA Digital Signature Generation and Verification ECDSA Signature Generation and Signature Verification DSA Digital Signature Generation and Verification RSA Signature Primitive	Crypto Officer - DS_SGK: E - DS_SVK: E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		[ECDSA SigGen: EC, (SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA3-224, SHA3-256, SHA3-384, SHA3-512)]; [ECDSA SigVer: EC, (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256)]; [ECDSA SigGen Component]: EC, hash: (null)]; [DSA, PQGGen: DSA, (L= 2048, N=28,				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256), (L=2048, 3072, N=32, =SHA2-256, SHA2-384, SHA2-512, SHA2-512/256)]; [DSA PQGVer: DSA, N=20 bytes, 28 bytes, 32 bytes]; [DSA, SigGen: DSA, (L= 2048, 3072), (N=28, 32), (SHA2-224, SHA2-256, SHA2-384, SHA2-				

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		512, SHA2-512/224, SHA2-512/256)]; [DSA, SigVer: DSA, (L=1024, N=20), (L=2048, N=28, 32), (L=3072, N=28, 32), (SHA1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256)]				
Zeroise (Perform zeroisation)	•The core Teardown operation zeroizes all Module scope SSPs •Call stack cleanup is the duty of the application •Restarting the general-purpose computer clears all SSPs in RAM •OPENSSL_cleanse	ZERO_OK	Memory pointer	Void	None	Crypto Officer - DS_SGK: Z - DS_SVK: Z - GKP_Private: Z - GKP_Public: Z - KAS_Private: Z -

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	provides zeroisation of SSPs managed by the caller; See the notes below this table for additional explanation					KAS_Public: Z - KAS_SS: Z - KD_DKM: Z - KH_Key: Z - KTS_KDK: Z - KTS_KEK: Z - KTS_SS: Z - DRBG_EI: Z - DRBG_Seed: Z - DRBG_State: Z - SC_EDK: Z - Software Integrity key: Z

Table 13: Approved Services

Note: The Indicators in Table 12 above follow the format:

[Algorithm name: Indicator 1, Indicator 2, etc.]

where Indicator 1 is an algorithm identifier and Indicators 2, 3 etc. depending on the algorithm are the specifics i.e. modes/supported curves/SafePrime groups/PRFs, etc.) per algorithm.

Each combination of the Indicator 1 along with Indicators 2, 3, etc. in the comma separated list can be observed when the corresponding modes/curves/SafePrimes, PRFs etc. are invoked for

a given algorithm in the context of a given service. The service indicators must be requested by the calling applications as by calling the following EVP APIs of the module in the context of each service:

- Hash (SHA) algorithms:
  - EVP_MD_get0_name
- Symmetric encryption algorithms (all AES modes, expect CMAC):
  - EVP_CIPHER_get0_name
- MAC algorithms (KMAC, HMAC, CMAC):
  - EVP_MAC_get0_name
- Key Derivation algorithms (KDFs):
  - EVP_KDF_get0_name used in conjunction with either of
    - EVP_MAC_get0_name (for a MAC as the PRF)
    - EVP_MD_get0_name (for Hash as the PRF)
- Key Exchange (KAS-ECC-SSC, KAS-FFC-SSC, KAS-IFC, SafePrimes), Key Generation (RSA, ECDSA, DSA):
  - EVP_PKEY_get0_type_name for ECDSA/KAS-ECC-SSC/KAS-IFC/RSA/DSA/KAS-FFC-SSC
  - EVP_PKEY_get_bits for RSA/DSA/IFC modulus size
  - EVP_PKEY_get_bn_param for DSA N value.
- Signature Generation/Verification (RSA, ECDSA, DSA):
  - EVP_PKEY_get0_type_name used in conjunction with EVP_MD_get0_name
  - EVP_PKEY_get_bits for RSA/DSA modulus size
  - EVP_PKEY_get_bn_param for DSA N value.
- ECDSA Signature Generation Component:
  - EVP_PKEY_get0_type_name and
  - EVP_MD_get0_name (returns null to indicate that a hash is not used)
- Random bit generators:
  - EVP_RAND_get0_name used in conjunction with
    - EVP_MD_get0_name for Hash and HMAC DRBG or
    - EVP_CIPHER_get0_name for Counter DRBG.
- Key Transport (OEAP):
  - EVP_PKEY_get0_type_name used in conjunction with
    - EVP_PKEY_get_bits for RSA modulus size and
    - EVP_PKEY_CTX_get_rsa_padding for padding (returns 4 to indicate OEAP).
- RSA Signature Primitive:
  - EVP_MD_get0_name and
  - EVP_PKEY_get_bits for RSA modulus size and
  - EVP_MD_get0_name (returns null to indicate that a hash is not used)
- DSA PQGGen:
  - EVP_PKEY_get_bits for L and
  - EVP_PKEY_get_bn_param for N
- DSA PQGVer:

- EVP_PKEY_get_bn_param for N

The OpenSSL toolkit *OSSL_PROVIDER_get_params* function when called with the Module's global handle and a pointer to a parameter structure (initialized using *provider_gettable_params* or the equivalent), can be used to retrieve the current status of the Module as well as the name and version; this information correlates to the validation listing. A 1 value returned in status indicates the Module is running without error (FIPS_OK); a 0 return indicates an error (with additional error details indicated as described in the release specific API documentation). Services are only operational in the running state. Any attempts to access services in any other state will result in an error being returned. If the integrity test or any CAST fails then any attempt to access any service will result in an error being returned.

Table 12 describes Module service access to SSPs; '--' indicates the cell is intentionally empty, not applicable or not relevant. The following annotations indicate the type of access by the Module service:
● **G = Generate**: The Module generates or derives the SSP.
● **R = Read**: The SSP is read from the Module (e.g. the SSP is output).
● **W = Write**: The SSP is updated, imported, or written to the Module.
● **E = Execute**: The Module uses the SSP in performing a cryptographic operation.
● **Z = Zeroise**: The Module zeroises the SSP.

Regarding the Indicator of approved security services, the Module conforms to [FIPS140-3_IG] 2.4.C *Approved Security Service Indicator*, similar to example 2. The Module's name and version parameters (as cited in Section 2) along with the Module's internal indicators of the security-check and conditional-errors settings are used to confirm the Module is the validated Module.

Each service provides context sensitive status responses as described in the OpenSSL 3 API manual pages; generally, functions of return type int return the value 1 for success with other error codes as appropriate for the call (described in API documentation).

Note that the caller provides the KAS_Private and KAS_Public keys for shared secret computation; the caller's exchange and assurance of PSPs with the remote participant is outside the scope of the Module.

All CSPs are zeroized (overwritten with 0s) when they are no longer needed:
● Temporary copies of CSPs are zeroised within the relevant function for the scope within which they are used.
● CSPs with a lifetime associated with an OpenSSL object (e.g., EVP_PKEY) will be zeroized when reinitialized.
● CSPs with a lifetime associated with the Module are zeroised on Module uninstantiation (the Teardown operation).
● The *OPENSSL_cleanse* function is used to zeroise CSPs owned by the caller.

4.4 Non-Approved Services

Name	Description	Algorithms	Role
Signature	Generate or verify digital signatures (SSPs are passed in by the calling process)	Ed448 Ed25519 FIPS 186-2 RSA SigGen/SigVer	Crypto Officer
Key Exchange	Perform key agreement primitives on behalf of the calling process (does not establish keys into the module)	X448 X25519	Crypto Officer
Cipher (Encryption/Decryption)	Encrypt or decrypt data (CSPs are passed in by the calling process)	Triple-DES	Crypto Officer
ECDSA SigVer Component	Verify ECDSA digital signatures (SSPs are passed in by the calling process)	ECDSA SigVer Component	Crypto Officer
Key Derivation	Derive keys (key derivation key passed in by the calling process)	X942KDF-CONCAT X963KDF HKDF OneStep KDF	Crypto Officer
Key Generation	Generate RSA public/private key pair per FIPS 186-2	FIPS 186-2 RSA KeyGen	Crypto Officer
Keyed Hash	Generate HMAC using key length less than 112 bits	HMAC	Crypto Officer
Random	Generate random bits using the non-approved Hash and HMAC DRBGs with PRFs SHA2-224, SHA2-384, SHA2-512/224 and SHA2-512/256	Hash and HMAC DRBG	Crypto Officer

Table 14: Non-Approved Services

4.5 External Software/Firmware Loaded

The module does not support loading of any additional software.

4.6 Bypass Actions and Status

The module does not support bypass.

4.7 Cryptographic Output Actions and Status

The module does not support self-initiated cryptographic output.

5 Software/Firmware Security

5.1 Integrity Techniques

The Module uses HMAC-SHA2-256 as the approved integrity technique; the file fipsmodule.cnf contains the integrity reference value. The HMAC key used for the integrity test is considered a non-SSP. The HMAC-SHA2-256 CAST is performed prior to the software integrity test. The Module is provided in an executable form (as fips.so shared object for use in Linux environments, fips.dylib for use in Mac environments and fips.dll for use in Windows environments). The module does not support loading of any additional software.

5.2 Initiate on Demand

The operator can initiate the integrity test on demand by calling *fips_self_test* (invoked using *OSSL_PROVIDER_self_test* called with the Module's global handle) or reloading the Module.

5.3 Open-Source Parameters

In accordance with [ISO19790] Annex B, as the Module is open source, the tools used to build the Module as tested are:

- gcc version 9.3.0
- perl v5.30.0
- gnu make v4.2.1

Compilers Used for Each Operational Environment
The specific compilers used to generate the Module for the respective operational environments are listed below:

- Ubuntu Linux 22.04.1 Server: gcc 11.2.0
- Debian 11.5: gcc 10.2.1
- FreeBSD 13.1: clang 11.0.1
- Windows 10: Visual Studio 2019
- macOS 11.5.2 (M1): clang 12.0.5
- macOS 11.5.2 (i7): clang 12.0.5

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied :

The operational environment for the Module is modifiable as it runs in General Purpose Computers (GPC). The Module conforms to [FIPS140-3_IG] 2.3.C Processor Algorithm Accelerators (PAA) and Processor Algorithm Implementation (PAI). The AES-NI functions are identified by [FIPS140-3_IG] 2.3.C as a known PAA.

6.2 Configuration Settings and Restrictions

Table 3 lists the operational environments on which the Module was tested; no operational environment restrictions are required for operation in the approved mode.

All conditions for operation of the Module in the approved mode are given in Section 2.

7 Physical Security

Physical Security requirements are not applicable for this software Module.

8 Non-Invasive Security

In accordance with current CMVP policy, Non-Invasive Security is not applicable.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Temporary, plaintext storage	Dynamic
Stored in the module's configuration file	Persistent, plaintext storage	Static

Table 15: Storage Areas

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
CALL STACK (API) INPUT PARAMETERS	Calling application	Module	Plaintext	Manual	Electronic	
CALL STACK (API) OUTPUT PARAMETERS	Module	Calling application	Plaintext	Manual	Electronic	
Stored at manufacture	Manufacturer	Stored in the module's configuration file	Plaintext	N/A	N/A	

Table 16: SSP Input-Output Methods

The module is complaint with FIPS 140-3 IG 9.5.A MD/EE (CM Software to/from App via TOEPP Path).

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
OPENSSL_cleanse	Zeroisation of SSPs managed by the caller	The OPENSSL_cleanse provides zeroisation of SSPs managed by the caller	Module initiated
cleared after use	Temporary copies of CSPs are zeroised within the relevant function for the scope within which they are used	CSPs with a lifetime associated with an OpenSSL object will be zeroized when reinitialized	Module initiated
Teardown	This operation triggers Module uninstantiation	CSPs with a lifetime associated with the Module are zeroised on Module uninstantiation	Operator initiated
Restarting the general-purpose computer	RAM (memory) is used for temporary storage of SSPs	Restarting the general-purpose computer clears all SSPs in RAM	Operator initiated

Table 17: SSP Zeroization Methods

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
DS_SGK	Private key for signature generation	RSA: 2048, 3072 and 4096 bits DSA: 2048 and 3072 bits ECDSA: B-233, K-233, P-224; B-283, K-283, P-256; B-409, K-409, P-384; B-571, K-571, P-521 - RSA: 112, 128 or 152 DSA: 112 or 128 ECDSA: 112, 128, 192, 521	Private key - CSP			RSA Digital Signature Generation and Verification ECDSA Signature Generation and Signature Verification DSA Digital Signature Generation and Verification

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						RSA Signature Primitive
DS_SVK	Public key for signature verification	RSA: 1024, 2048, 3072 and 4096 bits DSA: 1024, 2048 and 3072 bits ECDSA: ECDSA: B-233, K-233, P-224; B-283, K-283, P-256; B-409, K-409, P-384; B-571, K-571, P-521 - RSA: 80, 112, 128 or 152 DSA: 80, 112 or 128 ECDSA: 112, 128, 192, 256	Public key - PSP			RSA Digital Signature Generation and Verification ECDSA Signature Generation and Signature Verification DSA Digital Signature Generation and Verification
GKP_Private	Key pair (Private: DS_SGK, Public: DS_SVK) generated per caller request; the keypair purpose is unspecified	RSA: 2048, 3072, 4096 bits DSA: 2048 and 3072 bits ECDSA: ECDSA: B-233, K-233, P-224; B-283, K-283, P-256; B-409, K-409, P-384; B-571, K-571, P-521 - RSA: 112, 128 or 152 DSA: 112 or 128 ECDSA: 112, 128, 192, 256	Private key - CSP	Asymmetric Key Pair Generation Random Number Generation		

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
GKP_Public	Key pair (Private: GPK_Private, Public: GPK_Public) generated per caller request; the keypair purpose is unspecified	RSA: 2048, 3072, 4096 bits DSA: 2048 and 3072 bits ECDSA: ECDSA: B-233, K-233, P-224; B-283, K-283, P-256; B-409, K-409, P-384; B-571, K-571, P-521 - RSA: 112, 128 or 152 DSA: 112 or 128 ECDSA: 112, 128, 192, 256	Public key - PSP	Asymmetric Key Pair Generation Random Number Generation		
KAS_Private	Key pair component provided by the local participant, used for Diffie-Hellman shared secret generation	FFC: FB, FC, MODP2048, ffdhe2048, MODP3072, ffdhe3072, MODP4096, ffdhe4096, MODP6144, ffdhe6144, MODP8192, ffdhe 8192 ECC: B-233, K-233, P-224, B-283, K-283, P-256, B-409, K-409, P-384, B-571, K-571, P-521, IFC: k=2048, 3072, 4096, 6144, 8192 bits - FFC: between 112 and 200 ECC: 112,	Private key - CSP	Asymmetric Key Pair Generation Random Number Generation		KAS-1 KAS-2 KAS-3

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		128, 192, 256 IFC [SP800-56Br2]: 112, 128				
KAS_Public	Key pair component provided by the local participant, used for Diffie-Hellman shared secret generation	FFC: FB, FC, MODP2048, ffdhe2048, MODP3072, ffdhe3072, MODP4096, ffdhe4096, MODP6144, ffdhe6144, MODP8192, ffdhe 8192 ECC: B-233, K-233, P-224, B-283, K-283, P-256, B-409, K-409, P-384, B-571, K-571, P-521, IFC: k=2048, 3072, 4096, 6144, 8192 bits - FFC: between 112 and 200 ECC: 112, 128, 192, 256 IFC [SP800-56Br2]: 112, 128	Public key - PSP	Asymmetric Key Pair Generation Random Number Generation		KAS-1 KAS-2 KAS-3
KAS_SS	Shared secret calculation; z output value is expected to be used by a KDF	FFC: FB, FC, MODP2048, ffdhe2048, MODP3072, ffdhe3072, MODP4096, ffdhe4096, MODP6144, ffdhe6144, MODP8192, ffdhe 8192	Shared secret - CSP		KAS-1 KAS-2 KAS-3	

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		ECC: B-233, K-233, P-224, B-283, K-283, P-256, B-409, K-409, P-384, B-571, K-571, P-521, IFC: k=2048, 3072, 4096, 6144, 8192 bits - FFC: between 112 and 200 ECC: 112, 128, 192, 256 IFC: 112, 128				
KD_DKM	Key Derivation derived keying material	HMAC PRF: 160, 224, 256, 384, 512 - HMAC PRF: 160, 224, 256, 384, 512	Derived Keying Material - CSP		Key Derivation	
KH_Key	Keyed Hash key	CMAC: 128, 192, 256 GMAC: 128, 192, 256 HMAC: 160, 256, 512. KMAC: 128, 256 - CMAC: 128, 192, 256 GMAC: 128, 192, 256 HMAC: 160, 256, 512. KMAC: 128, 256	Symmetric key - CSP	Random Number Generation Cryptographic Key Generation (CKG)		Keyed Hash KTS-2
KTS_KDK	Private (KDK) component of an RSA key pair used for	2048, 3072, 4096 and 6144 bits - 112, 128, 152, 176	Private key - CSP			KTS-4

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
	[SP800-56Br2] RSA key transport					
KTS_KEK	Public (KEK) component of an RSA key pair used for [SP800-56Br2] RSA key transport	2048, 3072, 4096 and 6144 bits - 112, 128, 152, 176	Public key - PSP			KTS-4
KTS_SS	The RSA key transport shared secret	2048, 3072, 4096 and 6144 bits - 112, 128, 152, 176	Shared secret - CSP			KTS-4
DRBG_EI	Entropy input from an external source used for DRBG seeding	128 – 256 bits - 128 – 256 bits	Entropy input - CSP			Random Number Generation
DRBG_Seed	Seed generated from the entropy input for the DRBG	128 – 256 bits - 128 – 256 bits	DRBG seed - CSP			Random Number Generation
DRBG_State	Hash DRBG: V and C. HMAC DRBG: V and Key CTR DRBG: V and Key	Hash DRBG: 160, 224, 256, 384, 512 HMAC DRBG: 160, 224, 256, 384, 512. CTR DRBG: 128, 192, 256 - Hash DRBG: 160, 224, 256, 384, 512 HMAC DRBG: 160,	DRBG state - CSP	Random Number Generation		Random Number Generation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		224, 256, 384, 512. CTR DRBG: 128, 192, 256				
SC_EDK	AES key used for symmetric encryption and decryption (including use in key wrapping)	AES: 128, 192, 256 AES CCM: 128, 192, 256 AES GCM: 128, 192, 256 AES XTS: 128, 256. - AES: 128, 192, 256 AES CCM: 128, 192, 256 AES GCM: 128, 192, 256 AES XTS: 128, 256	Symmetric key - CSP	Random Number Generation Cryptographic Key Generation (CKG) Cryptographic Key Generation (CKG) - AES XTS		Symmetric Encryption and Decryption KTS-1 KTS-2 KTS-3
PBKDF Password	Input provided to the PBKDF	Recommended size is greater than 10 characters for passwords and greater than 20 characters for passphrases - 112 bits or greater	Symmetric key - CSP			Key Derivation
Software Integrity key	HMAC-SHA2-256 key used to perform the Software Integrity Test	256 bits - 256 bits	256 bits - Neither			Software Integrity Test

Table 18: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
DS_SGK	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	DS_SVK:Paired With
DS_SVK	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	DS_SGK:Paired With
GKP_Private	CALL STACK (API) OUTPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	GKP_Public:Paired With
GKP_Public	CALL STACK (API) OUTPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	GKP_Private:Paired With
KAS_Private	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	KAS_Public:Paired With
KAS_Public	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the	KAS_Private:Paired With

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
				general-purpose computer	
KAS_SS		RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	KAS_Private:Established using KAS_Public:Established using
KD_DKM	CALL STACK (API) OUTPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	
KH_Key	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	
KTS_KDK	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	KTS_KEK:Paired With
KTS_KEK	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	KTS_KDK:Paired With
KTS_SS	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse Teardown Restarting the	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
	CALL STACK (API) OUTPUT PARAMETERS			general-purpose computer	
DRBG_EI	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	DRBG_Seed:Used to derive
DRBG_Seed		RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	DRBG_EI:Derived From
DRBG_State		RAM:Plaintext	Until power-cycling of the underlying host platform	Teardown Restarting the general-purpose computer	DRBG_Seed:Derived From
SC_EDK	CALL STACK (API) INPUT PARAMETERS CALL STACK (API) OUTPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the general-purpose computer	
PBKDF Password	CALL STACK (API) INPUT PARAMETERS	RAM:Plaintext	cleared after use	OPENSSL_cleanse cleared after use Teardown Restarting the	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
				general-purpose computer	
Software Integrity key	Stored at manufacture	Stored in the module's configuration file :Plaintext	Until teardown operation is performed	Teardown	

Table 19: SSP Table 2

All SSPs used by the Module are described in this section, arranged for consistency with Table 12; '--' indicates the cell is intentionally empty, not applicable, or not relevant.

Keys used for CASTs and the temporary value used in the integrity test are not SSPs; however, the latter is deleted after use as required by AS05.10. Equivalent strength is given for each key or algorithm type (as some algorithms do not use or produce keys).

The Module maintains only the DRBG CSPs used for key generation as persistent CSPs; these are used exclusively for approved services.

DRBG outputs are used internally to the Module for asymmetric key pair generation and used by calling applications to generate a random value (potentially for use as a symmetric key).

The Module:
• Produces random values in accordance with [SP800-133r2] Section 4, in that the DRBG output is provided directly as the random output.
• SSPs used with symmetric key algorithms are provided by the calling application.
• Produces asymmetric keys in accordance with [SP800-133r2] Section 5, in that all asymmetric keys generated by the module (the Key management service) provide the output of the approved key generation algorithm with no post-processing or manipulation of the generated key pairs. As noted in the previous item, random values used in the asymmetric key generation algorithms are direct outputs of the DRBG. Keys produced by the module use an internal Counter DRBG for which the minimum key size and equivalent security strength is 128 bits.
• Supports direct generation of symmetric keys in accordance with [SP800-133r2] Section 6.1 and symmetric key derivation in accordance with [SP800-133r2] Section 6.2, using the approved and CAVP listed KDF algorithms. AES-KTS keys are generated in accordance with [SP800-133r2] Section 6.3.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A3548)	Key Length: 256 bits	KAT	SW/FW Integrity	Success: All self-tests passed (as expected)	MAC (HMAC-SHA2-256, A3548)

Table 20: Pre-Operational Self-Tests

The module is complaint with FIPS 140-3 IG 10.2.A in that it performs a self-test, a Known Answer Test (KAT) for the HMAC-SHA2-256 algorithm.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A3548)	Key Length: 128 bits	KAT	CAST	FIPS_OK	Decrypt	On reloading the module
AES-GCM (A3548)	Key Length: 256 bits	KAT	CAST	FIPS_OK	Encrypt	On reloading the module
AES-GCM (A3548)	Key Length: 256 bits	KAT	CAST	FIPS_OK	Decrypt	On reloading the module
Counter DRBG (A3548)	AES CTR (128 bits) with derivation function	KAT	CAST	FIPS_OK	Generate, Reseed, Instantiate functions	On reloading the module
DSA SigGen (FIPS186-4) (A3548)	Modulus: 2048 bits; Hash: SHA2-384	KAT	CAST	FIPS_OK	Sign	On reloading the module
DSA SigVer (FIPS186-4) (A3548)	Modulus: 2048 bits; Hash: SHA2-384	KAT	CAST	FIPS_OK	Verify	On reloading the module
ECDSA SigGen (FIPS186-4) (A3548)	Curve: P-224; Hash: SHA2-512	KAT	CAST	FIPS_OK	Sign	On reloading the module
ECDSA SigVer (FIPS186-4) (A3548)	Curve: P-224; Hash: SHA2-512	KAT	CAST	FIPS_OK	Verify	On reloading the module
Hash DRBG (A3548)	PRF: SHA2-256	KAT	CAST	FIPS_OK	Generate, Reseed, Instantiate functions	On reloading the module

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
HMAC DRBG (A3548)	PRF: HMAC-SHA-1	KAT	CAST	FIPS_OK	Generate, Reseed, Instantiate functions	On reloading the module
HMAC-SHA2-256 (A3548)	PRF: SHA2-256	KAT	CAST	FIPS_OK	HMAC tag Generation	Performed prior to the software integrity test
KAS-ECC-SSC Sp800-56Ar3 (A3548)	Scheme: Ephemeral Unified, Curve: P-256	KAT	CAST	FIPS_OK	Key Agreement - Shared Secret Computation	On reloading the module
KAS-FFC-SSC Sp800-56Ar3 (A3548)	Scheme: dhEphem; Modulus: L = 2048 bits, N = 256 bit	KAT	CAST	FIPS_OK	Key Agreement - Shared Secret Computation	On reloading the module
KAS-IFC-SSC (A3548)	Schemes: Basic, CRT, Modulus: L = 2048 bits	KAT	CAST	FIPS_OK	Key Agreement - Shared Secret Computation	On reloading the module
KDF SP800-108 (A3548)	Mode: Counter, PRF: HMAC-SHA2-256	KAT	CAST	FIPS_OK	Counter Mode (HMAC-SHA2-256).	On reloading the module
KDA OneStep SP800-56Cr2 (A3548)	Auxiliary Function, H = SHA2-224	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
KDA TwoStep SP800-56Cr2 (A3548)	Auxiliary Function, H = HMAC-SHA2-256	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
KTS-IFC (A3548)	Schemes: Basic Modulus: L = 2048 bits	KAT	CAST	FIPS_OK	Encrypt	On reloading the module
KTS-IFC (A3548)	Schemes: Basic, CRT, Modulus: L = 2048 bits	KAT	CAST	FIPS_OK	Decrypt	On reloading the module

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
PBKDF (A3548)	Derivation of the Master Key (MK), PRF: SHA2-256	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
RSA SigGen (FIPS186-4) (A3548)	Scheme: PKCS#1, Modulus: L = 2048, Hash: SHA2-256	KAT	CAST	FIPS_OK	Sign	On reloading the module
RSA SigVer (FIPS186-4) (A3548)	Scheme: PKCS#1, Modulus: L = 2048, Hash: SHA2-256	KAT	CAST	FIPS_OK	Verify	On reloading the module
SHA-1 (A3548)	SHA-1	KAT	CAST	FIPS_OK	Hash	On reloading the module
SHA2-512 (A3548)	SHA2-512	KAT	CAST	FIPS_OK	Hash	On reloading the module
SHA3-256 (A3548)	SHA3-256	KAT	CAST	FIPS_OK	Hash	On reloading the module
KDF ANS 9.42 (A3548)	PRFs: AES KW (128 bits), SHA-1	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
KDF ANS 9.63 (A3548)	PRF: SHA2-256	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
KDF SSH (A3548)	PRF: SHA-1	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
TLS v1.2 KDF RFC7627 (A3548)	PRF: SHA2-256	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
TLS v1.3 KDF (A3548)	PRF: SHA2-256	KAT	CAST	FIPS_OK	Key Derivation	On reloading the module
RSA KeyGen (FIPS186-4) (A3548)	Performed post key generation	PCT	PCT	FIPS_OK	Key Generation	On generating keys for Key Transport (KTS IFC)/Key Agreement (KAS IFC)/Signature Generation/Signature Verification

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA KeyGen (FIPS186-4) (A3548)	Performed post key generation	PCT	PCT	FIPS_OK	Key Generation	On generating keys for Key Agreement (KAS ECC)/Signature Generation/Signature Verification
DSA KeyGen (FIPS186-4) (A3548)	Performed post key generation	PCT	PCT	FIPS_OK	Key Generation	On generating keys for Key Agreement (KAS FFC)/Signature Generation/Signature Verification
ECDSA SigGen (FIPS186-4) (A3548)	Curve: K-233; Hash: SHA2-512	KAT	CAST	FIPS_OK	Sign	On reloading the module
ECDSA SigVer (FIPS186-4) (A3548)	Curve: K-233; Hash: SHA2-512	KAT	CAST	FIPS_OK	Verify	On reloading the module

Table 21: Conditional Self-Tests

Each time the Module is powered up it tests that the cryptographic algorithms still operate correctly and that sensitive data has not been damaged. On instantiation, the Module performs the pre-operational self-tests and all CASTs listed above. All KATs must complete successfully prior to any other use of cryptography by the Module.

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A3548)	KAT	SW/FW Integrity	On Demand	Manually by reloading the module or calling the fips_self_test function

Table 22: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-ECB (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-GCM (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
AES-GCM (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
Counter DRBG (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
DSA SigGen (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
DSA SigVer (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
ECDSA SigGen (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
ECDSA SigVer (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
Hash DRBG (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC DRBG (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
HMAC-SHA2-256 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KAS-ECC-SSC Sp800-56Ar3 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KAS-FFC-SSC Sp800-56Ar3 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KAS-IFC-SSC (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KDF SP800-108 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KDA OneStep SP800-56Cr2 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KDA TwoStep SP800-56Cr2 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KTS-IFC (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KTS-IFC (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
PBKDF (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
RSA SigGen (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
RSA SigVer (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
SHA-1 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
SHA2-512 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
SHA3-256 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF ANS 9.42 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KDF ANS 9.63 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
KDF SSH (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
TLS v1.2 KDF RFC7627 (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
TLS v1.3 KDF (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
RSA KeyGen (FIPS186-4) (A3548)	PCT	PCT	On Demand	On generation of keys
ECDSA KeyGen (FIPS186-4) (A3548)	PCT	PCT	On Demand	On generation of keys
DSA KeyGen (FIPS186-4) (A3548)	PCT	PCT	On Demand	On generation of keys
ECDSA SigGen (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or calling the fips_self_test function
ECDSA SigVer (FIPS186-4) (A3548)	KAT	CAST	On Demand	Manually by reloading the module or

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
				calling the fips_self_test function

Table 23: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
ERROR STATE	•The error state is persistent and no services are available •All attempts to use the Module's services result in the return of a non-zero error code, PROV_R_FIPS_MODULE_IN_ERROR_STATE	If one of the KATs or if the Software Integrity Test fails, the Module enters the self-test failure error state	To recover from an error state, reload the Module into memory	PROV_R_FIPS_MODULE_IN_ERROR_STATE

Table 24: Error States

10.5 Operator Initiation of Self-Tests

The operator can reload the module or the *fips_self_test* function (inclusive of software integrity verification) can also be called on demand, fulfilling AS05.11.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

The Module is provided to vendors who integrate it into their product, typically in a manufacturing environment, and is not provided directly to US or Canadian Federal agencies. Adherence to the instructions in this document maintains security throughout the distribution, build, installation and configuration processes.

An authorized Cryptographic Officer is required to perform these steps on each platform where it is intended to be used. The config file output contains information about the Module (such as the self-test status and the Module checksum) and must not be copied from one machine to another.

Crypto Officer Guidance

a. Installation and Usage Guidance

The Module is installed as part of the OpenSSL 3.1.2 library. The source distribution package is located at https://www.openssl.org/source/openssl-3.1.2.tar.gz.

The OpenSSL FIPS Provider can be installed on the Tested Configurations listed in Table 3 by performing the following steps:

1. Build and install OpenSSL 3.1.2 to the default location:

The OpenSSL FIPS Provider (i.e., the Module) does not get built and installed automatically. To install the Module automatically during the normal OpenSSL 3.1.2 installation process it must be enabled by configuring OpenSSL using the 'enable-fips' option.

Unix/Linux/macOS:

```
$ ./Configure enable-fips
$ make
$ make install
```

Windows:

```
$ perl Configure enable-fips
$ nmake
$ nmake install
```

The 'install_fips' make target can also be invoked explicitly to install the FIPS Provider independently, without installing the rest of OpenSSL:

```
$ make install_fips
```

Note: The instructions for building and installing OpenSSL 3.1.2 on other platforms can be found in the platform-specific guidance provided in INSTALL.md and README-FIPS.md in the OpenSSL 3.1.2 distribution package. Please see Appendix A for further information on porting the Module to platforms apart from the Tested Configurations in Table 3.

2. Verify the version:

```
$ openssl version -v
```

The Installation of the OpenSSL FIPS Provider that occurs as a result of Step 1 above ensures that the shared library and the configuration file containing information about the Module (e.g., the Module checksum) is copied to its installed location.

To install the configuration file to a non-default location, this can be achieved by running the 'fipsinstall' command line application manually:

```
$ openssl fipsinstall -pedantic
```

Please see fipsinstall.html /docs/man3.1/man1/openssl-fipsinstall.html for options supported for the ‘openssl fipsinstall’ command.

Note: The software integrity check (per Section 5 of this document) is performed using HMAC-SHA2-256 on the Module file to validate that the Module has not been modified. The integrity value is compared to a value written to the config file during installation.

b. CVEs

The publication of a CVE does not require immediate re-validation or maintenance in the CMVP process. The module may be updated in the field as needed depending on the severity or consequences of the CVE. The Module will be kept up to date with re-validation and maintenance as required, generally bundling fixes for known CVEs in a next release.

The OpenSSL organization maintains a Vulnerabilities page which describes known vulnerabilities and potential resolution. These are reported to the NVD, where they are independently assessed. The OpenSSL group publishes fixes for these vulnerabilities according to their triage process.

c. Miscellaneous

The module performs run-time checks related to enforcement of security parameters such as the minimum-security strength of keys, valid key sizes, and usage of approved curves. These checks shall not be disabled (by using OPENSSL_NO_FIPS_SECURITYCHECKS or any other method).

Validation of domain parameters prior to generating keys using functions provided by the module is the responsibility of the Cryptographic Officer and not enforced by the module itself.

11.2 Administrator Guidance

No additional guidance applies for the operation of the module apart from that specified in Sections 2, 3 of this document and other subsections under this section.

11.3 Non-Administrator Guidance

No additional guidance applies for the operation of the module apart from that specified in Sections 2, 3 of this document and other subsections under this section.

11.4 Design and Rules

No additional rules apply for the operation of the module apart from those specified in the remainder of this section and Section 2.4 of this document.

11.5 Maintenance Requirements

No maintenance requirements apply for operation of the module in the Approved/non-Approved modes as defined above.

11.6 End of Life

Module Sanitization and Destruction
Sanitization is defined in [ISO19790] as “... the process of removing sensitive information (e.g. SSPs, user data, etc.) from the module, so that it may either be distributed to other operators or disposed.”

The Module itself does not manage persistent SSPs, authentication data or any user data. The Module may be securely sanitized by deletion of the folder in which the Module was located.

There are no additional procedures required for secure destruction of the Module.

12 Mitigation of Other Attacks

12.1 Attack List

The Module implements mitigations for some types of attacks using the constant-time implementations and blinding.

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.