Cloud Linux Software, Inc. d/b/a TuxCare

TuxCare OpenSSL FIPS Provider

FIPS 140-3 Non-Proprietary Security Policy

Prepared by:

atsec information security corporation
4516 Seton Center Pkwy, Suite 250
Austin, TX 78759
www.atsec.com

Document version: 1.0
Last update: 2026-06-04

Table of Contents

List of Tables

List of Figures

1 General

1.1 Overview

This document is the non-proprietary FIPS 140-3 Security Policy for version 3.2.2-f9f9d133a30b6eb5 of the TuxCare OpenSSL FIPS Provider. It contains the security rules under which the module must operate and describes how this module meets the requirements as specified in FIPS PUB 140-3 (Federal Information Processing Standards Publication 140-3) for an overall Security Level 1 module.

This Non-Proprietary Security Policy may be reproduced and distributed, but only whole and intact and including this notice. Other documentation is proprietary to their authors.

1.2 Security Levels

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

Table 1: Security Levels

1.3 Additional Information

In preparing the Security Policy document, the laboratory formatted the vendor-supplied documentation for consolidation without altering the technical statements therein contained. The further refining of the Security Policy document was conducted iteratively throughout the conformance testing, wherein the Security Policy was submitted to the vendor, who would then edit, modify, and add technical contents. The vendor would also supply additional documentation, which the laboratory formatted into the existing Security Policy, and resubmitted to the vendor for their final editing.

2 Cryptographic Module Specification

2.1 Description

Purpose and Use:

The TuxCare OpenSSL FIPS Provider (hereafter referred to as "the module") is defined as a software module in a multi-chip standalone embodiment. It provides a C language application program interface (API) for use by other applications that require cryptographic functionality. The module consists of one software component, the "FIPS provider" i.e., fips.so, which implements the FIPS requirements, and the cryptographic functionality provided to the operator.

Module Type: Software

Module Embodiment: Multi-Chip Standalone

Cryptographic Boundary:

Figure 1 shows a block diagram that represents the design of the module when the module is operational and providing services to other user space applications.

The cryptographic boundary is represented by the component in the orange block, that is, the shared library implementing the FIPS provider (fips.so). The connecting lines indicate the flow of data between the cryptographic module and its operator application, through the logical interfaces defined in Section 3.

The components in white are only included in the diagram for informational purposes. They are not included in the cryptographic boundary (and therefore not part of the module's validation).

Tested Operational Environment's Physical Perimeter (TOEPP):

The TOEPP of the module is defined as the general-purpose computer on which the module is installed.

The PAA provided by the processor are located within the module's physical perimeter and outside of the module's cryptographic boundary.

The module makes use of an SP800-90B-compliant Entropy Source (described in Section 2.8) located within the TOEPP.

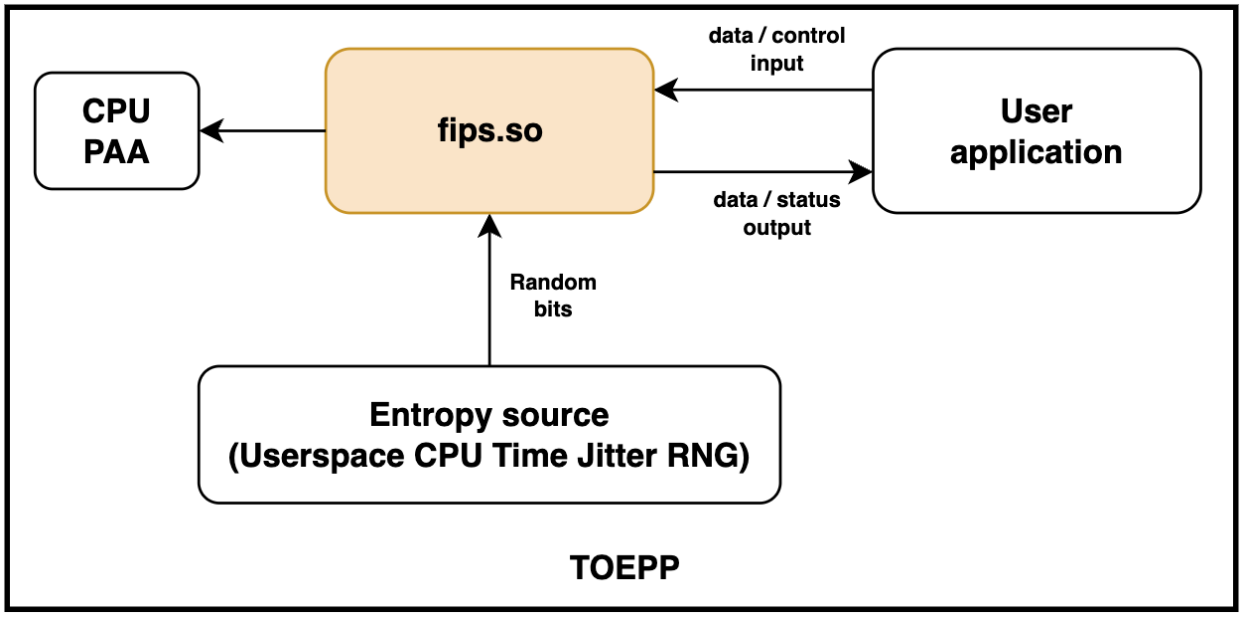

Figure 1: Block Diagram

2.2 Tested and Vendor Affirmed Module Version and Identification

Tested Module Identification – Hardware:

N/A for this module.

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

Package or File Name	Software/ Firmware Version	Features	Integrity Test
fips.so	3.2.2-f9f9d133a30b6eb5	N/A	HMAC-SHA2-256

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

Tested Module Identification – Hybrid Disjoint Hardware:

N/A for this module.

Tested Operational Environments - Software, Firmware, Hybrid:

Operating System	Hardware Platform	Processors	PAA/PAI	Hypervisor or Host OS	Version(s)
AlmaLinux OS 9.6	GIGABYTE E163-S30-AAG1	Intel® Xeon® Gold 5512U	Yes	N/A	3.2.2-f9f9d133a30b6eb5
AlmaLinux OS 9.6	GIGABYTE E163-S30-AAG1	Intel® Xeon® Gold 5512U	No	N/A	3.2.2-f9f9d133a30b6eb5

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

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

Operating System	Hardware Platform
Rocky Linux 9.6	GIGABYTE E163-S30-AAG1 on Intel® Xeon® Gold 5512U

Table 4: Vendor-Affirmed Operational Environments - Software, Firmware, Hybrid

CMVP makes no statement as to the correct operation of the module or the security strengths of the generated keys when so ported if the specific operational environment is not listed on the validation certificate.

2.3 Excluded Components

There are no components within the cryptographic boundary excluded from the FIPS 140-3 requirements.

2.4 Modes of Operation

Modes List and Description:

Mode Name	Description	Type	Status Indicator
Approved mode	Automatically entered whenever an approved service is requested	Approved	Equivalent to the indicator of the requested service. As described in Section 4.3.
Non-approved mode	Automatically entered whenever a non-approved service is requested	Non-Approved	Equivalent to the indicator of the requested service. As described in Section 4.4.

Table 5: Modes List and Description

Mode Change Instructions and Status:

The module automatically switches between the approved and non-approved modes depending on the services requested by the operator. The status indicator of the mode of operation is equivalent to the indicator of the service that was requested.

2.5 Algorithms

Approved Algorithms:

Algorithm	CAVP Cert	Properties	Reference
AES-CBC	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS1	A7094, A7095, A7099	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS2	A7094, A7095, A7099	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CBC-CS3	A7094, A7095, A7099	Direction - decrypt, encrypt Key Length - 128, 192, 256	SP 800-38A

Algorithm	CAVP Cert	Properties	Reference
AES-CCM	A7094, A7095, A7099	Key Length - 128, 192, 256	SP 800-38C
AES-CFB1	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB128	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CFB8	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-CMAC	A7094, A7095, A7099	Direction - Generation Key Length - 128, 192, 256	SP 800-38B
AES-CTR	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-ECB	A7094, A7095, A7099, A7115, A7118, A7122, A7123, A7124	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-GCM	A7100, A7105, A7106, A7107, A7108, A7109, A7110, A7127, A7128	Direction - Decrypt, Encrypt IV Generation - External, Internal Key Length - 128, 192, 256 IV Generation Mode - 8.2.2	SP 800-38D
AES-GMAC	A7100, A7105, A7106, A7107, A7108, A7109, A7110, A7127, A7128	Direction - Decrypt, Encrypt IV Generation - External Key Length - 128, 192, 256	SP 800-38D
AES-KW	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-KWP	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38F
AES-OFB	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 192, 256	SP 800-38A
AES-XTS Testing Revision 2.0	A7094, A7095, A7099	Direction - Decrypt, Encrypt Key Length - 128, 256	SP 800-38E
Counter DRBG	A7097	Prediction Resistance - No, Yes Mode - AES-128, AES-192, AES-256 Derivation Function Enabled - No, Yes	SP 800-90A Rev. 1

Algorithm	CAVP Cert	Properties	Reference
ECDSA KeyGen (FIPS186-5)	A7101, A7111, A7112, A7113, A7114	Curve - P-224, P-256, P-384, P-521 Secret Generation Mode - testing candidates	FIPS 186-5
ECDSA KeyVer (FIPS186-5)	A7101, A7111, A7112, A7113, A7114	Curve - P-224, P-256, P-384, P-521	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A7101, A7111, A7112, A7113, A7114	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Component - No, Yes	FIPS 186-5
ECDSA SigGen (FIPS186-5)	A7102	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 Component - No, Yes	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A7101, A7111, A7112, A7113, A7114	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256	FIPS 186-5
ECDSA SigVer (FIPS186-5)	A7102	Curve - P-224, P-256, P-384, P-521 Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512	FIPS 186-5
EDDSA KeyGen	A7098	Curve - ED-25519, ED-448	FIPS 186-5
EDDSA SigGen	A7098	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
EDDSA SigVer	A7098	Curve - ED-25519, ED-448 PreHash - Yes Pure - Yes	FIPS 186-5
Hash DRBG	A7097	Prediction Resistance - No, Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC DRBG	A7097	Prediction Resistance - No, Yes Mode - SHA-1, SHA2-256, SHA2-512	SP 800-90A Rev. 1
HMAC-SHA-1	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-224	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1

Algorithm	CAVP Cert	Properties	Reference
HMAC-SHA2-256	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-384	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/224	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA2-512/256	A7101, A7111, A7112, A7113, A7114	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-224	A7102	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-256	A7102	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-384	A7102	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
HMAC-SHA3-512	A7102	Key Length - Key Length: 112-524288 Increment 8	FIPS 198-1
KAS-ECC-SSC Sp800-56Ar3	A7101, A7111, A7112, A7113, A7114	Domain Parameter Generation Methods - P-224, P-256, P-384, P-521 Scheme - ephemeralUnified - KAS Role - initiator, responder	SP 800-56A Rev. 3
KAS-FFC-SSC Sp800-56Ar3	A7120	Domain Parameter Generation Methods - 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	A7101, A7111, A7112, A7113, A7114	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KAS1 -	SP 800-56A Rev. 3

Algorithm	CAVP Cert	Properties	Reference
		KAS Role - initiator, responder KAS2 - KAS Role - initiator, responder	
KDA HKDF SP800-56Cr2	A7096	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	A7121	Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-2048 Increment 8	SP 800-56C Rev. 2
KDA TwoStep SP800-56Cr2	A7121	MAC Salting Methods - default, random KDF Mode - feedback Derived Key Length - 2048 Shared Secret Length - Shared Secret Length: 224-2048 Increment 8	SP 800-56C Rev. 2
KDF ANS 9.42 (CVL)	A7101, A7111, A7112, A7113, A7114	KDF Type - DER Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Key Data Length - Key Data Length: 112-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.42 (CVL)	A7102	KDF Type - DER Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 112-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.63 (CVL)	A7101, A7111, A7112, A7113, A7114	Hash Algorithm - SHA2-224, SHA2-256, SHA2-384, SHA2-512, SHA2-512/224, SHA2-512/256 Key Data Length - Key Data Length: 128-4096 Increment 8	SP 800-135 Rev. 1
KDF ANS 9.63 (CVL)	A7102	Hash Algorithm - SHA3-224, SHA3-256, SHA3-384, SHA3-512 Key Data Length - Key Data Length: 128-4096 Increment 8	SP 800-135 Rev. 1
KDF SP800-108	A7119	KDF Mode - Counter, Feedback Supported Lengths - Supported Lengths: 112-4096 Increment 8	SP 800-108 Rev. 1
KDF SSH (CVL)	A7115, A7118, A7122, A7123, A7124	Cipher - AES-128, AES-192, AES-256 Hash Algorithm - SHA-1, SHA2-224, SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1

Algorithm	CAVP Cert	Properties	Reference
KTS-IFC	A7101, A7111, A7112, A7113, A7114	Modulo - 2048, 3072, 4096, 6144, 8192 Key Generation Methods - rsakpg1-basic, rsakpg1-crt, rsakpg1-prime-factor, rsakpg2-basic, rsakpg2-crt, rsakpg2-prime-factor Scheme - KTS-OAEP-basic - KAS Role - initiator, responder Key Transport Method - Key Length - 768	SP 800-56B Rev. 2
PBKDF	A7101, A7102, A7111, A7112, A7113, A7114	Iteration Count - Iteration Count: 1000-10000 Increment 1 Password Length - Password Length: 8-128 Increment 1	SP 800-132
RSA KeyGen (FIPS186-5)	A7101, A7111, A7112, A7113, A7114	Key Generation Mode - probableWithProbableAux Modulo - 2048, 3072, 4096, 6144, 8192 Primality Tests - 2powSecStr Private Key Format - standard	FIPS 186-5
RSA SigGen (FIPS186-5)	A7101, A7102, A7111, A7112, A7113, A7114	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
RSA SigVer (FIPS186-5)	A7101, A7102, A7111, A7112, A7113, A7114	Modulo - 2048, 3072, 4096 Signature Type - pkcs1v1.5, pss	FIPS 186-5
Safe Primes Key Generation	A7120	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	A7120	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	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-224	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-256	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-384	A7101, A7111, A7112, A7113, A7114	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	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/224	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA2-512/256	A7101, A7111, A7112, A7113, A7114	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 180-4
SHA3-224	A7102	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-256	A7102	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-384	A7102	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHA3-512	A7102	Message Length - Message Length: 0-65536 Increment 8 Large Message Sizes - 1, 2, 4, 8	FIPS 202
SHAKE-128	A7102	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
SHAKE-256	A7102	Output Length - Output Length: 16-65536 Increment 8	FIPS 202
TLS v1.2 KDF RFC7627 (CVL)	A7101, A7111, A7112, A7113, A7114	Hash Algorithm - SHA2-256, SHA2-384, SHA2-512	SP 800-135 Rev. 1
TLS v1.3 KDF (CVL)	A7096	HMAC Algorithm - SHA2-256, SHA2-384 KDF Running Modes - DHE, PSK, PSK-DHE	SP 800-135 Rev. 1

Table 6: Approved Algorithms

The above table lists all approved cryptographic algorithms of the module, including specific key lengths employed for approved services, and implemented modes or methods of operation of the algorithms.

Vendor-Affirmed Algorithms:

Name	Properties	Implementation	Reference
Asymmetric Cryptographic Key Generation (CKG)	Key Type:Asymmetric	N/A	SP 800-133r2, section 4, example 1

Table 7: Vendor-Affirmed Algorithms

Non-Approved, Allowed Algorithms:

N/A for this module.

Non-Approved, Allowed Algorithms with No Security Claimed:

N/A for this module.

Non-Approved, Not Allowed Algorithms:

Name	Use and Function
AES GCM (external IV)	Authentication Encryption
HMAC (< 112-bit keys)	Message Authentication Code
KBKDF, KDA OneStep, HKDF, ANS X9.42 KDF, ANS X9.63 KDF (< 112-bit keys)	Key Derivation
KDA OneStep (SHAKE128, SHAKE256)	Key Derivation
ANS X9.42 KDF (SHAKE128, SHAKE256)	Key Derivation
ANS X9.63 KDF (SHA-1, SHAKE128, SHAKE256)	Key Derivation
SSH KDF (SHA-512/224, SHA-512/256, SHA-3, SHAKE128, SHAKE256)	Key Derivation
TLS 1.2 KDF (SHA-1, SHA-224, SHA-512/224, SHA-512/256, SHA-3)	TLS Key Derivation
TLS 1.3 KDF (SHA-1, SHA-224, SHA-512, SHA-512/224, SHA-512/256, SHA-3)	TLS Key Derivation
PBKDF2 (< 8 characters password; < 128 salt length; < 1000 iterations; < 112-bit keys)	Password-based Key Derivation
RSA and ECDSA (pre-hashed message)	Signature generation; Signature verification
RSA-PSS (invalid salt length)	Signature generation; Signature verification

Table 8: Non-Approved, Not Allowed Algorithms

2.6 Security Function Implementations

Name	Type	Description	Properties	Algorithms
Symmetric Encryption with AES	BC-UnAuth	Symmetric encryption using AES		AES-CBC: (A7094, A7095, A7099) AES-CBC-CS1: (A7094, A7095, A7099) AES-CBC-CS2: (A7094, A7095, A7099) AES-CBC-CS3: (A7094, A7095, A7099) AES-CFB1: (A7094, A7095, A7099) AES-CFB128: (A7094, A7095, A7099) AES-CFB8: (A7094, A7095, A7099) AES-CTR: (A7094, A7095, A7099) AES-ECB: (A7094, A7095, A7099, A7115, A7118, A7122, A7123, A7124) AES-OFB: (A7094, A7095, A7099) AES-XTS Testing Revision 2.0: (A7094, A7095, A7099)
Symmetric Decryption with AES	BC-UnAuth	Symmetric decryption using AES		AES-CBC: (A7094, A7095, A7099) AES-CBC-CS1: (A7094, A7095, A7099) AES-CBC-CS2: (A7094, A7095, A7099) AES-CBC-CS3: (A7094, A7095, A7099) AES-CFB1: (A7094, A7095, A7099)

Name	Type	Description	Properties	Algorithms
				AES-CFB128: (A7094, A7095, A7099) AES-CFB8: (A7094, A7095, A7099) AES-CTR: (A7094, A7095, A7099) AES-ECB: (A7094, A7095, A7099, A7115, A7118, A7122, A7123, A7124) AES-OFB: (A7094, A7095, A7099) AES-XTS Testing Revision 2.0: (A7094, A7095, A7099)
Authenticated Encryption with AES	BC-Auth	Authenticated symmetric encryption		AES-CCM: (A7094, A7095, A7099) AES-GCM: (A7100, A7105, A7106, A7107, A7108, A7109, A7110, A7127, A7128) AES-KW: (A7094, A7095, A7099) AES-KWP: (A7094, A7095, A7099)
Authenticated Decryption with AES	BC-Auth	Authenticated symmetric decryption		AES-CCM: (A7094, A7095, A7099) AES-GCM: (A7100, A7105, A7106, A7107, A7108, A7109, A7110, A7127, A7128) AES-KW: (A7094, A7095, A7099) AES-KWP: (A7094, A7095, A7099)
Random Number Generation	DRBG	Random number generation		Counter DRBG: (A7097) Hash DRBG: (A7097)

Name	Type	Description	Properties	Algorithms
				HMAC DRBG: (A7097)
Message Digest	SHA XOF	Message digest		SHA-1: (A7101, A7111, A7112, A7113, A7114) SHA2-224: (A7101, A7111, A7112, A7113, A7114) SHA2-256: (A7101, A7111, A7112, A7113, A7114) SHA2-384: (A7101, A7111, A7112, A7113, A7114) SHA2-512: (A7101, A7111, A7112, A7113, A7114) SHA2-512/224: (A7101, A7111, A7112, A7113, A7114) SHA2-512/256: (A7101, A7111, A7112, A7113, A7114) SHA3-224: (A7102) SHA3-256: (A7102) SHA3-384: (A7102) SHA3-512: (A7102) SHAKE-128: (A7102) SHAKE-256: (A7102)
Message Authentication Code with AES	MAC	Message Authentication Code using AES		AES-CMAC: (A7094, A7095, A7099) AES-GMAC: (A7100, A7105, A7106, A7107, A7108, A7109, A7110, A7127, A7128)

Name	Type	Description	Properties	Algorithms
Message Authentication Code with HMAC	MAC	Message Authentication Code using HMAC		HMAC-SHA-1: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-224: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-256: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-384: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-512: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-512/224: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA2-512/256: (A7101, A7111, A7112, A7113, A7114) HMAC-SHA3-224: (A7102) HMAC-SHA3-256: (A7102) HMAC-SHA3-384: (A7102) HMAC-SHA3-512: (A7102)
Signature Generation with RSA	DigSig-SigGen	Digital signature generation using RSA		RSA SigGen (FIPS186-5): (A7101, A7102, A7111, A7112, A7113, A7114)
Signature Generation with ECDSA	DigSig-SigGen	Digital signature generation using ECDSA		ECDSA SigGen (FIPS186-5): (A7101, A7102,

Name	Type	Description	Properties	Algorithms
				A7111, A7112, A7113, A7114)
Signature Generation with EdDSA	DigSig-SigGen	Digital signature generation using EdDSA		EDDSA SigGen: (A7098)
Signature Verification with RSA	DigSig-SigVer	Digital signature verification using RSA		RSA SigVer (FIPS186-5): (A7101, A7102, A7111, A7112, A7113, A7114)
Signature Verification with ECDSA	DigSig-SigVer	Digital signature verification using ECDSA		ECDSA SigVer (FIPS186-5): (A7101, A7102, A7111, A7112, A7113, A7114)
Signature Verification with EdDSA	DigSig-SigVer	Digital signature verification using EdDSA		EDDSA SigVer: (A7098)
Public Key Verification with ECDSA	AsymKeyPair-KeyVer	Key pair verification		ECDSA KeyVer (FIPS186-5): (A7101, A7111, A7112, A7113, A7114)
Key Pair Generation with RSA	AsymKeyPair-KeyGen CKG	Key pair generation using RSA		RSA KeyGen (FIPS186-5): (A7101, A7111, A7112, A7113, A7114) Asymmetric Cryptographic Key Generation (CKG): () Key Type: Asymmetric
Key Pair Generation with ECDSA	AsymKeyPair-KeyGen CKG	Key pair generation using ECDSA		ECDSA KeyGen (FIPS186-5): (A7101, A7111, A7112, A7113, A7114) Asymmetric Cryptographic Key Generation (CKG): ()

Name	Type	Description	Properties	Algorithms
				Key Type: Asymmetric
Key Pair Generation with EdDSA	AsymKeyPair-KeyGen CKG	Key pair generation using EdDSA		EDDSA KeyGen: (A7098) Asymmetric Cryptographic Key Generation (CKG): () Key Type: Asymmetric
Key Pair Generation with Safe Primes	AsymKeyPair-KeyGen CKG	Key pair generation using Safe Primes		Safe Primes Key Generation: (A7120) Asymmetric Cryptographic Key Generation (CKG): () Key Type: Asymmetric
Key Pair Verification with Safe Primes	AsymKeyPair-KeyVer	Key pair verification using Safe Primes		Safe Primes Key Verification: (A7120)
Key Derivation with KDA OneStep	KAS-56CKDF	Key derivation using KDA OneStep		KDA OneStep SP800-56Cr2: (A7121)
Key Derivation with KDA TwoStep	KAS-56CKDF	Key derivation using KDA TwoStep		KDA TwoStep SP800-56Cr2: (A7121)
Key Derivation with X9.42 KDF	KAS-135KDF	Key derivation using X9.42 KDF		KDF ANS 9.42: (A7101, A7102, A7111, A7112, A7113, A7114)
Key Derivation with X9.63 KDF	KAS-135KDF	Key derivation using X9.63 KDF		KDF ANS 9.63: (A7101, A7102, A7111, A7112, A7113, A7114)
Key Derivation with SSH KDF	KAS-135KDF	Key derivation using SSH KDF		KDF SSH: (A7115, A7118, A7122, A7123, A7124)
Key Derivation with HKDF	KAS-56CKDF	Key derivation using HKDF		KDA HKDF SP800-56Cr2: (A7096)

Name	Type	Description	Properties	Algorithms
Key Derivation with KBKDF	KBKDF	Key derivation using KBKDF		KDF SP800-108: (A7119)
TLS Key Derivation	KAS-135KDF	TLS 1.2 / 1.3 key derivation		TLS v1.2 KDF RFC7627: (A7101, A7111, A7112, A7113, A7114) TLS v1.3 KDF: (A7096)
Password-based Key Derivation	PBKDF	Password-based key derivation		PBKDF: (A7101, A7102, A7111, A7112, A7113, A7114)
Shared Secret Computation with DH	KAS-SSC	Shared secret computation using Diffie-Hellman	Compliance:IG D.F scenario 2(1)	KAS-FFC-SSC Sp800-56Ar3: (A7120)
Shared Secret Computation with ECDH	KAS-SSC	Shared secret computation using EC Diffie-Hellman	Compliance:IG D.F scenario 2(1)	KAS-ECC-SSC Sp800-56Ar3: (A7101, A7111, A7112, A7113, A7114)
Shared Secret Computation with RSA	KAS-SSC	Shared secret computation using RSA	Compliance:IG D.F scenario 1(1)	KAS-IFC-SSC: (A7101, A7111, A7112, A7113, A7114)
Asymmetric Encryption with RSA	KTS-Encap	Asymmetric encryption using RSA		KTS-IFC: (A7101, A7111, A7112, A7113, A7114)
Asymmetric Decryption with RSA	KTS-Decap	Asymmetric decryption using RSA		KTS-IFC: (A7101, A7111, A7112, A7113, A7114)

Table 9: Security Function Implementations

2.7 Algorithm Specific Information

2.7.1 AES-GCM IV

The module implements AES GCM for being used in the TLS v1.2 and v1.3 protocols. AES GCM IV generation is compliant with [FIPS140-3_IG] IG C.H for both protocols as follows:

- For TLS v1.2, the module offers the AES GCM implementation and IV generation is compliant with scenario 1.a of IG C.H and [RFC5288]. The module supports acceptable AES-GCM cipher suites from section 3.3.1 of [SP800-52rev2].

- For TLS v1.3, IV generation is compliant with scenario 5 of IG C.H and [RFC8446]. The protocol that provides this compliance is TLS 1.3, defined in RFC8446 of August 2018, using the cipher-suites that explicitly select AES GCM as the encryption/decryption cipher (Appendix B.4 of RFC8446). The module supports acceptable AES-GCM cipher suites from section 3.3.1 of [SP800-52rev2]. The module's implementation of AES GCM is used together with an application that runs outside the module's cryptographic boundary.

The IV generated in both scenarios is only used within the context of the TLS protocol implementation. The nonce explicit part of the IV does not exhaust the maximum number of possible values for a given session key. The design of the TLS protocol in this module implicitly ensures that the nonce explicit, or counter portion of the IV will not exhaust all its possible values.

Alternatively, the Crypto Officer can use the module's API to perform AES GCM encryption using internal IV generation. These IVs are always 96 bits and generated using the approved DRBG internal to the module's boundary, compliant to Scenario 2 of FIPS 140-3 IG C.H.

In case the module's power is lost and then restored, the key used for the AES-GCM encryption or decryption shall be redistributed.

2.7.2 AES-XTS

The AES algorithm in XTS mode can be only used for the cryptographic protection of data on storage devices, as specified in [SP800-38E]. It shall not be used for other purposes, such as the encryption of data in transit.

The length of a single data unit encrypted with the XTS-AES shall not exceed 2^{20} AES blocks, that is 16MB of data.

To meet the requirement stated in IG C.I, the module implements a check that ensures, before performing any cryptographic operation, that the two AES keys used in AES XTS mode are not identical. As the module does not generate symmetric keys, the check is performed when keys are input the service APIs.

Key_1 and Key_2 shall be generated and/or established independently according to the rules for component symmetric keys from NIST SP 800-133rev2, Sec. 6.3.

2.7.3 Key Derivation using SP 800-132 PBKDF2

The module provides password-based key derivation (PBKDF), compliant with SP800-132 and IG D.N. The module supports option 1a from section 5.4 of [SP800-132], in which the Master Key (MK) or a segment of it is used directly as the Data Protection Key (DPK).

In accordance with [SP800-132] and FIPS 140-3 IG D.N, the following requirements shall be met.

- Derived keys shall only be used in storage applications. The Master Key (MK) shall not be used for other purposes. The module only allows use of MK or DPK with at least 112 bits.
- The module only allows a portion of the salt with a length of at least 128 bits, which shall be generated randomly using the SP800-90Arev1 DRBG.
- 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. The minimum value allowed by the module is 1000.
- Passwords or passphrases, used as an input for the PBKDF, shall not be used as cryptographic keys.
- The module only allows password or passphrase with at least 8 characters, consisting of lower-case, upper-case, and numeric characters. The probability of guessing the value is estimated to be $1/62^8 = 10^{-}$

[14], which is less than 2^{-112}. If the password consists of only digits (worst case), the probability of guessing the value is estimated to be 10^{-8} which is less than 2^{-112}.

If any of these requirements are not met, the requested service is non-approved (see Non-Approved Services table in Section 4.4 Non-Approved Services).

2.7.4 SP 800-56ARev3 Assurances

The module offers DH and ECDH shared secret computation services compliant to the SP 800-56ARev3 and meeting IG D.F scenario 2 path (1). In order to meet the required assurances listed in section 5.6 of SP 800-56Arev3, the module shall be used together with an application that implements the "TLS protocol" and the following steps shall be performed:

1. The entity using the module, must use the module's "Key pair generation" service for generating DH/ECDH ephemeral keys. This meets the assurances required by key pair owner defined in the section 5.6.2.1 of SP 800-56ARev3.
2. As part of the module's shared secret computation (SSC) service, the module internally performs the public key validation on the peer's public key passed in as input to the SSC function. This meets the public key validity assurance required by the sections 5.6.2.2.1/5.6.2.2.2 of SP 800-56ARev3.

The module does not support static keys therefore the "assurance of peer's possession of private key" is not applicable.

2.7.5 SHA-3

The module implements the SHA-3 algorithms as both standalone and part of higher-level algorithms (in compliance with FIPS 140-3 IG C.C). As detailed in Section 2.6 Security Function Implementations with corresponding certificates, the cryptographic algorithms that use of SHA-3 include RSA signature generation and verification, ECDSA signature generation and verification, KBKDF, KDA HKDF, X9.63 KDF, X9.42 KDF, PBKDF, OneStep KDA, and HMAC. In addition, the implementation of the extendable output functions SHAKE128 and SHAKE256 were verified to have a standalone usage.

2.7.6 RSA Signature Generation and Signature Verification

The module provides RSA signature generation and signature verification compliant with IG C.F. The module supports RSA modulus lengths greater than or equal to 2048 bits for both signature generation and signature verification. The RSA signature generation and signature verification implementations have been tested for all module lengths available in CAVP testing: 2048, 3072, and 4096 bits.

2.7.7 RSA Key Generation

In compliance with IG C.E, the module generates RSA signature keys using an approved method of FIPS 186-5: generation of random primes that are provably prime. The RSA key generation has been tested for all module lengths available in CAVP testing: 2048, 3072, 4096, 6144, and 8192-bits. The CAVP certificate in table 2.5 indicates that the RSA key generating algorithm has been tested and validated for conformance to the methods in FIPS 186-5. The number of Miller-Rabin tests is consistent with the bit sizes of p and q from Table B.1 of FIPS 186-5.

2.7.8 Key Agreement

The module does not establish SSPs using an approved key agreement scheme (KAS). However, it does offer some or all of the underlying KAS cryptographic functionality to be used by an external operator/application as part of an approved KAS.

2.7.9 Compliance to SP 800-56Br2 Assurances

To comply with the assurances found in Section 6.4 of SP 800-56Br2, the module's approved key pair generation service (see Section 4.3) must be used to generate RSA key pairs, or the key pairs must be obtained from another FIPS-validated module. As part of this service, the module will internally perform the key pair validation of the generated public key.

The operator must use the EVP_PKEY_public_check() API to perform partial public key validation of the peer public key, complying with Section 6.4.2.2 of SP 800-56Br2. The operator must also confirm the peer's possession of private key by using any method specified in Section 6.4.2.3 of SP 800-56Br2.

2.7.10 Key Transport

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

2.7.11 SHA-1 Use

SHA-1 is only approved when used in approved modes for message digest, HMAC, HKDF, KDA OneStep/TwoStep, PBKDF, SSH KDF, ANS x9.42 KDF, KBKDF, Hash DRBG, HMAC DRBG, RSA OAEP.

2.8 RBG and Entropy

Cert Number	Vendor Name
E127	Cloud Linux Software, Inc. d/b/a TuxCare

Table 10: Entropy Certificates

Name	Type	Operational Environment	Sample Size	Entropy per Sample	Conditioning Component
Cloudlinux Inc., TuxCare division Userspace CPU Time Jitter RNG Entropy Source	Non-Physical	AlmaLinux OS 9.6 on GIGABYTE E163-S30-AAG1 on Intel® Xeon® Gold 5512U	256 bits	Full entropy	SHA3-256 (A7065), HMAC DRBG (A7090)

Table 11: Entropy Sources

The module complies with the Public Use Document for ESV certificate E127 by reading entropy data from the getrandom() function of the underlined Kernel, which corresponds to the GetEntropy() conceptual interface. The operational environment on the ESV certificate is identical to the operating system described in this document. There are no maintenance requirements for the entropy source.

As per the Public document of entropy certificate E127, the entropy source provides full entropy of 256 bits.

When the module needs random data for internal purposes it uses two separate instances of AES-256 CTR_DRBG DRBG based on use case. i.e., it uses the "private DRBG" accessed via RAND_priv_bytes() for asymmetric key generation, signature generation, or other SSP use cases and it uses the "public DRBG" accessed via RAND_bytes() when it needs to generate IV or other non-SSP use cases.

When an external caller needs the random data, it can access it via “Random Number Generation” service of the module and it has a choice to choose between Hash, HMAC or CTR DRBG listed in the algorithms table.

2.9 Key Generation

The module implements Cryptographic Key Generation (CKG, vendor affirmed) and key derivation methods as listed in Section 2.6.

When random values are required, they are obtained from the SP 800-90Ar1 approved DRBG, compliant with Section 4 of SP 800-133r2. This method does not use the value V as described in Additional Comment 2 of FIPS 140-3 IG D.H.

Intermediate key generation values are not output from the module and are explicitly zeroized after processing the service.

2.10 Key Establishment

The module implements shared secret computation methods, asymmetric encryption and decryption services using RSA with OAEP padding, as listed Section 2.6.

2.11 Industry Protocols

The module implements the SSH key derivation function for use in the SSH protocol (RFC 4253 and RFC 6668).

GCM with internal IV generation in the approved mode is compliant with versions 1.2 and 1.3 of the TLS protocol (RFC 5288 and 8446) and shall only be used in conjunction with the TLS protocol. Additionally, the module implements the TLS 1.2 and TLS 1.3 key derivation functions for use in the TLS protocol.

No parts of the SSH, TLS, or IKE protocols, other than those mentioned above, have been tested by the CAVP and CMVP.

3 Cryptographic Module Interfaces

3.1 Ports and Interfaces

Physical Port	Logical Interface(s)	Data That Passes
N/A	Data Input	API Input Parameters
N/A	Data Output	API Output Parameters
N/A	Control Input	API Function Calls
N/A	Status Output	API Return Codes, Error Queue

Table 12: Ports and Interfaces

The logical interfaces are the APIs through which the applications request services. These logical interfaces are logically separated from each other by the API design. The module does not implement a control output interface.

4 Roles, Services, and Authentication

4.1 Authentication Methods

The module does not support authentication methods.

4.2 Roles

Name	Type	Operator Type	Authentication Methods
Crypto Officer	Role	CO	None

Table 13: Roles

The module supports the Crypto Officer role only. This sole role is implicitly and always assumed by the operator of the module.

The module does not support multiple concurrent operators.

4.3 Approved Services

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Symmetric Encryption	Used to perform symmetric encryption of an entry plaintext	EVP_EncryptFinal_ex returns 1	Plaintext, AES key	Ciphertext	Symmetric Encryption with AES	Crypto Officer - AES key: W,E
Symmetric Decryption	Used to perform symmetric decryption of an entry ciphertext	EVP_DecryptFinal_ex returns 1	Ciphertext, AES key	Plaintext	Symmetric Decryption with AES	Crypto Officer - AES key: W,E
Authenticated Encryption	Used to perform authenticated encrypti	AES GCM: EVP_CIPHER_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_EncryptFinal_ex returns 1	Plaintext, AES key, IV	Ciphertext, MAC tag	Authenticated Encryption with AES	Crypto Officer - AES key: W,E - GCM

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
	on with AES					IV: W,E
Authenticated Decryption	Used to perform authenticated decryption with AES	AES GCM: EVP_CIPHER_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_DecryptFinal_ex returns 1	Ciphertext, AES key, MAC tag, IV	Plaintext or failure	Authenticated Decryption with AES	Crypto Officer - AES key: W,E - GCM IV: W,E
Message Authentication Code	Compute a MAC tag	HMAC: OSSL_MAC_PARAM_REDHAT_FIPS_INDICATOR_APPROVED; Others: EVP_MAC_final returns 1	Message, AES key or HMAC key	MAC tag	Message Authentication Code with AES Message Authentication Code with HMAC	Crypto Officer - HMAC key: W,E - AES key: W,E
Message Digest	Used to generate a SHA-1, SHA-2, or SHA-3/SHAKE message digest	Hash: EVP_DigestFinal returns 1; XOF: EVP_DigestFinalXOF returns 1	Message	Message digest	Message Digest	Crypto Officer
Key Derivation with KBKDF	Derive a key from a key-derivation key	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Key-derivation key	KBKDF Derived Key	Key Derivation with KBKDF	Crypto Officer - Key Derivation Key: W,E - KBKDF Derived Key: G,R

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Key Derivation with HKDF	Derive a key from a shared secret using HKDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	HKDF Derived Key	Key Derivation with HKDF	Crypto Officer - Shared Secret: W,E - HKDF Derived Key: G,R
Key Derivation with SSH KDF	Derive a key from a shared secret using SSH KDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	SSH Derived Key	Key Derivation with SSH KDF	Crypto Officer - Shared Secret: W,E - SSH Derived Key: G,R
Key Derivation with X9.63 KDF	Derive a key from a shared secret using X9.63 KDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	X9.63 Derived Key	Key Derivation with X9.63 KDF	Crypto Officer - Shared Secret: W,E - X9.63 Derived Key: G,R
Key Derivation with X9.42 KDF	Derive a key from a shared secret using X9.63 KDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	X9.42 Derived Key	Key Derivation with X9.42 KDF	Crypto Officer - Shared Secret: W,E - X9.42 Derived Key: G,R
Key Derivatio	Derive a key	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	KDA OneSte	Key Derivatio	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
n with KDA OneStep	from a shared secret using KDA OneStep			p Derived Key	n with KDA OneStep	- Shared Secret: W,E - KDA OneStep Derived Key: G,R
Key Derivation with KDA TwoStep	Derive a key from a shared secret using KDA TwoStep	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Shared secret	KDA TwoStep Derived Key	Key Derivation with KDA TwoStep	Crypto Officer - Shared Secret: W,E - KDA TwoStep Derived Key: G,R
TLS Key Derivation	Derive a key from a shared secret using TLS 1.2 KDF / TLS 1.3 KDF	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	TLS pre-master secret	TLS Derived Key	TLS Key Derivation	Crypto Officer - TLS pre-master secret: W,E - TLS master secret: G,E,Z - TLS Derived Key: G,R
Password-based Key Derivation	Derive a key from a password	EVP_KDF_REDHAT_FIPS_INDICATOR_APPROVED	Password or passphrase	PBKDF Derived Key	Password-based Key Derivation	Crypto Officer - Password or passphr

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						ase: W,E - PBKDF Derived Key: G,R
Shared Secret Computation	Compute a shared secret	KAS-IFC-SSC: EVP_PKEY_REDHAT_FIPS_INDICATOR_APPROVED; KAS-FFC-SSC, KAS-ECC-SSC: EVP_PKEY_derive returns 1	DH private key, DH public key; EC private key, EC public key; RSA public key, RSA private key	Shared secret	Shared Secret Computation with DH Shared Secret Computation with ECDH Shared Secret Computation with RSA	Crypto Officer - DH private key: W,E - DH public key: W,E - EC public key: W,E - EC private key: W,E - RSA public key: W,E - RSA private key: W,E - Shared Secret: G,R
Signature Generation	Generate a digital signature	RSA: OSSL_RH_FIPSINDICATOR_APPROVED and EVP_SIGNATURE_REDHAT_FIPS_INDICATOR_APPROVED; ECDSA:	Message, private key	Signature	Signature Generation with RSA Signature Generati	Crypto Officer - RSA private key: W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
		OSSL_RH_FIPSINDICATOR_APPROVED; EDDSA: EVP_PKEY_sign returns 1			on with ECDSA Signature Generation with EdDSA	- EC private key: W,E - EdDSA private key: W,E
Signature Verification	Verify a digital signature	RSA: OSSL_RH_FIPSINDICATOR_APPROVED and EVP_SIGNATURE_REDHAT_FIPS_INDICATOR_APPROVED; ECDSA: OSSL_RH_FIPSINDICATOR_APPROVED; EDDSA: EVP_PKEY_verify returns 1	Message, public key, signature	Pass/fail	Signature Verification with RSA Signature Verification with ECDSA Signature Verification with EdDSA	Crypto Officer - RSA public key: W,E - EC public key: W,E - EdDSA public key: W,E
Key Pair Generation	Generate a key pair	EVP_PKEY_generate returns 1	Group or Curve or Modulus bits	Module-generated DH private key, Module-generated DH public key or Module-generated EC private key, Modul	Key Pair Generation with RSA Key Pair Generation with ECDSA Key Pair Generation with Safe Primes Key Pair Generation with EdDSA	Crypto Officer - Module-generated RSA private key: G,R - Module-generated DH private key: G,R -

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
				e-generated EC public or Module-generated EdDSA private key, Module-generated EdDSA public key or key Module-generated RSA private key, Module-generated RSA public key		Module-generated RSA public key: G,R - Module-generated DH public key: G,R - Module-generated EC private key: G,R - Module-generated EC public key: G,R - Module-generated EdDSA private key: G,R - Module-generat

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						ed EdDSA public key: G,R - Intermediate key generation value: G,E,Z
Key Pair Verification	Verify a key pair	EVP_PKEY_public_check or EVP_PKEY_private_check or EVP_PKEY_check returns 1	Key pair	Pass/fail	Key Pair Verification with Safe Primes Public Key Verification with ECDSA	Crypto Officer - DH public key: W,E - EC public key: W,E - DH private key: W,E - EC private key: W,E
Random Number Generation	Generate random bytes	EVP_RAND_generate returns 1	Output length	Random bytes	Random Number Generation	Crypto Officer - Entropy input: W,E - DRBG internal state (V value, C value): G,W,E

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- DRBG internal state (V value, Key): G,W,E - DRBG seed: G,E
Asymmetric Encryption	Perform RSA-based encryption (compliant with SP 800-56B Rev. 2))	EVP_PKEY_REDHAT_FIPS_INDICATOR_APPROVED	RSA public key, plaintext	Cipher text	Asymmetric Encryption with RSA	Crypto Officer - RSA public key: W,E
Asymmetric Decryption	Perform RSA-based decryption (compliant with SP 800-56B Rev. 2))	EVP_PKEY_REDHAT_FIPS_INDICATOR_APPROVED	RSA private key, cipher text	Plaintext	Asymmetric Decryption with RSA	Crypto Officer - RSA private key: W,E
Show status	Show the current status of the module	None	N/A	Module status	None	Crypto Officer
Show module name and version	Show module name and the version of the module	None	N/A	Name and version information	None	Crypto Officer

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
Self-test	Perform CASTs and integrity test	None	N/A	Pass/fail result of self-tests	Message Digest Message Authentication Code with AES Message Authentication Code with HMAC Symmetric Encryption with AES Symmetric Decryption with AES Authenticated Encryption with AES Authenticated Decryption with AES Signature Generation with RSA Signature Generation with ECDSA Signature Generation with	

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					EdDSA Signature Verification with RSA Signature Verification with ECDSA Signature Verification with EdDSA Key Derivation with KBKDF Key Derivation with KDA OneStep Key Derivation with HKDF Key Derivation with X9.42 KDF Key Derivation with X9.63 KDF Key Derivation with SSH KDF TLS Key Derivation Password	

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
					-based Key Derivation Random Number Generation Shared Secret Computation with DH Shared Secret Computation with ECDH Asymmetric Encryption with RSA	
Zeroization	Zeroize SSPs.	None	Any SSP	N/A	None	Crypto Officer - AES key: Z - GCM IV: Z - HMAC key: Z - Module-generated RSA private key: Z - Module-generated RSA

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						public key: Z - RSA private key: Z - RSA public key: Z - Module - generated DH private key: Z - Module - generated DH public key: Z - DH private key: Z - DH public key: Z - Module - generated EC private key: Z - Module - generated EC public key: Z - EC private

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						key: Z - EC public key: Z - Module - generated EdDSA private key: Z - Module - generated EdDSA public key: Z - EdDSA private key: Z - EdDSA public key: Z - Key Derivation Key: Z - KBKDF Derived Key: Z - HKDF Derived Key: Z - SSH Derived Key: Z - X9.63 Derived

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						Key: Z - X9.42 Derived Key: Z - Password or passphrase: Z - PBKDF Derived Key: Z - KDA OneStep Derived Key: Z - KDA TwoStep Derived Key: Z - TLS pre-master secret: Z - TLS master secret: Z - TLS Derived Key: Z - Shared Secret: Z - Entropy input: Z - DRBG seed: Z

Name	Description	Indicator	Inputs	Outputs	Security Functions	SSP Access
						- DRBG internal state (V value, C value): Z - DRBG internal state (V value, Key): Z - Intermediate key generation value: Z

Table 14: Approved Services

The module provides services to operators that assume the available role. All services are described in detail in the API documentation (manual pages). The convention below applies when specifying the access permissions (types) that the service has for each SSP.

- **Generate (G):** The module generates or derives the SSP.
- **Read (R):** The SSP is read from the module (e.g. the SSP is output).
- **Write (W):** The SSP is updated, imported, or written to the module.
- **Execute (E):** The module uses the SSP in performing a cryptographic operation.
- **Zeroize (Z):** The module zeroizes the SSP.

To interact with the module, a calling application must use the EVP API layer provided by OpenSSL. This layer will delegate the request to the FIPS provider, which will in turn perform the requested service. Additionally, this EVP API layer can be used to retrieve the approved service indicator for the module. The redhat_ossl_query_fipsindicator() function indicates whether an EVP API function is approved.

4.4 Non-Approved Services

Name	Description	Algorithms	Role
AES GCM (external IV)	Authenticated Encryption	AES GCM (external IV)	CO
HMAC (< 112-bit keys)	Compute a MAC tag	HMAC (< 112-bit keys)	CO

Name	Description	Algorithms	Role
Key derivation	Derive a key from a key-derivation key or a shared secret	KBKDF, KDA OneStep, HKDF, ANS X9.42 KDF, ANS X9.63 KDF (< 112-bit keys) KDA OneStep (SHAKE128, SHAKE256) ANS X9.42 KDF (SHAKE128, SHAKE256) ANS X9.63 KDF (SHA-1, SHAKE128, SHAKE256) SSH KDF (SHA-512/224, SHA-512/256, SHA-3, SHAKE128, SHAKE256) TLS 1.2 KDF (SHA-1, SHA-224, SHA-512/224, SHA-512/256, SHA-3) TLS 1.3 KDF (SHA-1, SHA-224, SHA-512, SHA-512/224, SHA-512/256, SHA-3)	CO
PBKDF (<112-bit keys)	Derive a key from a password	PBKDF2 (< 8 characters password; < 128 salt length; < 1000 iterations; < 112-bit keys)	CO
Signature generation	Generate a signature	RSA and ECDSA (pre-hashed message) RSA-PSS (invalid salt length)	CO
Signature verification	Verify a signature	RSA and ECDSA (pre-hashed message) RSA-PSS (invalid salt length)	CO

Table 15: Non-Approved Services

The table above lists the non-approved services in this module, the algorithms involved and the roles that can request the service. In this table, CO specifies the Crypto Officer role.

4.5 External Software/Firmware Loaded

The module does not load external software or firmware.

5 Software/Firmware Security

5.1 Integrity Techniques

The integrity of the module is verified by comparing a HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value embedded in the fips.so file that was computed at build time. The module performs a KAT for the HMAC SHA-256 algorithm in order to test its proper operation before performing the checksum of the module file.

5.2 Initiate on Demand

Integrity tests are performed as part of the pre-operational self-tests, which are executed when the module is initialized. The integrity test may be invoked on-demand by unloading and subsequently re-initializing the module, or by calling the OSSL_PROVIDER_self_test function. This will perform (among others) the software integrity test.

6 Operational Environment

6.1 Operational Environment Type and Requirements

Type of Operational Environment: Modifiable

How Requirements are Satisfied:

Any SSPs contained within the module are protected by the process isolation and memory separation mechanisms, and only the module has control over these SSPs.

6.2 Configuration Settings and Restrictions

The module shall be installed as stated in Section 11 Life-Cycle Assurance. If properly installed, the operating system provides process isolation and memory protection mechanisms that ensure appropriate separation for memory access among the processes on the system. Each process has control over its own data and uncontrolled access to the data of other processes is prevented.

Instrumentation tools like the ptrace system call, gdb and strace, userspace live patching, as well as other tracing mechanisms offered by the Linux environment such as ftrace or systemtap, shall not be used in the operational environment. The use of any of these tools implies that the cryptographic module is running in a non-validated operational environment.

7 Physical Security

The module is comprised of software only and therefore this section is not applicable.

8 Non-Invasive Security

This module does not implement any non-invasive security mechanism and therefore this section is not applicable.

9 Sensitive Security Parameters Management

9.1 Storage Areas

Storage Area Name	Description	Persistence Type
RAM	Temporary storage for SSPs used by the module as part of service execution. SSPs are stored until they are zeroized by the operator (using a zeroization call or removing power from the module) or zeroized automatically	Dynamic

Table 16: Storage Areas

The module does not perform persistent storage of SSPs. The SSPs are temporarily stored in the RAM in plaintext form.

9.2 SSP Input-Output Methods

Name	From	To	Format Type	Distribution Type	Entry Type	SFI or Algorithm
API input parameters	Operator calling application (TOEPP)	Cryptographic module	Plaintext	Manual	Electronic	
API output parameters	Cryptographic module	Operator calling application (TOEPP)	Plaintext	Manual	Electronic	

Table 17: SSP Input-Output Methods

The module only supports SSP entry and output to and from the calling application running on the same operational environment. This corresponds to manual distribution, electronic entry/output ("CM Software to/from App via TOEPP Path") per FIPS 140-3 IG 9.5.A Table 1.

9.3 SSP Zeroization Methods

Zeroization Method	Description	Rationale	Operator Initiation
Free cipher handle	Zeroizes the SSPs contained within the cipher handle.	By calling the appropriate zeroization functions: AES key: EVP_CIPHER_CTX_free and EVP_MAC_CTX_free; HMAC key: EVP_MAC_CTX_free; Key-derivation key: EVP_KDF_CTX_free; Shared secret: EVP_KDF_CTX_free; Password: EVP_KDF_CTX_free; KBKDF Derived Key:EVP_KDF_CTX_free; HKDF Derived Key: EVP_KDF_CTX_free; TLS pre-master secret, TLS master secret, TLS Derived Key: EVP_KDF_CTX_free; SSH Derived Key:	By calling the cipher related zeroization API

Zeroization Method	Description	Rationale	Operator Initiation
		EVP_KDF_CTX_free; X9.63 Derived Key: EVP_KDF_CTX_free; X9.42 Derived Key: EVP_KDF_CTX_free; PBKDF Derived Key: EVP_KDF_CTX_free; KDA OneStep Derived Key: EVP_KDF_CTX_free; KDA TwoStep Derived Key: EVP_KDF_CTX_free; Entropy input: EVP_RAND_CTX_free; DRBG internal state (V value, Key), DRBG internal state (V value, C value): EVP_RAND_CTX_free; DH public & private key: EVP_PKEY_free; EC public & private key: EVP_PKEY_free; RSA public & private key: EVP_PKEY_free; EdDSA public & private key: EVP_PKEY_free	
Automatic	Automatically zeroized by the module when no longer needed	Memory occupied by SSPs is overwritten with zeroes, which renders the SSP values irretrievable	N/A
Module Reset	De-allocates the volatile memory used to store SSPs	Volatile memory used by the module is overwritten within nanoseconds when power is removed	By unloading and reloading the module

Table 18: SSP Zeroization Methods

The application that uses the module is responsible for the appropriate zeroization of SSPs. The module provides key allocation and destruction functions, which overwrites the memory occupied by the SSP´s information with zeros before its deallocation.

All data output is inhibited during zeroization.

9.4 SSPs

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
AES key	AES key used for encryption, decryption, and computing MAC tags	AES-XTS: 256, 512 bits; Other modes: 128, 192, 256 bits - AES-XTS: 128, 256 bits; Other modes: 128, 192, 256 bits	Symmetric key - CSP			Symmetric Encryption with AES Symmetric Decryption with AES Message Authentication Code with AES Authenticated Encryption with AES

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						Authenticated Decryption with AES
GCM IV	Initialization vector used for authenticated encryption and authenticated decryption	96 bit - N/A	Initialization vector - PSP			Authenticated Encryption with AES Authenticated Decryption with AES
HMAC key	HMAC key used for computing MAC tags	112-524288 bits - 112-256 bits	Symmetric key - CSP			Message Authentication Code with HMAC
Module-generated RSA private key	RSA private key generated by the module	2048-15360 bits - 112-256 bits	Private key - CSP	Key Pair Generation with RSA		Key Pair Generation with RSA
Module-generated RSA public key	RSA public key generated by the module	2048-15360 bits - Key pair generation: 112-256 bits	Public key - PSP	Key Pair Generation with RSA		Key Pair Generation with RSA
RSA private key	RSA private key written to the module	2048-16384 bits - 112-256 bits	Private key - CSP			Signature Generation with RSA Shared Secret Computation with RSA Asymmetric Decryption with RSA
RSA public key	RSA public key written to the module	Signature verification: 2048-16384 bits - Signature verification: 112-256 bits	Public key - PSP			Signature Verification with RSA Shared Secret Computation with RSA Asymmetric Encryption with RSA

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
Module-generated DH private key	DH private key generated by the module	2048-8192 bits - 112-200 bits	Private key - CSP	Key Pair Generation with Safe Primes		Key Pair Generation with Safe Primes
Module-generated DH public key	DH public key generated by the module	2048-8192 bits - 112-200 bits	Public key - PSP	Key Pair Generation with Safe Primes		Key Pair Generation with Safe Primes
DH private key	DH private key written to the module	2048-8192 bits - 112-200 bits	Private key - CSP			Shared Secret Computation with DH Key Pair Verification with Safe Primes
DH public key	DH public key written to the module	2048-8192 bits - 112-200 bits	Public key - PSP			Shared Secret Computation with DH Key Pair Verification with Safe Primes
Module-generated EC private key	EC private key generated by the module	P-224, P-256, P-384, P-521 bits - 112, 128, 192, 256 bits	Private key - CSP	Key Pair Generation with ECDSA		Key Pair Generation with ECDSA
Module-generated EC public key	EC public key generated by the module	P-224, P-256, P-384, P-521 bits - 112, 128, 192, 256 bits	Public key - PSP	Key Pair Generation with ECDSA		Key Pair Generation with ECDSA
EC private key	EC private key written the module and used by ECDSA and ECDH	P-224, P-256, P-384, P-521 bits - 112, 128, 192, 256 bits	Private key - CSP			Shared Secret Computation with ECDH Signature Generation with ECDSA Public Key

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
						Verification with ECDSA
EC public key	EC public key written the module and used by ECDSA and ECDH	P-224, P-256, P-384, P-521 bits - 112, 128, 192, 256 bits	Public key - PSP			Signature Verification with ECDSA Shared Secret Computation with ECDH Public Key Verification with ECDSA
Module-generated EdDSA private key	EdDSA private key generated by the module	Ed25519, Ed448 - Ed25519: 128 bits of security strength; Ed448: 256 bits of security strength	Private key - CSP	Key Pair Generation with EdDSA		Key Pair Generation with EdDSA
Module-generated EdDSA public key	EdDSA public key generated by the module	Ed25519, Ed448 - Ed25519: 128 bits of security strength; Ed448: 256 bits of security strength	Public key - PSP	Key Pair Generation with EdDSA		Key Pair Generation with EdDSA
EdDSA private key	EdDSA private key used for digital signature generation	Ed25519, Ed448 - Ed25519 128 bits of security strength, Ed448 256 bits of security strength	Private key - CSP			Signature Generation with EdDSA

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
EdDSA public key	EdDSA public key used for digital signature verification	Ed25519, Ed448 - Ed25519 128 bits of security strength, Ed448 256 bits of security strength	Public key - PSP			Signature Verification with EdDSA
Key Derivation Key	Symmetric key used to derive symmetric keys	112-4096 bits - 112-256 bits	Symmetric key - CSP			Key Derivation with KBKDF
KBKDF Derived Key	Symmetric key derived from a key-derivation key	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with KBKDF		Key Derivation with KBKDF
HKDF Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with HKDF		Key Derivation with HKDF
SSH Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with SSH KDF		Key Derivation with SSH KDF
X9.63 Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with X9.63 KDF		Key Derivation with X9.63 KDF
X9.42 Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with X9.42 KDF		Key Derivation with X9.42 KDF
Password or passphrase	Password or passphrase used by PBKDF to derive symmetric keys	8-128 characters - N/A	Password - CSP			Password-based Key Derivation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
PBKDF Derived Key	Key derived from PBKDF password/passphrase during key derivation	112-4096 bits - 112-256 bits	Symmetric key - CSP	Password-based Key Derivation		Password-based Key Derivation
KDA OneStep Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with KDA OneStep		Key Derivation with KDA OneStep
KDA TwoStep Derived Key	Symmetric key derived from a shared secret	112-4096 bits - 112-256 bits	Symmetric key - CSP	Key Derivation with KDA TwoStep		Key Derivation with KDA TwoStep
TLS pre-master secret	Pre-master secret used in Transport Layer Security (TLS) network protocol key derivation	112-4096 bits - 112-256 bits	Shared Secret - CSP		Shared Secret Computation with DH Shared Secret Computation with ECDH	TLS Key Derivation
TLS master secret	Master secret used in Transport Layer Security (TLS) network protocol key derivation function for deriving the TLS Derived Key	112-4096 bits - 112-256 bits	Secret - CSP	TLS Key Derivation		TLS Key Derivation
TLS Derived Key	Derived key used in Transport Layer Security (TLS) network protocol	112-4096 bits - 112-256 bits	Derived secret - CSP	TLS Key Derivation		
Shared Secret	Shared secret generated by ECDH/DH/RSA shared secret computation	224-8912 bits - 112-256 bits	Shared Secret - CSP		Shared Secret Computation with DH Shared Secret	Key Derivation with KDA OneStep Key Derivation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
					Computation with ECDH Shared Secret Computation with RSA	with KDA TwoStep Key Derivation with X9.42 KDF Key Derivation with X9.63 KDF Key Derivation with SSH KDF Key Derivation with HKDF Shared Secret Computation with DH Shared Secret Computation with ECDH Shared Secret Computation with RSA
Entropy input	Entropy input string used to seed the DRBG (IG D.L compliant)	128-384 bits - 128-256 bits	Entropy Input - CSP			Random Number Generation
DRBG seed	DRBG seed derived from entropy input (IG D.L compliant)	CTR_DRBG: 256, 320, 384 bits; Hash_DRBG: 440, 888 bits; HMAC_DRBG: 160, 256, 512 bits - CTR_DRBG: 128, 192, 256 bits; HMAC_DRBG,	Seed - CSP	Random Number Generation		Random Number Generation

Name	Description	Size - Strength	Type - Category	Generated By	Established By	Used By
		Hash_DRBG: 128, 256 bits				
DRBG internal state (V value, C value)	Internal state of the Hash_DRBG (IG D.L compliant)	880, 1776 bits - 128, 256 bits	Internal state - CSP	Random Number Generation		Random Number Generation
DRBG internal state (V value, Key)	Internal state of the CTR_DRBG and HMAC_DRBG (IG D.L compliant)	CTR_DRBG: 256, 320, 384 bits; HMAC_DRBG: 320, 512, 1024 bits - CTR_DRBG: 128, 192, 256 bits; HMAC_DRBG: 128, 256 bits	Internal state - CSP	Random Number Generation		Random Number Generation
Intermediate key generation value	Intermediate key pair generation value generated during key generation and key derivation services (SP 800-133 Rev. 2 Section 4, 5.1, and 5.2)	112-15360 bits - 112-256 bits	Intermediate value - CSP	Key Pair Generation with RSA Key Pair Generation with ECDSA Key Pair Generation with Safe Primes Key Pair Generation with EdDSA		Key Pair Generation with RSA Key Pair Generation with ECDSA Key Pair Generation with Safe Primes Key Pair Generation with EdDSA

Table 19: SSP Table 1

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
AES key	API input parameters	RAM:Plaintext	From service invocation until	Free cipher handle	

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
			cipherhandle is freed	Module Reset	
GCM IV	API input parameters API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	
HMAC key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	
Module-generated RSA private key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated RSA public key:Paired With Intermediate key generation value:Generated From
Module-generated RSA public key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated RSA private key:Paired With Intermediate key generation value:Generated From
RSA private key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	RSA public key:Paired With
RSA public key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	RSA private key:Paired With
Module-generated DH private key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated DH public key:Paired With Intermediate key generation value:Generated From
Module-generated DH public key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated DH private key:Paired With Intermediate key

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					generation value:Generated From
DH private key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	DH public key:Paired With
DH public key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	DH private key:Paired With
Module-generated EC private key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated EC public key:Paired With Intermediate key generation value:Generated From
Module-generated EC public key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated EC private key:Paired With Intermediate key generation value:Generated From
EC private key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	EC public key:Paired With
EC public key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	EC private key:Paired With
Module-generated EdDSA private key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated EdDSA public key:Paired With Intermediate key generation value:Generated From
Module-generated EdDSA public key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Module-generated EdDSA private key:Paired With Intermediate key

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					generation value:Generated From
EdDSA private key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	EdDSA public key:Paired With
EdDSA public key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	EdDSA private key:Paired With
Key Derivation Key	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	KBKDF Derived Key:Derives
KBKDF Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Key-derivation key:Derived From
HKDF Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
SSH Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
X9.63 Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
X9.42 Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
Password or passphrase	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	PBKDF Derived Key:Derives

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
PBKDF Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Password or passphrase:Derived From
KDA OneStep Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
KDA TwoStep Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	Shared secret:Derived From
TLS pre-master secret	API input parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Automatic Module Reset	TLS master secret:Generates DH private key:Established By DH public key:Established By EC private key:Established By EC public key:Established By
TLS master secret		RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Automatic Module Reset	TLS pre-master secret:Generated From TLS Derived Key:Derives
TLS Derived Key	API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	TLS pre-master secret:Derived From TLS master secret:Derived From
Shared Secret	API input parameters API output parameters	RAM:Plaintext	From service invocation until cipherhandle is freed	Free cipher handle Module Reset	DH private key:Established By DH public key:Established By EC private key:Established By EC public key:Established By HKDF Derived Key:Derives KDA OneStep Derived

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					Key:Derives KDA TwoStep Derived Key:Derives TLS Derived Key:Derives SSH Derived Key:Derives X9.63 Derived Key:Derives X9.42 Derived Key:Derives RSA private key:Established By RSA public key:Established By
Entropy input		RAM:Plaintext	From generation until DRBG seed is created	Automatic Module Reset	DRBG seed:Derives
DRBG seed		RAM:Plaintext	While the DRBG is instantiated	Automatic Module Reset	Entropy input:Derived From DRBG internal state (V value, C value):Generates DRBG internal state (V value, Key):Generates
DRBG internal state (V value, C value)		RAM:Plaintext	From DRBG instantiation until DRBG termination	Free cipher handle Module Reset	DRBG seed:Generated From
DRBG internal state (V value, Key)		RAM:Plaintext	From DRBG instantiation until DRBG termination	Free cipher handle Module Reset	DRBG seed:Generated From
Intermediate key generation value		RAM:Plaintext	From service invocation until cipherhandle is freed	Automatic	DH private key:Generates DH public key:Generates EC private key:Generates EC public key:Generates RSA private

Name	Input - Output	Storage	Storage Duration	Zeroization	Related SSPs
					key:Generates RSA public key:Generates EdDSA private key:Generates EdDSA public key:Generates

Table 20: SSP Table 2

9.5 Transitions

The SHA-1 algorithm as implemented by the module will be non-approved for all purposes, starting January 1, 2031.

10 Self-Tests

10.1 Pre-Operational Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
HMAC-SHA2-256 (A7101)	256-bits key	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity test of the shared library component of the module. Verified by comparing an HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value embedded in the fips.so file that was computed at build time.
HMAC-SHA2-256 (A7111)	256-bits key	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity test of the shared library component of the module. Verified by comparing an HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value embedded in the fips.so file that was computed at build time.
HMAC-SHA2-256 (A7112)	256-bits key	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity test of the shared library component of the module. Verified by comparing an HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value embedded in the fips.so file that was computed at build time.
HMAC-SHA2-256 (A7113)	256-bits key	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity test of the shared library component of the module. Verified by comparing an HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value embedded in the fips.so file that was computed at build time.
HMAC-SHA2-256 (A7114)	256-bits key	Message authentication	SW/FW Integrity	Module becomes operational and services are available for use	Integrity test of the shared library component of the module. Verified by comparing an HMAC SHA-256 value calculated at run time with the HMAC SHA-256 value

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details
					embedded in the fips.so file that was computed at build time.

Table 21: Pre-Operational Self-Tests

The pre-operational software integrity tests are performed automatically when the module is initialized, before the module transitions into the operational state. While the module is executing the self-tests, services are not available, and data output (via the data output interface) is inhibited until the tests are successfully completed. The module transitions to the operational state only after the pre-operational self-tests are passed successfully.

Prior the first use, a CAST is executed for the algorithms used in the Pre-operational Self-Tests.

10.2 Conditional Self-Tests

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
SHA-1 (A7101)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA-1 (A7111)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA-1 (A7112)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA-1 (A7113)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA-1 (A7114)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-512 (A7101)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-512 (A7111)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
						before the integrity test
SHA2-512 (A7112)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-512 (A7113)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA2-512 (A7114)	24-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A7101)	256-bit key	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A7111)	256-bit key	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A7112)	256-bit key	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A7113)	256-bit key	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
HMAC-SHA2-256 (A7114)	256-bit key	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test
SHA3-256 (A7102)	32-bit message	KAT	CAST	Module becomes operational	Message digest	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM - Encrypt (A7100)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7105)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7106)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7107)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7108)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7109)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7110)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7127)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Encrypt (A7128)	256-bit key and 96-bit IV, encrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7100)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-GCM - Decrypt (A7105)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7106)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7107)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7108)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7109)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7110)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7127)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-GCM - Decrypt (A7128)	256-bit key and 96-bit IV, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7094)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7095)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
AES-ECB (A7099)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7115)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7118)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7122)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7123)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
AES-ECB (A7124)	128-bit key, decrypt	KAT	CAST	Module becomes operational	Symmetric operation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-5) (A7101)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-5) (A7102)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-5) (A7111)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-5) (A7112)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA SigGen (FIPS186-5) (A7113)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigGen (FIPS186-5) (A7114)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7101)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7102)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7111)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7112)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7113)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
RSA SigVer (FIPS186-5) (A7114)	PKCS#1 v1.5 with SHA-256 and 2048-bit key	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-5) (A7101)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-5) (A7102)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
ECDSA SigGen (FIPS186-5) (A7111)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-5) (A7112)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-5) (A7113)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigGen (FIPS186-5) (A7114)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7101)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7102)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7111)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7112)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7113)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
ECDSA SigVer (FIPS186-5) (A7114)	SHA-256 and P-224	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
EDDSA SigGen (A7098)	Ed25519; Ed4488	KAT	CAST	Module becomes operational	Digital signature generation	Test runs at power-on before the integrity test
EDDSA SigVer (A7098)	Ed25519; Ed4488	KAT	CAST	Module becomes operational	Digital signature verification	Test runs at power-on before the integrity test
KDF SP800-108 (A7119)	HMAC-SHA2-256 in counter mode and 128-bit input key	KAT	CAST	Module becomes operational	Key Derivation with KBKDF	Test runs at power-on before the integrity test
KDA OneStep SP800-56Cr2 (A7121)	SHA2-224 and 448-bit input secret	KAT	CAST	Module becomes operational	Shared secret key derivation	Test runs at power-on before the integrity test
KDA TwoStep SP800-56Cr2 (A7121)	SHA2-224 and 448-bit input secret	KAT	CAST	Module becomes operational	Shared secret key derivation	Test runs at power-on before the integrity test
KDA HKDF SP800-56Cr2 (A7096)	SHA2-256 and 48-bit secret	KAT	CAST	Module becomes operational	Shared secret key derivation	Test runs at power-on before the integrity test
KDF ANS 9.42 (A7101)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.42 (A7102)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.42 (A7111)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.42 (A7112)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF ANS 9.42 (A7113)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.42 (A7114)	AES-128 KW and SHA-1 and 160-bit input secret	KAT	CAST	Module becomes operational	ANS X9.42 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7101)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7102)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7111)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7112)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7113)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF ANS 9.63 (A7114)	SHA2-256 and 192-bit input secret	KAT	CAST	Module becomes operational	ANS X9.63 key derivation	Test runs at power-on before the integrity test
KDF SSH (A7115)	SHA-1 and 1056-bit input secret	KAT	CAST	Module becomes operational	SSH KDF key derivation	Test runs at power-on before the integrity test
KDF SSH (A7118)	SHA-1 and 1056-bit input secret	KAT	CAST	Module becomes operational	SSH KDF key derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
KDF SSH (A7122)	SHA-1 and 1056-bit input secret	KAT	CAST	Module becomes operational	SSH KDF key derivation	Test runs at power-on before the integrity test
KDF SSH (A7123)	SHA-1 and 1056-bit input secret	KAT	CAST	Module becomes operational	SSH KDF key derivation	Test runs at power-on before the integrity test
KDF SSH (A7124)	SHA-1 and 1056-bit input secret	KAT	CAST	Module becomes operational	SSH KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A7101)	SHA2-256 and 384-bit input secret	KAT	CAST	Module becomes operational	TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A7111)	SHA2-256 and 384-bit input secret	KAT	CAST	Module becomes operational	TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A7112)	SHA2-256 and 384-bit input secret	KAT	CAST	Module becomes operational	TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A7113)	SHA2-256 and 384-bit input secret	KAT	CAST	Module becomes operational	TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.2 KDF RFC7627 (A7114)	SHA2-256 and 384-bit input secret	KAT	CAST	Module becomes operational	TLS v1.2 KDF key derivation	Test runs at power-on before the integrity test
TLS v1.3 KDF (A7096)	SHA2-256, expand and extract mode	KAT	CAST	Module becomes operational	TLS v1.3 KDF key derivation	Test runs at power-on before the integrity test
PBKDF (A7101)	SHA2-256 with 24 characters password, 288-	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
	bit salt, 4096 iterations					
PBKDF (A7102)	SHA2-256 with 24 characters password, 288-bit salt, 4096 iterations	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test
PBKDF (A7111)	SHA2-256 with 24 characters password, 288-bit salt, 4096 iterations	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test
PBKDF (A7112)	SHA2-256 with 24 characters password, 288-bit salt, 4096 iterations	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test
PBKDF (A7113)	SHA2-256 with 24 characters password, 288-bit salt, 4096 iterations	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test
PBKDF (A7114)	SHA2-256 with 24 characters password, 288-bit salt, 4096 iterations	KAT	CAST	Module becomes operational	Password-based Key Derivation	Test runs at power-on before the integrity test
Counter DRBG (A7097)	AES-128 with derivation function and prediction resistance	KAT	CAST	Module becomes operational	Instantiate; Generate; Reseed (compliant to SP 800-90A Rev. 1 Section 11.3)	Test runs at power-on before the integrity test
Hash DRBG (A7097)	SHA2-256 and prediction resistance	KAT	CAST	Module becomes operational	Instantiate; Generate; Reseed (compliant to SP 800-90A Rev. 1 Section 11.3)	Test runs at power-on before the integrity test
HMAC DRBG (A7097)	HMAC-SHA-1 and prediction resistance	KAT	CAST	Module becomes operational	Instantiate; Generate; Reseed (compliant to SP	Test runs at power-on before the integrity test

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
					800-90A Rev. 1 Section 11.3)	
KAS-FFC-SSC Sp800-56Ar3 (A7120)	ffdhe2048	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A7101)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A7111)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A7112)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A7113)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
KAS-ECC-SSC Sp800-56Ar3 (A7114)	P-256	KAT	CAST	Module becomes operational	Shared secret computation	Test runs at power-on before the integrity test
Safe Primes Key Generation (A7120)	N/A	PCT	PCT	Key pair generation is successful	SP 800-56A Rev. 3 Section 5.6.2.1.4	Key pair generation
RSA KeyGen (FIPS186-5) (A7101)	PKCS#1 v1.5 with SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
RSA KeyGen (FIPS186-5) (A7111)	PKCS#1 v1.5 with SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
RSA KeyGen (FIPS186-5) (A7112)	PKCS#1 v1.5 with SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
RSA KeyGen (FIPS186-5) (A7113)	PKCS#1 v1.5 with SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
RSA KeyGen (FIPS186-5) (A7114)	PKCS#1 v1.5 with SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A7101)	SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A7111)	SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A7112)	SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A7113)	SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
ECDSA KeyGen (FIPS186-5) (A7114)	SHA-256	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
EDDSA KeyGen (A7098)	PCT using Ed25519 and Ed448	PCT	PCT	Key pair generation is sucessful	Signature generation and verification	Key pair generation
KTS-IFC (A7101)	OAEP with 2048-bit key	KAT	CAST	Module becomes operational	Key encapsulation and un-encapsulation	Test runs at power-on before the integrity test
KTS-IFC (A7111)	OAEP with 2048-bit key	KAT	CAST	Module becomes operational	Key encapsulation and un-encapsulation	Test runs at power-on before the integrity test
KTS-IFC (A7112)	OAEP with 2048-bit key	KAT	CAST	Module becomes operational	Key encapsulation and un-encapsulation	Test runs at power-on

Algorithm or Test	Test Properties	Test Method	Test Type	Indicator	Details	Conditions
						before the integrity test
KTS-IFC (A7113)	OAEP with 2048-bit key	KAT	CAST	Module becomes operational	Key encapsulation and un-encapsulation	Test runs at power-on before the integrity test
KTS-IFC (A7114)	OAEP with 2048-bit key	KAT	CAST	Module becomes operational	Key encapsulation and un-encapsulation	Test runs at power-on before the integrity test

Table 22: Conditional Self-Tests

Data output through the data output interface is inhibited during the conditional self-tests. The module does not return control to the calling application until the tests are completed. If any of these tests fails, the module transitions to the error state (Section 10.4 Error States).

10.3 Periodic Self-Test Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
HMAC-SHA2-256 (A7101)	Message authentication	SW/FW Integrity	On Demand	Manually
HMAC-SHA2-256 (A7111)	Message authentication	SW/FW Integrity	On Demand	Manually
HMAC-SHA2-256 (A7112)	Message authentication	SW/FW Integrity	On Demand	Manually
HMAC-SHA2-256 (A7113)	Message authentication	SW/FW Integrity	On Demand	Manually
HMAC-SHA2-256 (A7114)	Message authentication	SW/FW Integrity	On Demand	Manually

Table 23: Pre-Operational Periodic Information

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
SHA-1 (A7101)	KAT	CAST	On Demand	Manually
SHA-1 (A7111)	KAT	CAST	On Demand	Manually
SHA-1 (A7112)	KAT	CAST	On Demand	Manually
SHA-1 (A7113)	KAT	CAST	On Demand	Manually
SHA-1 (A7114)	KAT	CAST	On Demand	Manually
SHA2-512 (A7101)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
SHA2-512 (A7111)	KAT	CAST	On Demand	Manually
SHA2-512 (A7112)	KAT	CAST	On Demand	Manually
SHA2-512 (A7113)	KAT	CAST	On Demand	Manually
SHA2-512 (A7114)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A7101)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A7111)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A7112)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A7113)	KAT	CAST	On Demand	Manually
HMAC-SHA2-256 (A7114)	KAT	CAST	On Demand	Manually
SHA3-256 (A7102)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7100)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7105)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7106)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7107)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7108)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7109)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7110)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7127)	KAT	CAST	On Demand	Manually
AES-GCM - Encrypt (A7128)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7100)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
AES-GCM - Decrypt (A7105)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7106)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7107)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7108)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7109)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7110)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7127)	KAT	CAST	On Demand	Manually
AES-GCM - Decrypt (A7128)	KAT	CAST	On Demand	Manually
AES-ECB (A7094)	KAT	CAST	On Demand	Manually
AES-ECB (A7095)	KAT	CAST	On Demand	Manually
AES-ECB (A7099)	KAT	CAST	On Demand	Manually
AES-ECB (A7115)	KAT	CAST	On Demand	Manually
AES-ECB (A7118)	KAT	CAST	On Demand	Manually
AES-ECB (A7122)	KAT	CAST	On Demand	Manually
AES-ECB (A7123)	KAT	CAST	On Demand	Manually
AES-ECB (A7124)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A7101)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A7102)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A7111)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A7112)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
RSA SigGen (FIPS186-5) (A7113)	KAT	CAST	On Demand	Manually
RSA SigGen (FIPS186-5) (A7114)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7101)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7102)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7111)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7112)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7113)	KAT	CAST	On Demand	Manually
RSA SigVer (FIPS186-5) (A7114)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A7101)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A7102)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A7111)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A7112)	KAT	CAST	On Demand	Manually
ECDSA SigGen (FIPS186-5) (A7113)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
ECDSA SigGen (FIPS186-5) (A7114)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7101)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7102)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7111)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7112)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7113)	KAT	CAST	On Demand	Manually
ECDSA SigVer (FIPS186-5) (A7114)	KAT	CAST	On Demand	Manually
EDDSA SigGen (A7098)	KAT	CAST	On Demand	Manually
EDDSA SigVer (A7098)	KAT	CAST	On Demand	Manually
KDF SP800-108 (A7119)	KAT	CAST	On Demand	Manually
KDA OneStep SP800-56Cr2 (A7121)	KAT	CAST	On Demand	Manually
KDA TwoStep SP800-56Cr2 (A7121)	KAT	CAST	On Demand	Manually
KDA HKDF SP800-56Cr2 (A7096)	KAT	CAST	On Demand	Manually
KDF ANS 9.42 (A7101)	KAT	CAST	On Demand	Manually
KDF ANS 9.42 (A7102)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
KDF ANS 9.42 (A7111)	KAT	CAST	On Demand	Manually
KDF ANS 9.42 (A7112)	KAT	CAST	On Demand	Manually
KDF ANS 9.42 (A7113)	KAT	CAST	On Demand	Manually
KDF ANS 9.42 (A7114)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7101)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7102)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7111)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7112)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7113)	KAT	CAST	On Demand	Manually
KDF ANS 9.63 (A7114)	KAT	CAST	On Demand	Manually
KDF SSH (A7115)	KAT	CAST	On Demand	Manually
KDF SSH (A7118)	KAT	CAST	On Demand	Manually
KDF SSH (A7122)	KAT	CAST	On Demand	Manually
KDF SSH (A7123)	KAT	CAST	On Demand	Manually
KDF SSH (A7124)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A7101)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A7111)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A7112)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A7113)	KAT	CAST	On Demand	Manually
TLS v1.2 KDF RFC7627 (A7114)	KAT	CAST	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
TLS v1.3 KDF (A7096)	KAT	CAST	On Demand	Manually
PBKDF (A7101)	KAT	CAST	On Demand	Manually
PBKDF (A7102)	KAT	CAST	On Demand	Manually
PBKDF (A7111)	KAT	CAST	On Demand	Manually
PBKDF (A7112)	KAT	CAST	On Demand	Manually
PBKDF (A7113)	KAT	CAST	On Demand	Manually
PBKDF (A7114)	KAT	CAST	On Demand	Manually
Counter DRBG (A7097)	KAT	CAST	On Demand	Manually
Hash DRBG (A7097)	KAT	CAST	On Demand	Manually
HMAC DRBG (A7097)	KAT	CAST	On Demand	Manually
KAS-FFC-SSC Sp800-56Ar3 (A7120)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7101)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7111)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7112)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7113)	KAT	CAST	On Demand	Manually
KAS-ECC-SSC Sp800-56Ar3 (A7114)	KAT	CAST	On Demand	Manually
Safe Primes Key Generation (A7120)	PCT	PCT	On Demand	Manually

Algorithm or Test	Test Method	Test Type	Period	Periodic Method
RSA KeyGen (FIPS186-5) (A7101)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A7111)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A7112)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A7113)	PCT	PCT	On Demand	Manually
RSA KeyGen (FIPS186-5) (A7114)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A7101)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A7111)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A7112)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A7113)	PCT	PCT	On Demand	Manually
ECDSA KeyGen (FIPS186-5) (A7114)	PCT	PCT	On Demand	Manually
EDDSA KeyGen (A7098)	PCT	PCT	On Demand	Manually
KTS-IFC (A7101)	KAT	CAST	On demand	Manually
KTS-IFC (A7111)	KAT	CAST	On demand	Manually
KTS-IFC (A7112)	KAT	CAST	On demand	Manually
KTS-IFC (A7113)	KAT	CAST	On demand	Manually
KTS-IFC (A7114)	KAT	CAST	On demand	Manually

Table 24: Conditional Periodic Information

10.4 Error States

Name	Description	Conditions	Recovery Method	Indicator
Error	If the module fails any of the self-tests, the module enters the error state. In the error state, the module immediately stops functioning and ends the application process	Software integrity test failure CAST failure PCT failure	Module reinitialization	Software integrity test failure CAST failure: OSSL_PROV_PARAM_STATUS is set to 0. Module will not load; PCT failure: module is aborted

Table 25: Error States

If the module fails any of the self-tests, the module enters the error state. In the error state, the module immediately stops functioning and ends the application process. Consequently, the data output interface is inhibited, and the module no longer accepts inputs or requests (as the module is no longer running).

10.5 Operator Initiation of Self-Tests

Both conditional and pre-operational self-tests can be executed on-demand by unloading and subsequently re-initializing the module, or by calling the OSSL_PROVIDER_self_test function. The pair-wise consistency tests can be invoked on demand by requesting the key pair generation service.

11 Life-Cycle Assurance

11.1 Installation, Initialization, and Startup Procedures

11.1.1 Configuration of the Operating Environment

Before the openssl-3.2.2-7.el9_6.tuxcare.1 RPM package is installed, the AlmaLinux OS 9.6 system must operate in the FIPS validated configuration. This can be achieved by:

- Adding the fips=1 option to the kernel command line during the system installation. During the software selection stage, do not install any third-party software.
- Switching the system into the FIPS validated configuration after the installation. Execute the fips-mode-setup --enable command. Restart the system.

In both cases, the Crypto Officer must verify the system operates in the FIPS validated configuration by executing the fips-mode-setup --check command, which should output "FIPS mode is enabled."

If the module is not installed, initialized, and configured according to this section, the module is in a non-compliant state. If the module is in a non-compliant state, it can be placed into the compliant state by un-initializing and uninstalling the module and then installing, initializing, and configuring the module according to this section.

11.1.2 Delivery of the module

On the GIGABYTE E163-S30-AAG1 hardware platform with the Intel® Xeon® Gold 5512U processor, the module is delivered through the following RPM packages:

- openssl-3.2.2-7.el9_6.tuxcare.1.x86_64

11.2 Administrator Guidance

The binaries of the module are contained in the RPM packages for delivery, listed in section 11.1.2. After the RPM package is installed, the Crypto Officer must execute the openssl list -providers command. This command should display the base/default and FIPS providers as follows:

```
Providers
base
        name: OpenSSL Base Provider
        version: 3.2.2
        status: active
default
        name: OpenSSL Default Provider
        version: 3.2.2
        status: active
fips
        name: TuxCare OpenSSL FIPS Provider
        version: 3.2.2-f9f9d133a30b6eb5
        status: active
```

The cryptographic boundary consists only of the FIPS provider as listed. If any other OpenSSL or third-party provider is invoked, the user is not interacting with the module specified in this Security Policy.

11.3 Non-Administrator Guidance

There is no non-administrator guidance.

11.4 End of Life

As the module does not persistently store SSPs, secure sanitization of the module consists of unloading the module. This will zeroize all SSPs in volatile memory. Then, if desired, the openssl-3.2.2-7.el9_6.tuxcare.1 RPM package can be uninstalled from the AlmaLinux OS 9.6 system.

12 Mitigation of Other Attacks

12.2 Attack List

RSA and ECDSA timing attacks.

12.2 Mitigation Effectiveness

Certain cryptographic subroutines and algorithms are vulnerable to timing analysis. The module claims mitigation of timing-based side-channel attacks implementing two methods: Constant-time Implementations and Numeric Blinding:

- Constant-time Implementations protect cryptographic implementations in the module against timing cryptanalysis ensuring that the variations in execution time for different cryptographic algorithms cannot be traced back to the key, CSP or secret data.
- Numeric Blinding protects the RSA 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 factor which is provided as an input to the decryption/signature operation which discarded once the operation results in an output. This makes it difficult for attackers to attempt timing attacks making impossible correlating execution time to the RSA/ECDSA key.

Appendix A. Glossary and abbreviations

AES	Advanced Encryption Standard
API	Application Programming Interface
CAST	Cryptographic Algorithm Self-Test
CAVP	Cryptographic Algorithm Validation Program
CCM	Counter with Cipher Block Chaining-Message Authentication Code
CKG	Cryptographic Key Generation
CMVP	Cryptographic Module Validation Program
CSP	Critical Security Parameter
CTR	Counter Mode
DF	Derivation Function
DRBG	Deterministic Random Bit Generator
ECC	Elliptic Curve Cryptography
FFC	Finite Field Cryptography
FIPS	Federal Information Processing Standards Publication
GCM	Galois Counter Mode
GMAC	Galois Counter Mode Message Authentication Code
HMAC	Hash Message Authentication Code
KAS	Key Agreement Scheme
KAT	Known Answer Test
KW	AES Key Wrap
MAC	Message Authentication Code
NIST	National Institute of Science and Technology
OFB	Output Feedback
PAA	Processor Algorithm Acceleration
PAI	Processor Algorithm Implementation
PBKDF2	Password-based Key Derivation Function v2
PKCS	Public-Key Cryptography Standards
PCT	Pairwise Consistency Test
PR	Prediction Resistance
RNG	Random Number Generator
RSA	Rivest, Shamir, Addleman

SHA	Secure Hash Algorithm
SHS	Secure Hash Standard
SSC	Shared Secret Computation
SSH	Secure Shell
SSP	Sensitive Security Parameter
TLS	Transport Layer Security
TOEPP	Tested Operational Environment's Physical Perimeter
XTS	XEX-based Tweaked-codebook mode with cipher text Stealing

Appendix B. References

ANS X9.42-2001

Public Key Cryptography for the Financial Services Industry: Agreement of Symmetric Keys Using Discrete Logarithm Cryptography

2001

https://webstore.ansi.org/standards/ascx9/ansix9422001

ANS X9.63-2001

Public Key Cryptography for the Financial Services Industry, Key Agreement and Key Transport Using Elliptic Curve Cryptography

2001

https://webstore.ansi.org/standards/ascx9/ansix9632001

FIPS 140-3

FIPS PUB 140-3 - Security Requirements for Cryptographic Modules

March 22, 2019

https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf

FIPS 140-3 IG

Implementation Guidance for FIPS PUB 140-3 and the Cryptographic Module Validation Program

April 18, 2025

https://csrc.nist.gov/csrc/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf

FIPS 180-4

Secure Hash Standard (SHS)

August 2015

https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

FIPS 186-5

Digital Signature Standard (DSS)

February 3, 2023

https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.186-5.pdf

FIPS 197

Advanced Encryption Standard

November 26, 2001

https://csrc.nist.gov/publications/fips/fips197/fips-197.pdf

FIPS 198-1

The Keyed Hash Message Authentication Code (HMAC)

July 2008

https://csrc.nist.gov/publications/fips/fips198-1/FIPS-198-1_final.pdf

FIPS 202	**SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions** August 2015 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.202.pdf
RFC 3526	**More Modular Exponential (MODP) Diffie-Hellman groups for Internet Key Exchange (IKE)** May 2003 https://www.ietf.org/rfc/rfc3526.txt
RFC 4253	**The Secure Shell (SSH) Transport Layer Protocol** January 2006 https://www.ietf.org/rfc/rfc4253.txt
RFC 5288	**AES Galois Counter Mode (GCM) Cipher Suites for TLS** August 2008 https://www.ietf.org/rfc/rfc5288.txt
RFC 6668	**SHA-2 Data Integrity Verification for the Secure Shell (SSH) Transport Layer Protocol** July 2012 https://www.ietf.org/rfc/rfc6668.txt
RFC 7919	**Negotiated Finite Field Diffie-Hellman Ephemeral Parameters for Transport Layer Security (TLS)** August 2016 https://www.ietf.org/rfc/rfc7919.txt
RFC 8446	**The Transport Layer Security (TLS) Protocol Version 1.3** August 2018 https://www.ietf.org/rfc/rfc8446.txt
SP 800-140B Rev. 1	**NIST Special Publication 800-140B - CMVP Security Policy Requirements** June 25, 2025 https://csrc.nist.gov/projects/cmvp/sp800-140b
SP 800-38A	**Recommendation for Block Cipher Modes of Operation Methods and Techniques** December 2001 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a.pdf

SP 800-38A Addendum	**Recommendation for Block Cipher Modes of Operation: Three Variants of Ciphertext Stealing for CBC Mode** October 2010 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38a-add.pdf
SP 800-38B	**Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication** May 2005 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38b.pdf
SP 800-38C	**Recommendation for Block Cipher Modes of Operation: the CCM Mode for Authentication and Confidentiality** July 2007 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38c.pdf
SP 800-38D	**Recommendation for Block Cipher Modes of Operation: Galois/Counter Mode (GCM) and GMAC** November 2007 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38d.pdf
SP 800-38E	**Recommendation for Block Cipher Modes of Operation: The XTS AES Mode for Confidentiality on Storage Devices** January 2010 https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-38e.pdf
SP 800-38F	**Recommendation for Block Cipher Modes of Operation: Methods for Key Wrapping** December 2012 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-38F.pdf
SP 800-52 Rev. 2	**Guidelines for the Selection, Configuration, and Use of Transport Layer Security (TLS) Implementations** August 2019 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r2.pdf
SP 800-56A Rev. 3	**Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography** April 2018 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Ar3.pdf

SP 800-56C Rev. 2	**Recommendation for Key-Derivation Methods in Key-Establishment Schemes** August 2020 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-56Cr2.pdf
SP 800-90A Rev. 1	**Recommendation for Random Number Generation Using Deterministic Random Bit Generators** June 2015 https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf